IoT-aware Multi-layer Transport SDN and Cloud Architecture for Traffic Congestion Avoidance Through Dynamic Distribution of IoT Analytics

Raul Muñoz[1], Ricard Vilalta[1], Noboru Yoshikane[2], Ramon Casellas[1], Ricardo Martínez[1], Takehiro Tsuritani[2], Itsuro Morita[2]

[1] Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Castelldefels, Spain.
[2] KDDI Research, Inc., Saitama, Japan.

Abstract *We present and experimentally assess the first IoT-aware SDN and cloud architecture that deploys IoT flow monitoring and traffic-congestion avoidance techniques in order to dynamically and efficiently distribute the processing of IoT analytics from core datacentres to the network edge.*

Introduction

It is envisioned that the Internet of Things (IoT) will connect billions of heterogeneous devices (ranging from complex interactive systems to tiny sensors) to the transport network using widely deployed wireless, mobile and fixed access technologies (e.g, Wi-Fi, LoRa, eMTC, NB-IoT, PLC, Ethernet). Much like 5G, IoT also requires cloud computing and storage datacentres (DC) in order to perform IoT analytics from the data collected of sensors and actuators (e.g., Siemens wind power develops IoT analytics to predict and diagnose potential problems in 9,000 wind turbines with 400 sensors each, sending data several times a second[1]).

Collecting information from multiple sensors and analyzing it into a relatively small number of core DCs does not scale, particularly as the number of IoT devices and volume of data is forecasted to explode. On the transport network side, the generation of large number of flows (e.g. telematics, sensors) or huge aggregated volumes of data (e.g. remote monitoring, digital signage) from the edge of the network to the core DCs could congest the network. It will also be costly because transporting bits from the edge to core network actually costs money. Additionally, some mission-critical IoT applications with very stringent delay requirements may also require to perform IoT analytics in the edge in order to perform real-time actions.

An solution recently proposed to address the new IoT requirements such as dense high traffic processing, large number of connections processing, peak traffic processing and low-latency processing is to distribute the processing of the IoT analytics from the core DC to the edge of the network (edge computing), known as edge IoT analytics[2]. First analytics can be carried out on the edge cloud and only the necessary data or results are sent for further analysis (e.g. Big data) /storage in the core DC. Thus, the distribution of IoT analytics offloads the network and the DCs by creating a model that scales and releases bottlenecks. However for an efficient distribution of IoT analytics and use of network resources it requires a tight coordination between the IoT analytics platform, the Transport SDN network and the cloud infrastructure. This paper presents and experimentally validates the first IoT-aware multi-layer (packet/optical) transport SDN and cloud architecture that deploys an IoT-traffic control and congestion avoidance mechanism for dynamic distribution of IoT processing to the edge of the network (i.e., edge computing) based on the actual network resource state.

IoT-aware SDN and cloud architecture

Fig. 1 depicts the proposed IoT-aware multi-layer transport SDN and cloud architecture. At the infrastructure layer, it is composed of several packet and optical transport domains (access, metro and core) providing connectivity to core-DCs and micro & small-DCs (located at the network edge) providing computing and storage resources. The proposed solution deploys IoT Flow Monitors (IoT-FM) at the edge of the packet transport network domains. The IoT-FMs are responsible of monitoring the average bandwith of the aggregated IoT traffic.

At the control layer, each micro, small and core DC can have a dedicated DC controller (e.g. OpenStack). On top of the multiple DC controllers we deploy a cloud orchestrator (CO) that enables to deploy general cloud services (e.g. IoT analytics) across distributed DC infrastructures (micro, small, core) resources for multiple tenants[3]. On the transport side, each

