Empowering the Internet of Things with Software Defined Networking

[WHITE PAPER]

Pedro Martinez-Julia, Antonio F. Skarmeta

{pedromj,skarmeta}@um.es

Abstract

The flexibility and general programmability offered by the Software Defined Networking (SDN) technology has supposed a disruption in the evolution of the network. It offers enormous benefits to network control and opens new ways of communication by defining powerful but simple switching elements (forwarders) that can use any single field of a packet or message to determine the outgoing port to which it will be forwarded. Such benefits can be applied to the Internet of Things (IoT) and thus resolve some of the main challenges it exposes, such as the ability to let devices connected to heterogeneous networks to communicate each other. In the present document we describe a general model to integrate SDN and IoT so that heterogeneous communications are achieved. However, it exposes other (simpler) challenges must be resolved, evaluated, and validated against current and future solutions before the design of the integrated approach can be finished.

1. Introduction

The spread of the Internet of Things (IoT) concept has imposed new complex requirements to both networking and internetworking schemes in current and future networks, specially the Internet. To make it real, networks must welcome heterogeneity, not just in devices but also in networking behavior and underlying protocols. This is because each IoT object (device or *thing*) has been configured, or even designed, to accomplish specific objectives. Moreover, the entire environment where some objects are deployed is usually designed with a specific objective. At the end, IoT implies the broad interconnection of several heterogeneous networks, the objects that compose them, the environments where they are running, the upper and lower layer protocols they are using, and even the disparate objectives they have.

One of the most widespread approach to build such view is the adaptation of a common protocol to all objects and environments. This has been the role of IP for the Internet and it is often proposed as the solution for IoT, specially with the advent of IPv6. However, such huge enterprise is far from being real and it even has its own challenges and drawbacks. The main negative impact is the loss of the intrinsic heterogeneity in the network level. Objects and protocols have specific designs because they have to cover with specific requirements and objectives, so forcing them to fit with a common and singular protocol is not a good option for most object designers.

From the opposite perspective of the network comes the Software Defined Networking (SDN) approach. It encompasses the widespread programmability of network elements, both endpoints and intermediate elements. The first step towards this huge enterprise has been the definition of general schemes to separate control and data planes in switching and routing elements. This has been accompanied with the simplification of intermediate network elements, which now become mere packet forwarders, and the definition of a general control protocol that is used to set them with the necessary forwarding rules to accomplish with the objective of the network. Moreover, this scheme also proposes a conceptually centralized *brain* that knows the topology and state of the network to take decisions about packet forwarding, represents them into forwarding rules, and communicate them to the forwarding entities.

Contemplating the characteristics of SDN from the IoT perspective has led us to consider how SDN can be used to keep heterogeneity in networks and objects while building a bigger cooperation scheme by just integrating into the network a new higher layer control solution that interacts with the SDN controllers.

2. Proposed Approach

The first step in the design of the integration of SDN and IoT is to gather and analyze the different types of workloads that IoT elements will push to the network. This is the key aspect of the design and will determine the structure and modularity level of the *IoT Controller*. This is a high level controller that is connected to the SDN controller to interact with it and thus model the underlying network behavior and response to IoT operations.

