SIMECA: SDN-based IoT Mobile Edge Cloud Architecture

Binh Nguyen, Nakjung Choi, Marina Thottan, and Jacobus Van der Merwe
{binh,kobus}@cs.utah.edu, {nakjung.choi,marina.thottan}@nokia-bell-lab.com

***Abstract*—In future mobile networks, e.g., 5G, emerging IoT services are expected to support billions of IoT devices with unique characteristics and traffic patterns. In this paper we propose an SDN-based IoT Mobile Edge Cloud Architecture (SIMECA[1]) which can deploy diverse IoT services at the mobile edge by leveraging distributed, lightweight control and data planes optimized for IoT communications. We prototyped our architecture using a pre-commercial mobile networking software stack to demonstrate the feasibility and utility of our approach.**

I. Introduction

Future mobile networks (i.e., 5G) will need to support a large number of Internet-of-Things (IoT) devices; around 26 billion by 2020 according to some estimates [2]. Emerging IoT devices are expected to also use cellular radio access technologies to take advantage of mobile network features, such as wider coverage, better support for mobility, well-managed and secured [3]. The massive number of devices implies a fundamental question of how well the current cellular architecture would support IoT and what might need to change to support IoT applications, without adversely affecting the network.

While some expect the 5G architecture to be radically different from 4G [4], it will nonetheless be informed and derived from 4G principles and approaches [5], [6], [7], [8], [9]. In this work we follow a similar approach by considering the limitations of 4G networks, in particular the EPC core network, in supporting IoT and then combining evolved and re-factored 4G mechanisms with software-defined networking (SDN) and network function virtualization (NFV) to realize an IoT friendly mobile (core) network architecture.

The 4G network architecture is heavy-weight and optimized for human-to-human and human-to-machine communication. This is, however, a poor fit for IoT devices, where we might expect a much larger number of devices with different traffic patterns. For example, millions of simple sensors that might sporadically send only a few bytes of data every half hour. Such applications might prefer low overhead over high quality data delivery. Further, the control signaling in 4G networks has grown significantly and already present significant pressures on network components even just to support conventional devices and services. Control traffic grows 50% faster than data traffic while generating no revenue [10]. MME (a main control entity in LTE/EPC network) already experienced up to 1500 messages per User Equipment (UE) per hour under adverse conditions [11].

In terms of service abstractions, current 4G networks offer a "Device to Server" service, that is suitable for human-centric devices, where the network mostly needs to deliver an "Internet" service (e.g., web browsing). This is ill-suited for machine-to-machine or machine-to-human type communications. All peer-to-peer traffic in the current LTE/EPC networks must go through a centralized gateway (Packet Data Gateway - PGW) that adds extra latency to the end-to-end path event when end points are geographically close.

In this work we present an SDN-based IoT Mobile Edge Cloud Architecture (SIMECA). SIMECA leverages SDN for forwarding and NFV for network function deployment to realize a new service API that replaces LTE/EPC and is geared towards IoT devices and services. The design of SIMECA is based on the following insights: (1) Services and mobility functions are hosted inside edge clouds to improve network latency; (2) We argue that *best-effort* forwarding is more suitable for IoT devices instead of high QoS delivery in LTE/EPC. Best-effort forwarding eliminates core network components (i.e., SGW/PGW) and therefore reduces control plane overhead involving end-to-end connectivity; (3) Instead of extra tunneling overhead in LTE/EPC, SIMECA proposes a *packet header translation* mechanism realized by SDN forwarding that reduces packet header overhead and favors small payload traffic; (4) Separating end-point identity and routing identity together with a proper address tracking mechanism enable *seamless mobility* and *peer-to-peer* communications in SIMECA. We make the following contributions:

- We propose a novel mobile edge cloud architecture, SIMECA enables direct P2P communication for IoT end devices while reducing EPC core network latency.
- SIMECA proposes light-weight control and data planes which scale better and have lower overhead to better support IoT services, compared to the current LTE/EPC control plane. SIMECA's data plane overhead is 20% smaller than LTE/EPC for small sensor payload sizes (e.g., $<$100 Bytes) while the control plane overhead is 37% smaller.
- We validate SIMECA by prototyping and evaluating our approach by refactoring pre-commercial EPC software and using an SDR-based eNodeB.

