Orchestration in the Internet of Things - Towards a fully Integrated Solution

Jonas Burster - 20165136

jburst16@student.aau.dk

xx.xx.2019

Contents

1 Introduction

1.1 Defining Key Terms

The IoT landscape is often hard to understand, especially as names are defined loosely and inconsistently. The same holds true for the cloud and terms around cluster technology. This section will explain the network topolgy, key terms and their concepts as the foundation for the following report.

Network Topology

Figure 1 shows the network topolgy used in this report. The focus will be on the management and security of the device edge as well as its interaction with the IoT devices themselves. The upper layers two layers, the cloud and infrastructure edge, and their connections are all operated by big telecompanies and content providers. The cloud is on demand available data storage and processing power. The infrastructure edge is operated by the network providers and is mainly used to provide access to the network to households and companies. It provides some services, e.g. to improve network characteristics such as to lower latency. But, devices operating in this layer are not directly related to IoT and thus excluded in this report.

Figure 1: Network Topology inspired by [1].

The device edge is operated by normal households and other non network-related

businesses. It is respsonsible for the control and data plane of the constrained devices (explained below). The data produced from these devices can be aggregated and processed in this layer. The benefits of doing so are discussed in chapter xxx.

Constrained Device

Constraint devices are the elementary part of the IoT making up the "things" or devices[2]. They can have three main purposes. Either sensing or actuating (or both), where sensing is the passive action of measuring the environment (e.g. a motion detector) and actuating is the active action of influencing the environment (e.g. control of pressure in test tube). Or, finally, they can be smart objects enhancing the interaction between other smart objects and people.
They are usually defined by their limitation, mainly, small computing power (CPU, RAM, storage etc.) and limited power supply and operate in constrained network[1]. In summary, it is fair to assume that many constrained devices have very limited computing power, operate on battery in constrained networks using protocols like BLE and ZigBee.

Smart devices

Smart devices are often not clearly defined in the academic literature, in this report I will use the definition given by Poslad[3], to clearly divide them from constrained devices. They are traditional computing devices and "tend to be multi purpose ICT devices"[3], examples are mobile phones (smart phones) or tablets. They connect to the rest of the infrastructure directly but are free to move between networks. They also often rely on battery power and, importantly, are mainly end user devices. This has important privacy implication when using their computing power for computations on the edge. **(I should pick this up somewhere in**

[1]See [2] page 5 for more information.

the report or go more in detail here).

IoT Gateway

IoT Gateways are the connection between smart devices and a network, this could be a local network or even the Internet. They facilitate inter-network and intra-network communications and because smart devices and especially constrained devices often communicate via wireless and non-Internet protocols, IoT gateways often translate protocols "between wireless sensor networks [...] traditional communication networks"[4]. In recent years IoT Gateways have become a major field of interest and new research. As these devices got more powerful, developers started using them for preprocessing and datagathering locally at the edge. In conclusion, IoT gateways can take a wide variety of forms. They can be simple L3 routers or more powerful devices. Importantly, they are situated at the edge of a network and act as bridge between two autonomous systems.

Edge node

Edge nodes are defined as nodes "that act as an end user portal for communication with other nodes in cluster computing"[5]. It operates on the edge, it must be able to run containers, has far inferior processing power compared to servers and can (only) run as a worker node (also known as a minion in kubernetes)[6]. Note that, this is in contrast to IBM which defines edge nodes as ingress traffic nodes in a cluster[7]. In this report an edge node is a powerful IoT gateway situated at the edge of the network. Because of new and exciting developments it will not be restricted by the type of software it can run (it can thus be a master node as well), but rather by its network topology and mobility. It is a node without reliable and fast connection to the rest of the network, does not need a static IP

and is (somewhat) portable[2]. Reoccurring themes are the restrictions compared to cloud clusters, which are smaller processing power, no permament connection to the backbone network and greater portability.

Edge cluster

The term edge cluster is closely related to cluster technologies like Kubernetes or Mesos. Edge clusters are computers powerful enough to run a full control plane of the cluster technology. However, there is no clear definition of what constitutes to an edge cluster in neither the academic literature nor the industry. They are more powerful edge nodes, often packing additional hardware like GPUs or SSDs. However, emerging technologies like tiny builds of a full Kubernetes cluster like K3s from Rancher Labs[8] make this distinction between edge nodes and edge clusters increasingly difficult [3].