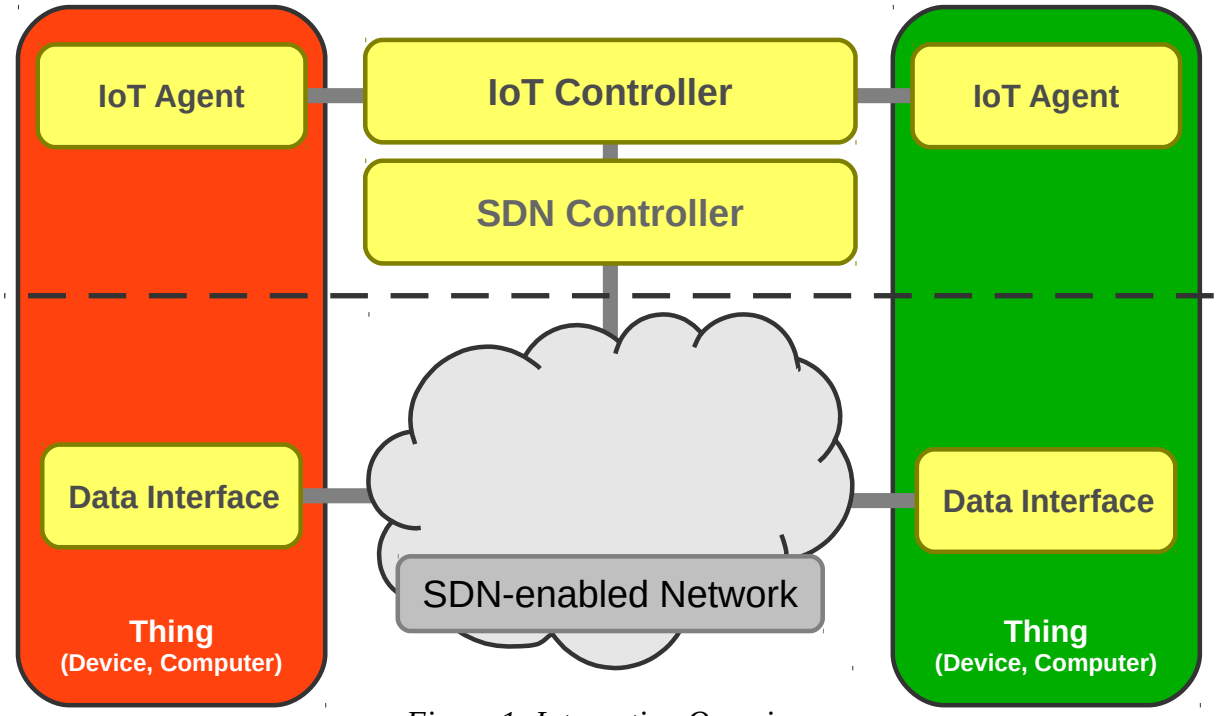

Figure 1: Integration Overview

The general view of the integration of SDN and IoT , as shown in Figure 1, includes a minimum set of functional blocks, differentiated by the actor and plane to which they pertain, that is object or network, data or control plane. Thus, two objects connected to an SDN-enabled network will be able to interact with the IoT controller by using their internal IoT agents. The objective is to provide context information to the controller for it to take the necessary decisions and reflect them into the underlying network. Although the IoT controller is depicted as just one functional block, it is internally modular so new functionality can be added to the IoT overlay without affecting other elements, neither needing to establish new relations with the SDN controller.

In current network architectures and protocols, a normal communication begins when a network object, the requester, asks the network to send a data packet or message of some type to another network object, the responder. Before this can take place, the requester has to know the identifier or address of the responder, and it has to be specified into the transaction. This model applicable to most protocols, including IoT protocols.

For such communication, the network establishes a path from the requester to the responder. Such path may be logical (establishing a persistent connection or circuit) or virtual (merely following routing tables). Here is where SDN enters into the game to allow objects relying on different (and thus heterogeneous) protocols to *talk* to each other. Thus, SDN mechanisms can be used to establish a path that connects both endpoints. Here it is called a forwarding path and it is achieved by setting

up the necessary forwarding rules into all forwarding elements found in such path.

At this point is where the IoT controller finds its role. It has to receive the communication interest from the requester, find the responder in the network graph, calculate the path using some routing algorithm, build the forwarding rules depending on the nature of the protocols used by the objects, and finally communicating such rules to the SDN controller for it to set them into the forwarders.

Communication interests are sent to the IoT controller by IoT agents installed into the objects. They will be integrated with other components of the object in order to find out the necessary details of the communication that is being established, such as the identifier or address of the destination object. This information is sent to the IoT controller before the communication is actually initiated, so the network will be prepared for it to proceed.

Once the IoT controller receives such interest with the identifier or address of the responder it has to find it in the network graph. This is easy because IoT agents will be registered into the IoT controller and they provide their corresponding identifier or address. Then, this controller calculates a path that connects both objects by running a routing algorithm with topology information from both IoT and SDN levels. This way, it is sure that the resulting path fits into the network and follows the possible rules established by network administrators at IoT or SDN levels.