II. Motivation and Overview

With reference to the current LTE/EPC architecture [12] SIMECA targets three limitations in the current *EPC core*.

[1]Instructions to access SIMECA profile is at [1]. This work was initiated when the student was an intern at Bell labs and was supported in part by the National Science Foundation under grant number 1305384.

Inappropriate service abstraction and inflexibility: While built according to well-defined standards, current LTE/EPC data network was designed to provide *Internet service* and its equipment is proprietary and closed. This specialized hardware (Packet data gateway - PGW, Service gateway - SGW) is deployed in a limited number of physical locations. Moreover, traffic between endpoints need to travel to these centralized locations, adding extra latency to the end-to-end path. This results in several draw-backs for IoT devices: (1) These devices *must* use the LTE/EPC abstraction even when the abstraction is not optimized for its characteristic, i.e., LTE/EPC incurs high latency and overhead for IoT devices [13], (2) devices cannot establish a (native) peer-to-peer (P2P) communication, as the architecture does not offer a direct way to do it, and have to rely on additional techniques such as NAT traversal [14], [15], (3) P2P traffic incurs high end-to-end latency because of hierarchical routing [16].

We argue that there should be a more suitable service abstraction for IoT devices that should: replace the *Internet* abstraction in LTE/EPC by a new abstraction that is *more suitable* for IoT communication models, e.g., supports peer-to-peer communications natively. The abstraction should also be realized by control/data planes that favor IoT devices, e.g., less signaling and data overhead. Moreover, the network should apply NFV/SDN to allow flexible new service deployment and resource management, e.g., 3rd party service providers could deploy their own services on top of a shared infrastructure provided by an operator. The infrastructure that run the network should be more distributed, i.e., a *mobile edge network and edge cloud* architecture where compute resources are placed closer to the mobile network edge.

Heavy weight data and control plane: The current LTE/EPC network was designed to support devices streaming large amount of high quality data/voice (e.g., a smart phone streaming a video or making a voice call) rather than *massive* IoT/M2M devices with sporadic, best effort traffic (e.g., millions of meters sending a temperature sample every 20 minutes.) For example, data packets are delivered by EPS bearers in the EPC core network which are GTP tunnels. Although those tunnels support QoS (via different QCI indexes [17]) which might be needed for human devices (smart phones), setting up or maintaining them incurs significant overhead [18]. A standard LTE initial attach procedure incurs up to 28 control messages, while a service request procedure incurs up to 14 control messages [19]. An EPS bearer consists of 8 tunnel end-points (2 at eNodeB, 4 at SGW, and 2 at PGW.)

In the data plane, the GTP tunnels adds data plane overhead to IoT traffic. This overhead becomes relatively more significant if the packet's payload is small. For example, a body temperature sensor which generates 1.5 Bytes of data per sample will incur 36 Bytes of additional GTP/UDP tunneling overhead. Moreover, as the RAN capacity is expected to dramatically increase with emerging access technologies in 5G [20], increases in the amount of traffic would exacerbate the problem.

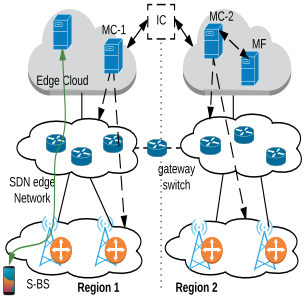


Fig. 1: Mobile Edge Network & Cloud Infrastructure

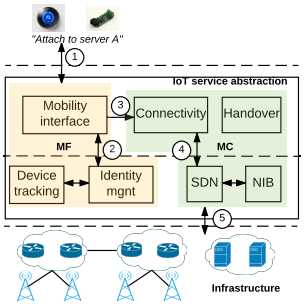


Fig. 2: IoT service abstraction (ISA)

III. SIMECA Architecture

We present our work on SIMECA, an SDN-based IoT Mobile Edge Cloud Architecture. The main design points of SIMECA are: i) A mobile infrastructure consists of multiple Mobile edge clouds deployed close to users. The edge cloud is SDN controlled. (ii) Running on top of the infrastructure is SIMECA service abstraction, i.e., an *IoT service abstraction (ISA)*, which is low overhead and scalable. ISA also offers P2P connectivity natively.