Because of their similarities, in this report the distinction between edge nodes and clusters will be solely made up on the cluster topology separating them on an architectural level.

2 Extended State of the Art

What is the purpose of this section?

problem area: what is actually the problem with the current system? hint into solution

competitor anaylsis: compare different approaches to solve the management and security of IoT gateways as well as solutions inside the individual categories.

[2]Comparing edge nodes to cloud nodes is also common academic practice[2] but not considered at this point.

[3]Rancher provides a 40MB Kubernetes binary and claims that 500MB of RAM is sufficient it stable.

2.1 Problem Area

IoT has seen a rapid growth over the last few years. According to IoT Analytics the total number of IoT devices is set to surpass the total number of other connected devices around 2021 [9]. Further, most IoT devices will be used in WPAN [4] and WLAN[5]. Traditionally these devices connect to the enterprise or service provider core networks through a gateway. This gateway was mainly a router operating at L3[6] to route packets and translate between different types of network protocols. [10]. However, according to Dejan Bosanac, a senior software engineer at Red Hat in the field of cloud messaging and IoT platforms, due to their proximity to the sensors and the end user, these devices have three main advantages over the cloud: "Low latency, availability and locality" [11]. Because of this advantage system designers started to construct hybrid systems, in which the gateway plays an active role in the data processing pipeline. This architectural style of carrying out substantial amount of computation and storage at the edge is called "fog computing" [12] and a simple example can be seen in figure 2.

Figure 2: IoT Device Setup [1].

But fog computing does not come without its drawbacks. Depending on the protocol edge devices need to be close to their peers and slaves and physically

[4]Wireless Private Area Networks includes technologies like Zigbee,Z-wave and Bluteooth
[5]Wireless Local Area Networks includes mainly Wi-Fi
[6]L3 stands for Layer 3, the networking layer in the OSI model.

accessible for maintenance. Which also poses a major security risk as they could be accessed by malicious intruders. The software maintenance is another critical aspects. Often IoT and edge devices are not update and patched with critical consequences. The "2016 Dyn cyberattack" used IoT devices like residential gateways, smart fridges, baby phones ect. to bring down the DNS-Servers operated by Dyn making large part of the Internet unaccessible for hours[13]. The authors also stress that "large number of IoT devices are accessible over public Internet" and that "security (if considered at all) is often an afterthought in the architecture of many wide spread IoT devices"[13].
The question is then, how can manage and secure those devices. In this report, I will solely be concerned with the software aspect, which can mitigate some effects of exposing physical hardware to more accessible places.
Many challenges facing edge devices today have already been solved, although in a slight different context: The cloud[11]. In the cloud

2.2 Candidate Technologies

Traditionally, edge devices were isolated gateways mainly forwarding traffic from their slaves to the cloud. This meant, all processing and storing was done in the cloud and thus the software of the gateways didn't need much management or maintenance. With the increase of putting processing power at the edge, so called fog computing, the software on the gateways expanded to include business logic as well as routing. Today gateways are an integral part of the data flow, pre-processing data and storing data and forwarding only certain data. This new software needs much more maintenance and management as it is part of the business logic and is part of the API offering of the solution. The software running on these devices needs to be managed and, especially with the advent of IIoT, the number of IoT gateways is set to increase dramatically. This chapter will compare

three different solutions for IoT gateways: Traditional gateway solutions, orchestrated gateway solutions, decentralized but containerized solutions and, finally, centralized containerized solutions.

First, https://www.pac-online.com/sites/pac-online.com/files/upload$_path/PDFs/Thema_des_Monat$
//blog.bosch−si.com/bosch−iot−suite/lessons−learned−using−kubernetes−in − iot − deployments/

2.3 Traditional Gateway Solutions

Tradition gateway solutions are all solutions which do not interact with data or only in a very limitted amount. The software running these devices is usually pre-installed by an OEM and hard to update and maintain. In such system each gateway is isolated and information is only accumulated and processed in the cloud. Multiple gateways do not have a shared or centralized control plane and thus need to be managed independently and locally. This is the base or reference case, where the gateway is a pure L3 device, and the least interesting one from a managenement and security perspective. Technologies in this category are not considered to do fog computing, but because of their prevalence are included here for completeness.