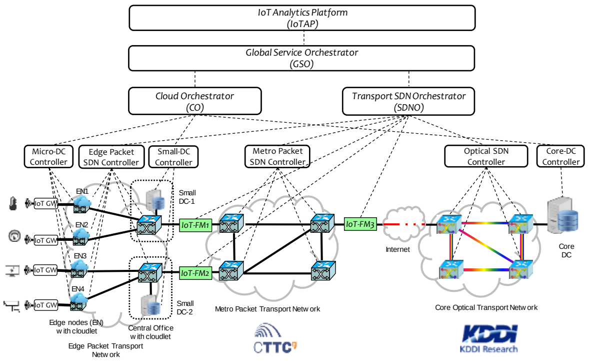


Fig.1: Proposed architecture and experimental scenario

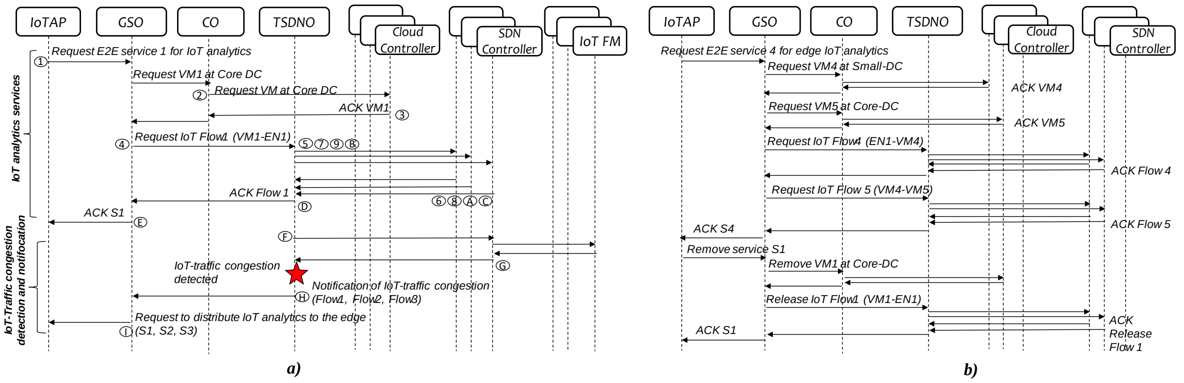

IoTAP
GSO
CO
TSDNO
Cloud Controller
SDN Controller
IoT FM
IoT analytics services
IoT-Traffic congestion detection and notifocation
① Request E2E service 1 for IoT analytics
Request VM1 at Core DC
② Request VM at Core DC
ACK VM1 ③
④ Request IoT Flow1 (VM1-EN1)
⑤⑦⑨Ⓑ
ACK Flow 1
⑥⑧ⒶⒸ
Ⓓ
ACK S1
Ⓔ
Ⓕ
Ⓖ
IoT-traffic congestion detected
Notification of IoT-traffic congestion (Flow1, Flow2, Flow3)
Ⓗ
Request to distribute IoT analytics to the edge
Ⓘ (S1, S2, S3)
a)
IoTAP
GSO
CO
TSDNO
Cloud Controller
SDN Controller
Request E2E service 4 for edge IoT analytics
Request VM4 at Small-DC
ACK VM4
Request VM5 at Core-DC
ACK VM5
Request IoT Flow4 (EN1-VM4)
ACK Flow 4
Request IoT Flow 5 (VM4-VM5)
ACK Flow 5
ACK S4
Remove service S1
Remove VM1 at Core-DC
Release IoT Flow1 (VM1-EN1)
ACK Release Flow 1
ACK S1
b)

Fig. 2: Orchestration workflows. a) Provisioning of IoT analytics and congestion detection; b) distribution of IoT analytics services