A. SDN-based mobile edge network and cloud infrastructure.

Figure 1 depicts the infrastructure that SIMECA runs on. The infrastructure is distributed and consists of smaller regions and is shared. Service providers could request resources (VMs) via an Infrastructure Controller (IC in Figure 1.) We note that our assumed infrastructure is aligned with broad industry vision towards an infrastructure in support of mobile edge computing [21], [22]. Each region serves a metropolitan area or similar. It consists of:

- *SDN-enabled Base Stations - S-BS*: In SIMECA base stations are SDN-enabled, allowing them to apply the SDN match/action paradigm to modify flows per request via a SDN controller. The SDN-enabled Base stations are responsible for packet classification (i.e., modifying packet's source and destination IP) for packets forwarding inside the SDN edge network (described next).
- *SDN edge network*: The SDN edge network consists of multiple SDN switches that connect the S-BS and a local edge cloud. This SDN edge network has several gateway switches that connect to the SDN of a neighboring region. Intra-region traffic is forwarded by this SDN edge network based on header destination address while inter-region traffic goes across regions via gateway switches.
- *Edge cloud*: The edge cloud consists of multiple computation nodes that are capable of running *both* virtualized mobility functions (NFVs) and IoT services. The edge cloud is connected to the SDN edge network via an SDN switch to enable end-to-end SDN control.

B. IoT Service Abstraction (ISA)

ISA exposed interface. Unlike conventional smart phones, there are two communication models for IoT applications:

client-server and publisher-subscriber. Client-server model is suitable for one-to-one communications, e.g., client controls an actuator. Publisher-subscriber model is suitable for many-to-many communications, e.g., multiple publishers publish data to a broker and multiple subscribers get data from the broker. These two models suggest both device-to-edge cloud communication (i.e., we call this C2S communications) and device-to-device communication (i.e., we call this P2P communications). SIMECA's ISA therefore exposes to the IoT devices computational resources and an interface to request *end-to-end connectivity* between end IoT devices (i.e., P2P connectivity) or between devices and services (i.e., C2S connectivity) (More detail provided in § III-C).

The ISA is realized by two control plane entities: Mobility Network Functions (MF) and Mobility SDN Controller (MC).

Mobility Functions (MF). The MF mainly deals with mobility specific functionalities such as: tracking device's location, performing mobility procedures with devices and base stations. As shown in Figure 2, the MF consists of 3 components (depicted in yellow boxes): *Mobility Interface, Device Tracking, and Identity Management*. The *Mobility Interface* component speaks standard protocols with the devices and base stations (i.e., NAS and S1AP protocols in 3GPP [23]). Using the interfaces provided by this entity, an IoT device can request end-to-end connectivity in the network (i.e., Service Request procedure). This enables SIMECA to operate with base stations with normal 3GPP control plane and is compatible with existing LTE devices.

The *Device Tracking* component keeps track of devices attachment point when devices move (i.e., which base station a device currently attaches to). By tracking a device, SIMECA is aware of where to forward traffic to when the device moves. The *Identity Management* component manages identities for devices. Specifically, a device in SIMECA has two identities: one for end-point application use and one for network routing. This identity separation allows seamless handover in SIMECA (details about identity, device tracking will be described in more detailed in § III-C.)

For example, a device turns on, authenticates with the network using existed protocols, and sends an "Attach to Server A" request to the *Mobility Interface* (step 1 in Fig. 2). The *Mobility Interface* then requests identities with the *Identity Management* and replies to the device. The *Identity Management* updates the *Device Tracking* component the current location of the device (step 2).