2.4 Smart Gateway Solutions

https://www.bosch-iot-suite.com/service/gateway-software/

Smart gateway solutions are similar in the way they interact with Fog computing No central control plane

Technologies in this category have a control plane and a data plane, however they are not centralized. without containerization are solutions with a centralized control plane provide many of the features cloud native (with containerization) do. Orchestrated solutions on the other hand offer a central configuration point, so a

shared control plane between multiple gateways. These solutions were specifically developed for a distributed and large scale IoT environemnt. In many ways they are comparable to the solutions described in the next two subchapters, but are not developed with containerization at its foundation. The OSGi technology[14] is a aliance driven project from the Open Services Gateway initiative. It is a set of specifications (with reference implemetation and tests) for a dynamic modular system based on so called bundles, third party software, running on the Java Virtual Machine (JVM)[7]. The solution consists of a layered model shown in figure 3. The

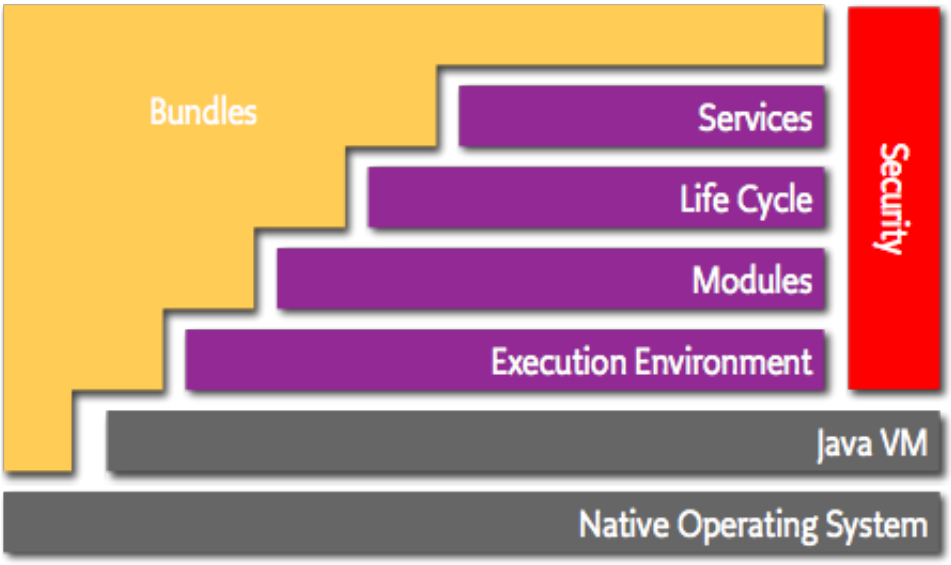

Figure 3: From the official OSGI documentation[15].

Service layer interconnects the bundles makingit possible for them to communicate via plain old java objects (POJO).The Life-Cycle layer handles the the state of the application (start, stop, update and uninstall). The Modules layer defines how an application can import and export code and the Execution Environemnt defines

[7]This means it is possible to use other languages apart from Java which can run in a JVM, e.g. Kotlin.

which methods and classes are available in a specific environemnt. Finally, the Security layer encompases all other layers and handles for example code authentication, the digital signing of jar files, file access restrictions, certificates and more. To the authors best knowledge there are currently six frameworks implementing the OSGi model. One such solution is the Bosch IoT Gateway Software[16]. In a blog entry from 2015 Bosch compares the OSGi technology to other gateway solutions and says it "is the only one with clearly defined specs and an open specification process behind them"[17]. Boschs solution is proprietary and tailer made for edge-computing devices with IIoT in mind[18]. It runs on Linux, Windows, mac OS, Android, and VxWorks and according to Bosch more than 40 different gateway devices[16]. The software is stable at major version 9 and still under heavy development. Because of these reasons, it is used as an examplary solution. Figure 4 shows where Boschs IoT solution is situated in the IoT environment. Bosch provides the OSGi framework implementation for the gateways and additional features for the cloud to ease the management of the gateways and store the accumulated data. A strong argument for this solution is the number of protocols

Figure 4: From the official Bosch documentation [16].

it supports (BLE, ZigBee and MQTT, just to name a few) as well as its maturity.