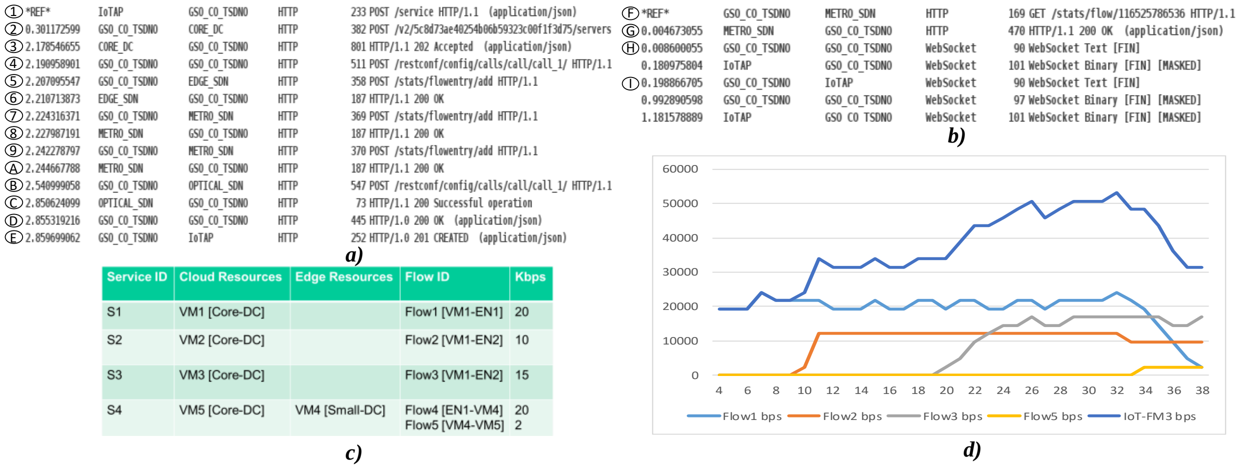
packet or optical transport domain can have a dedicated SDN controller (e.g. OpenDaylight, ONOS, Ryu). On top of the SDN controllers, we deploy an IoT-aware Transport SDN orchestrator (TSDNO) to provide end-to-end connectivity services using the common Transport API (T-API)[4]. In this paper, we have extended the TSDNO to provision packet flows tagged as IoT, and to monitor the IoT-FMs in order to detect and prevent IoT-traffic congestion. More specifically, the TSDNO can define for each monitored link the maximum IoT-traffic bandwidth threshold and the time over threshold (ToT) allowed to avoid the generation of alarms for peak traffics above the bandwidth threshold. Then, the TSDNO requests statistics about the IoT traffic in the IoT-FMs to the SDN controllers on a periodic basis. When the TSDNO detects that a defined bandwidth threshold has been exceeded for a duration longer than the allowed ToT, the TSDNO identifies all IoT flows going through the congested link and notifies the IoT-aware global service orchestrator (GSO) to trigger the distribution of IoT analytics to the edge.

The IoT-aware GSO is deployed on top of TSDNO and the CO. It is responsible to provide global orchestration of end-to-end services by decomposing the global service into cloud services, and network services, and forwarding these service requests to the CO and TSDNO. For example, the GSO can provide E2E services for IoT analytics by requesting to the CO the provisioning of a VM in the core-DC for the deployment of IoT analytics, and to the TSDNO the required connections between the VM and the service end-points. The GSO is also responsible to serve application requests through a northbound interface (NBI). In this paper, we consider the IoT Analytics Platform (IoTAP) on top of the GSO using the NBI to request the provisioning of end-to-end services for IoT analytics services. The NBI is also used to request the IoTAP to distribute IoT analytics services the edge when an IoT-traffic congested link is detected.

Provisioning of IoT analytics, congestion detection, and distribution of IoT analytics

When the IoTAP needs to deploy a new IoT analytics service as depicted in Fig. 2.a, it requests to the GSO the provisioning of and E2E service (i.e. S1) composed of a cloud computing resource and an IoT flow with the required bandwidth between the VM and the edge node where the IoT flow is originated (service end-point). First, the GSO requests to the CO the provisioning of a VM at the Core-DC. The CO forward this request to the specific core-DC controller responsible of the actual provisioning (i.e. VM1). Second, the GSO requests the provisioning of an IoT flow (i.e. IoT Flow1) to the TSDNO between the provisioned VM and the edge node specified by the IoTAP (i.e. EN1). The TSDNO is responsible to compute an end-to-end multi-layer under QoS constraints (e.g., bandwidth), to split the computed path into domain segments, and to request the actual provisioning of the path segments to the involved domain SDN controllers. The flows are identified with the IoT tag inserted as an OpenFlow cookie. This process is repeated any time the IoTAP requires to setup an IoT analytics service.