Knowing the specific protocols used by both communication endpoints and having calculated the path that information will follow, the IoT controller now builds the necessary forwarding rules for each element of the path. Depending on the protocol used by the objects, the rules will have different matching fields from packets and will have different actions. If the protocols are different and not compatible, some rules will make the forwarders to adapt or translate it so the packets reach their destination in a manner that the destination understands.

Finally, the resulting rules are communicated to the SDN controller which uses the corresponding control protocol to set them into the forwarders indicated by the IoT controller. At this moment the path is built and both objects may begin their conversation. The whole process is very quick but may introduce some delay. However, it is only introduced to the first packet because once the rules are set into the forwarders, they run almost as efficient as any switch or router, specially if SDN switches are built on specific and optimized hardware.

As mentioned above, in order to form part of the IoT *overlay*, each object has to be registered into the IoT controller. This task is performed by the IoT agent associated to the object. The registration process will provide the IoT controller with the necessary information about the object, such as its network protocol, the underlying network to which it is tied, and its identifier or address.

3. Research Challenges and Objectives

Building the architecture described in the previous section exposes a set of research issues that have to be addressed by the design of the integration mechanism and the architecture in general. They are summarized as follows:

- *Common and heterogeneous identification scheme*: Objects may use different identification schemes, mainly related to the underlying network protocol they are using. In order to allow they to interoperate we need to survey the different identification approaches used by the protocols that will be supported by the IoT network. This will determine the form of the common mapping reference that will be used by the IoT controller to project every naming space into the others so identifiers used by objects are compatible with their protocols.
- *Where to instantiate IoT agents*: Although IoT agents have little work to do, so they are lightweight, not all objects may be able to instantiate an agent by themselves. Nevertheless, there are other places where they can be instantiated on behalf of those objects, such as the network gateway to which they are connected, the SDN forwarder (switch), or even together with the IoT controller as a different module. These alternatives have to be evaluated from

different perspectives so that the best one is included into the design or even concluding that more than one has to be considered.

- *Routing algorithm*: There are many good routing algorithms that can be used to find the path between two objects but they work with just one topology. The architecture described here requires a routing algorithm that considers the state of two separate but overlapped topologies, the underlying SDN topology and the overlay IoT topology. The algorithm must also consider different aspects such as policies or bandwidths. A proper algorithm to deal with this requirement has to be found and evaluated in order to determine how it fits with the objectives of this approach.
- *Forwarding rule formulation*: Once the path has been calculated, the IoT controller has to reflect it into different rules that will be sent to different forwarders (switches). Although it is easy to do, the formulation is not direct and has to consider the different adaptations, mappings, underlying protocols, matching fields, and general actions necessary to properly forward a packet towards its destination.
- *Northbound API stabilization*: Current SDN models include the so called northbound API that can be used by external modules, such as the IoT controller, to communicate control operations to the SDN controller. However, this API is not well defined nor stabilized, so a requirement of the current approach is to find (or wait for) the stabilization of such interface.
- *IoT controller modularity*: In the previous section we have described the main functionality of the IoT controller but it should be open to future enhancements in the form of new modules. This requires a thorough study of the most extensive and durable module systems and how they have evolved on time so the selected alternative ensures the evolution ability of the proposed approach.
- *Deployment procedure*: An initial deployment procedure should be designed and analyzed from the point of view of the SDN and currently available infrastructures. As this is the key aspect for the success of the proposal, it is important to validate it against current and future networks during the initial phases of the research.

4. Conclusions

As discussed throughout this document, the flexibility offered by SDN can be effectively used to allow objects connected to heterogeneous networks to communicate each other. This is independent of the capabilities of such objects, so it fits perfectly into IoT scenarios. Limited computation or communication ability is an important factor that determines the *shape* of IoT networks and the protocols they use. It has meant the design of specific protocols for specific purposes which are generally incompatible each other and does not permit the objects to easily interact. This problem can be resolved by using the mechanisms offered by SDN by just building a network service on its top that gives support to IoT objects. Nevertheless, this exposes other challenges that must be resolved before continuing with the design of the discussed approach. However, the new challenges and research objectives are much simpler than the premises of the general problem, so resolving them would be a good start towards the design of the integrated architecture.