2.5 Orchestrated Gateway Solutions

IoT cluster solutions

Isolated k8s cluster: Solution k3s

Kubernetes cluster where edge is part of the cloud: kubeedge

2.6 Cloud Native Gateway Solutions

Edged: an agent that runs on edge nodes and manages containerized applications. EdgeHub: a web socket client responsible for interacting with Cloud Service for the edge computing (like Edge Controller as in the KubeEdge Architecture). This includes syncing cloud-side resource updates to the edge, and reporting edge-side host and device status changes to the cloud. CloudHub: a web socket server responsible for watching changes at the cloud side, caching and sending messages to EdgeHub. EdgeController: an extended kubernetes controller which manages edge nodes and pods metadata so that the data can be targeted to a specific edge node. EventBus: a MQTT client to interact with MQTT servers (mosquitto), offering publish and subscribe capabilities to other components. ServiceBus: a HTTP client to interact with HTTP servers (REST), offering HTTP client capabilities to components of cloud to reach HTTP servers running at edge. DeviceTwin: responsible for storing device status and syncing device status to the cloud. It also provides query interfaces for applications. MetaManager: the message processor between edged and edgehub. It is also responsible for storing/retrieving metadata to/from a lightweight database (SQLite).

https://randomnerdtutorials.com/esp32-over-the-air-ota-programming/

file:///home/joe/Downloads/DDoSSecurity containers: Mirai attack: weak passwords, needs shell Counter: Docker, binary files, non root user, traffic analysis via istio

References

[1] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg, "Exploiting smart e-health gateways at the edge of healthcare internet-of-things: A fog computing approach", *Future Generation Computer Systems*, vol. 78, pp. 641–658, 2018.

[2] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks", no. 7228, May 2014, `http://www.rfc-editor.org/rfc/rfc7228.txt`, ISSN: 2070-1721. [Online]. Available: `http://www.rfc-editor.org/rfc/rfc7228.txt`.

[3] S. Poslad, *Ubiquitous computing: smart devices, environments and interactions*. John Wiley & Sons, 2011.

[4] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, "Iot gateway: Bridging-wireless sensor networks into internet of things", in *2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, Ieee, 2010, pp. 347–352.

[5] M. Rouse, *What is edge node? - definition from whatis.com*, `https://whatis.techtarget.com/definition/edge-node`, (Accessed on 04/11/2019).

[6] K. Community, *Nodes - kubernetes*, `https://kubernetes.io/docs/concepts/architecture/nodes/`, (Accessed on 04/11/2019).

[7] *Ibm cloud container service edge nodes - ibm cloud blog*, `https://www.ibm.com/blogs/bluemix/2017/11/ibm-cloud-container-service-edge-nodes/`, (Accessed on 04/11/2019).

[8] R. Labs, *K3s - lightweight kubernetes | k3s*, `https://k3s.io/`, (Accessed on 04/11/2019).

[9] *State of the iot 2018: Number of iot devices now at 7b – market accelerating*, https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/, (Accessed on 04/01/2019).

[10] S. Lee, M. Bae, and H. Kim, "Future of iot networks: A survey", *Applied Sciences*, vol. 7, no. 10, p. 1072, 2017.

[11] D. Bosanac, *Introducing kubernetes iot edge working group - cloud native computing foundation*, https://www.cncf.io/community/webinars/introducing-kubernetes-iot-edge-working-group/, (Accessed on 04/01/2019), Mar. 2019.

[12] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things", in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, ACM, 2012, pp. 13–16.

[13] K. Angrishi, "Turning internet of things (iot) into internet of vulnerabilities (iov): Iot botnets", *arXiv preprint arXiv:1702.03681*, 2017.

[14] O. Alliance, *What is osgi?*, https://www.osgi.org/developer/what-is-osgi/, (Accessed on 04/23/2019).

[15] ——, *Architecture*, https://www.osgi.org/developer/architecture/, (Accessed on 04/23/2019).

[16] Bosch, *Bosch iot gateway software: Edge computing for iot - bosch iot suite*, https://www.bosch-iot-suite.com/service/gateway-software/, (Accessed on 04/22/2019).

[17] S. Ferber, *The future of osgi in the internet of things - bosch connectedworld blogthefutur*, https://blog.bosch-si.com/developer/the-future-of-osgi-in-the-internet-of-things/, (Accessed on 04/23/2019).

[18] K. HACKBARTH, *Osgi for iot solutions: A perfect match - bosch connectedworld blog*, `https://blog.bosch-si.com/bosch-iot-suite/osgi-for-iot-solutions-a-perfect-match/`, (Accessed on 04/23/2019).