In parallel, the TSDNO monitors the IoT traffic from the IoT-FMs. Once the TSDNO detects an IoT-traffic congested link, it generates an alarm to the GSO, identifying all IoT flows going through the congested link (i.e., flow1, flow2, flow3). After the notification, the GSO identifies all E2E services involved (i.e., S1, S2, S3) and requests to the IoTAP the distribution of some of the affected IoT analytics services to the edge. The IoTAP is responsible for selecting the IoT analytics services that will be distributed to the edge, based on the characteristics of the IoT applications. Once selected the services to be distribute (i.e. S1 in Fig.2.b), first the IoTAP requests to the GSO the provisioning of a new E2E service (i.e., S4 in Fig.2.b) composed of a cloud computing resource (i.e., VM4 at core-DC) connected with an edge computing resource (i.e., VM5 at small-DC) and connected with the service end-point (i.e., EN1). Then, the IoTAP proceeds to release the established IoT Analytics service, by specifying to the GSO the identifier of the E2E service (i.e., S1).



```
①*REF*        IoTAP        GSO_CO_TSDNO  HTTP     233 POST /service HTTP/1.1  (application/json)
②0.301172599  GSO_CO_TSDNO CORE_DC       HTTP     382 POST /v2/5c8d73ae40254b06b59323c00f1f3d75/servers
③2.178546655  CORE_DC      GSO_CO_TSDNO  HTTP     801 HTTP/1.1 202 Accepted  (application/json)
④2.190958901  GSO_CO_TSDNO GSO_CO_TSDNO  HTTP     511 POST /restconf/config/calls/call/call_1/ HTTP/1.1
⑤2.207095547  GSO_CO_TSDNO EDGE_SDN      HTTP     358 POST /stats/flowentry/add HTTP/1.1
⑥2.210713873  EDGE_SDN     GSO_CO_TSDNO  HTTP     187 HTTP/1.1 200 OK
⑦2.224316371  GSO_CO_TSDNO METRO_SDN     HTTP     369 POST /stats/flowentry/add HTTP/1.1
⑧2.227987191  METRO_SDN    GSO_CO_TSDNO  HTTP     187 HTTP/1.1 200 OK
⑨2.242278797  GSO_CO_TSDNO METRO_SDN     HTTP     370 POST /stats/flowentry/add HTTP/1.1
Ⓐ2.244667788  METRO_SDN    GSO_CO_TSDNO  HTTP     187 HTTP/1.1 200 OK
Ⓑ2.540999058  GSO_CO_TSDNO OPTICAL_SDN   HTTP     547 POST /restconf/config/calls/call/call_1/ HTTP/1.1
Ⓒ2.850624099  OPTICAL_SDN  GSO_CO_TSDNO  HTTP      73 HTTP/1.1 200 Successful operation
Ⓓ2.855319216  GSO_CO_TSDNO GSO_CO_TSDNO  HTTP     445 HTTP/1.0 200 OK  (application/json)
Ⓔ2.859699062  GSO_CO_TSDNO IoTAP         HTTP     252 HTTP/1.0 201 CREATED  (application/json)
```

a)

```
Ⓕ*REF*        GSO_CO_TSDNO METRO_SDN     HTTP       169 GET /stats/flow/116525786536 HTTP/1.1
Ⓖ0.004673055  METRO_SDN    GSO_CO_TSDNO  HTTP       470 HTTP/1.1 200 OK  (application/json)
Ⓗ0.008600055  GSO_CO_TSDNO GSO_CO_TSDNO  WebSocket   90 WebSocket Text [FIN]
 0.180975804  IoTAP        GSO_CO_TSDNO  WebSocket  101 WebSocket Binary [FIN] [MASKED]
Ⓘ0.198866705  GSO_CO_TSDNO IoTAP         WebSocket   90 WebSocket Text [FIN]
 0.992890598  GSO_CO_TSDNO GSO_CO_TSDNO  WebSocket   97 WebSocket Binary [FIN] [MASKED]
 1.181578889  IoTAP        GSO CO TSDNO  WebSocket  101 WebSocket Binary [FIN] [MASKED]
```

b)

Service ID	Cloud Resources	Edge Resources	Flow ID	Kbps
S1	VM1 [Core-DC]		Flow1 [VM1-EN1]	20
S2	VM2 [Core-DC]		Flow2 [VM1-EN2]	10
S3	VM3 [Core-DC]		Flow3 [VM1-EN2]	15
S4	VM5 [Core-DC]	VM4 [Small-DC]	Flow4 [EN1-VM4] Flow5 [VM4-VM5]	20 2

c)

d)