Mobility SDN Controller (MC). While the MF deals with mobility functionalities, the MC mainly deals with end-to-end path implementation using SDN. I.e., after dealing with mobility requests and assigning identities, the MF notifies MC to implement paths accordingly. As shown in Figure 2, the MC consists of 4 components (depicted in green boxes): *Connectivity, Handover, SDN, and Network Information Base (NIB)*. The *Connectivity* component is specific to implement end-to-end paths (e.g., set up a path from a device to a server when the device attaches) while the *Handover* component is responsible for modifying paths during mobility events (e.g., forward traffic to a new base station if the device moves). The two components use the SDN interface (*SDN* component) in combination with network topology information (i.e., *NIB* component) to implement the paths. The NIB includes information such as on which base station SDN rules should be installed to realize an end-to-end connectivity.

For example, after receiving the "Service Request" from the device, the *Mobility Interface* tells the *Connectivity* component in the MC to install a path from the base station (e.g., cell-ID) to server A (step 3 in Fig. 2). The *Connectivity* component uses the *NIB* to translate the base station ID to the SDN data path object and use *SDN* to realize the end-to-end connectivity (step 4,5).

C. Lightweight IoT Data Plane

In contrast to heavy-weight tunneling with QoS guarantees in the standard EPC core, SIMECA's best-effort forwarding data plane could benefit IoT services. It: (1) reduces packet header overhead for IoT devices, (2) reduces the number of forwarding states inside the network, while (3) enables seamless mobility.

Device identity and forwarding identity. A device in SIMECA is identified using two identities: *device identity* (DI) and *device routing identity* (RI).

• *Device identity-DI:* Each device has a unique DI assigned by the network when the device attaches to the network. This DI is associated with the device and *does not change* until the device detaches from the network. As this DI does not change upon device mobility, applications running on the device could use this DI to enable continuous operation, i.e., seamless mobility.
• *Routing identity-RI:* In the SDN edge network, packets are forwarded using a separate RI (different from the DI). This RI is also allocated by the network when a device attaches to the network. Unlike DI, RI is specific to a base station and *changes* when a device moves to another base station. Under a specific base station, there is a unique mapping between DI and RI.

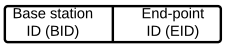


Fig. 3: Routing identity header structure

Lightweight header translation. Unlike in EPC where extra tunnel overhead is added to packets at eNodeB and S/PGW for delivery over a transport network, SIMECA does not add *any* extra overhead to packets. SIMECA instead uses header translation mechanism to translate DI-RI at network edge (i.e., base stations). Inside the SDN edge network, packets have source and destination RIs in their header for forwarding. When the packets reach end-points, they have DIs as their source and destination for seamless mobility. When a device moves across base stations, it has a new RI and therefore the DI-RI mapping also gets updated accordingly. To realize this DI-RI mapping, an SDN controller installs SDN rules to base stations in a *proactive* manner during device attach/service request/handover.

Seamless mobility. SIMECA supports seamless mobility as it separates device identity, which is unchanged by mobility,

and forwarding identity which changes with mobility. When a device moves to a new base station, it is assigned a new RI that has the new base station's BID. Note that the DI of the device does not change therefore from the end-points (device, server) perspective, the RI change is *transparent*.

For example, device A with DI of *192.168.3.33* attaches to base station 1 and gets RI *192.168.10.10/24*. When it moves to base station 2 it gets a new RI *192.168.20.10/24*. Packets inside the SDN edge network are now forwarded to base station 2 where device A is currently attached using the new BID *192.168.20.0/24*. However, packets arriving at device A and server still have device A's DI *192.168.3.33* in the header.

P2P forwarding and device location tracking. Unlike EPC, SIMECA naturally enables peer-to-peer communications in which a device is reachable via its DI. This type of communication is enabled when a device requests a *P2P-attach* to the other device with the destination device's DI specified (e.g., "Device DI1 attaches to device DI2", Section III-B). When receiving a P2P-attach request, SIMECA control plane needs to know the RI of the destination device to install SDN rules in the source base station for DI-RI header translation. This DI-RI mapping information is dynamic as the peer device moves across base stations. SIMECA control plane stores this DI-RI mapping in a centralized table (location tracking table) and updates the table accordingly when a device changes its attachment point (hands over to another base station).