Fig. 3: a) Provisioning of IoT analytics; b) congestion detection; c) requested services; IoT traffic monitored by IoT-FM3

Experimentation

A proof-of-concept (PoC) of the proposed architecture has been evaluated and validated in a joint experimentation between the CTTC ADRENALINE Testbed[5] in Barcelona (Spain) and the KDDI Research Testbed in Saitama (Japan) as depicted in Fig.1. The ADRENALINE testbed provides an edge (access) and metro packet transport networks for traffic aggregation and switching of flows, and distributed edge computing platform. The KDDI Research testbed provides a core-DC and an optical core network. Both infrastructures are connected using OpenVPN tunnels on top of internet.

The packet transport network leverages OpenFlow switches using Open vSwitch (OVS) technology. The edge and metro segments are controlled with two Ryu SDN controllers. The distributed edge computing platform is composed of two small-DCs, and four micro-DCs, leveraging virtual machines (VM) and container-based technologies. It is controlled using three OpenStack controllers, one for the core-DC, another for the two small-DCs, and the third one for the four micro-DCs.

For the experimentation in the PoC, the maximum IoT-traffic bandwidth threshold is configured to 40 Kbps and the TOT to 10s. The IoT-FMs have been integrated in the OpenFlow switches for simplicity. Individual flow statistics information is requested with the OFPST_FLOW stats request type. The TSDNO requests the byte_count of the IoT flows on the output port of the OpenFlow switch where the IoT-FM should be placed with a periodicity of 1s. Then, the TSDNO computes, each second, the average IoT bandwidth (bw) for each IoT flow in the last 5 seconds, that is, the bandwidth at time n (t_n) is (byte_count (t_n) - byte_count (t_{n-5})) / 5. The aggregated IoT traffic is the sum of the average bw of all IoT flows. First, we provision three E2E services (i.e., S1, S2 and S3) for deploying IoT analytics in the cloud according to the parameters shown in Fig. 3.c. Fig 3.a shows the network traffic capture at the GSO for the provisioning of an E2E service, showing all the communications workflows between the different involved systems. After the provisioning of each service, we use a traffic generator to generate packets with a constant bitrate, according to Fig. 3.c (20Kbps, 10 Kbps and 15 Kbps). Fig. 3.d shows the IoT traffic monitored by IoT-FM3 (Fig.1), showing the average IoT bandwidth employed by flow1, flow2 and flow3, as well as the overall aggregated traffic (IoT-FM). We can see that at time 21s the maximum IoT bandwidth (40Kps) is exceeded and therefore at time 31s the TSDON notifies about the IoT-traffic congested link, as shown in the network traffic capture of Fig.3.b. Then, the IoTAP decides to provision a new service S4 (Fig. 3.b) and remove service S1. From Fig. 3.d, we can appreciate that at time 34s, the flow1 (20Kbps) from S1 starts to decrease, and the new flow5 (2Kbps) from S4 appears. Flow4 is not shown because VM4 is located in the small-DC, and therefore does not cross IoT-FM3.

Conclusions

This paper has shown the first integration of IoT, SDN and cloud services aiming at efficiently deploying edge IoT analytics in order to offload the transport networks.

Acknowledgements

This research was partially funded by MINECO project DESTELLO (TEC2015-69256-R).

References

[1] *https*://en.wikipedia.org/wiki/Siemens_Wind_Power

[2] R. Vilalta, et al., End-to-End SDN/NFV Orchestration of Video Analytics Using Edge and Cloud.. OFC 2017.

[3] A. Mayoral, et al., Multi-tenant 5G Network Slicing Architecture with Dynamic Deployment..", ECOC 2016.

[4] R. Muñoz et al., "The need for a The Need for a Transport API in 5G networks", Proc. OFC, Th3K.4 Anaheim (2016)

[5] R. Muñoz et al., "The ADRENALINE Testbed: An SDN/NFV Packet/Optical Transport..." EUCNC2017