Table I shows an example of location tracking table in SIMECA. A new entry is created by SIMECA control plane when a device attaches *and* registers itself as a reachable service (Section III-B). It is updated when the registered device moves across base stations (more details in Section III-D).

TABLE I: Attachment point tracking table for P2P communications

Device identity	Routing identity
192.168.3.33	192.168.10.10/24
192.168.3.34	192.168.11.11/24

Example of packet forwarding. Figure 4a shows the translation of device identity to routing identity at base station for intra-region forwarding: P2P forwarding on the left and C2S forwarding on the right. For P2P forwarding, device 1 initiates a flow to device 2 in the *same* region. The packet header at device 1 is $[DI1, DI2]$ where $DI1$ and $DI2$ are device 1 and 2's *device identity*. Base station 1 has a SDN flow rule that *replaces* uplink packets header with device 1 and 2's *routing identity* and forwards the packets to the SDN edge network. The SDN edge network has pre-installed SDN rules that matches on base station ID in packet header (i.e., $[RI1, RI2]$). Upon arriving at destination base station 2, the packet header is translated back to $[DI1, DI2]$ and delivered to device 2.

Similarly for C2S forwarding, base station 3 translates packets source address from $DI3$ to $RI3$. The destination address which is the server address does not change (i.e., A in figure 4a). The SDN edge network forwards uplink packets based on server destination address. For downlink packets from the server, the SDN edge network simply forwards packets by matching the destination address in the packet which is embedded in the uplink packets (i.e., $RI3$).

As SIMECA uses an intelligent-edge dumb-core principle for packet forwarding, RI in the SDN edge network could have a predefined structure that aggregates end-points in the same base station. This reduces the number of forwarding states in the SDN edge. Moreover, for each end-to-end connection, only 2 SDN rules are installed at the base station in SIMECA compared to 8 GTP tunnel IDs for each EPS bearer in LTE/EPC.

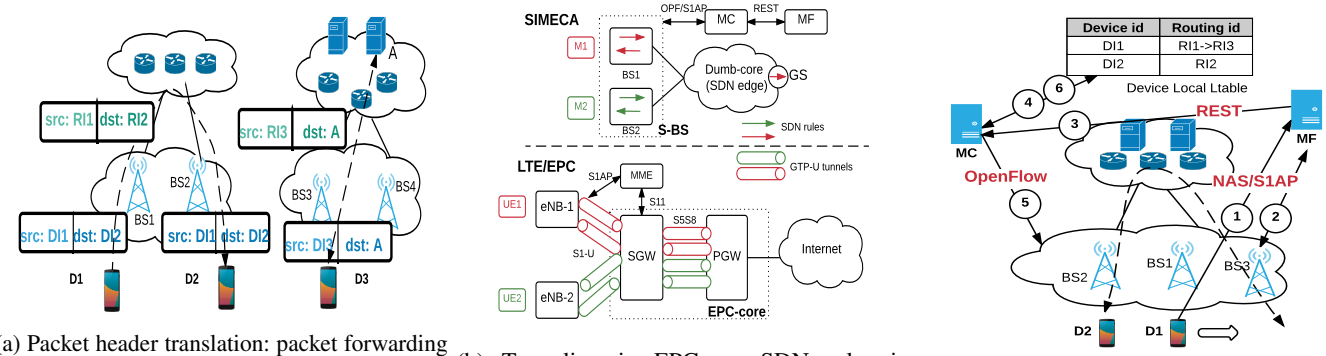
D. Lightweight IoT Control Plane

Eliminate tunneling to reduce control plane overhead. By eliminating EPC tunnels in the SDN edge network and allowing packet classification at the edge, SIMECA eliminates control signaling overhead to setup/maintain the tunnels and reduces the number of forwarding states in the data path. Compared to EPC, SIMECA's control plane is more lightweight for multiple reasons: (1) packet classification happens only at the network edge (base stations) therefore requires fewer interactions to setup a path (i.e., SDN controller only needs to push SDN rules into the base station as opposed to interactions between eNodeB, SGW, PGW and MME in EPC), (2) SIMECA's dumb-core eliminates the overhead to setup forwarding states in the SDN edge network as compared to GTP-U tunnels in EPC, this also results in a lower number of forwarding states in the data plane in SIMECA, (3) unlike in LTE/EPC networks the forwarding states in SIMECA do not expire due to radio bearer release therefore eliminates the control signaling overhead incurred re-establish the data path when devices become active again after an idle period (i.e., Service Request and Paging), and (4) OpenFlow control messages are more lightweight than EPC bearer creation/modification messages.

Figure 4b shows forwarding states in the data path of SIMECA (upper) and LTE/EPC (lower). As SIMECA classifies packets at network edge, only 2 SDN rules (red arrows) are installed in each base station per C2S-attached *device* as opposed to 4 GTP-U tunnels (*uplink and downlink*) per *EPS bearer* (i.e., according to 8 forwarding states or tunnel IDs) at eNodeB, SGW, and PGW in EPC. Installing SDN rules at base stations also incurs less control signaling than setting up tunnels between multiple components as in EPC: only 2 OpenFlow $flow\_mod$ messages and a RESTful message are needed for each C2S-attached *per device* as opposed to 8 GTP-C and S1AP messages exchanged on S11, S1AP, S5S8 interfaces as in EPC [19] to set up both ends of each tunnel.

For *Attach Request* procedure, SIMECA saves up to 42% of signaling overhead. Because of space limitation, we will discuss in detailed only two control procedures next.

Device Handover: To modify the flows between source/-target base station and the destination, SIMECA incurs 4/5 $flow\_mod$ messages to maintain a C2S/P2P connection of the mobile device. In EPC, to realize path switch and forwarding tunnels to support lossless handover, there are 8 control messages incurred (i.e., 6 GTP-C messages on S11 interface



(a) Packet header translation: packet forwarding example

(b) Tunneling in EPC vs. SDN rules in SIMECA

(c) Handover interaction

Fig. 4: SIMECA operation

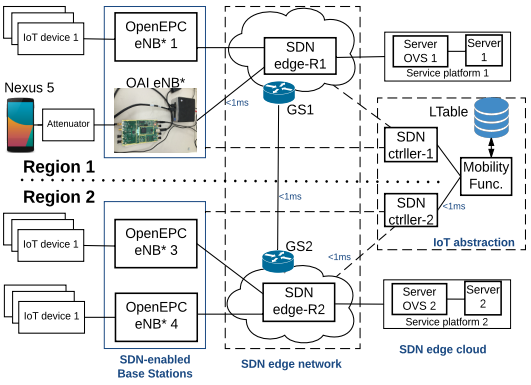


Fig. 5: SIMECA prototype implementation

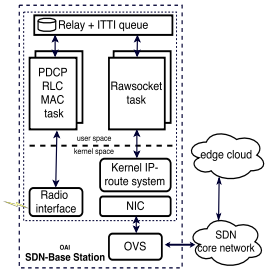


Fig. 6: OAI SDN-enabled base station

and 2 S1AP messages for S1 handover [19]). Compared to LTE/EPC, SIMECA reduces 51% and 49% of control traffic for the two types of handovers.

Control plane interaction. As discussed in § III-B, the control plane consists of the following logical components: *Mobility Functions - MF*, *Mobility SDN controller - MC*. Also, to keep track of device attachment point, a *Device location table-LTable* is used. Because of space limitation, we will only describe the interaction of SIMECA control components during a *Handover* procedure.

Figure 4c shows the interactions when device 1 (D1) handovers and maintains an end-to-end connection with device 2 (D2). At step 1, D1 notifies the MF that it is handing off to base station 3 (BS3). MF makes a request to BS3 and notifies the MC about the handover (step 2). MC looks up D1 and D2's current RIs in the local Ltable (step 4) and installs SDN rules to realize the path switch (step 5): (1) at BS2, packets heading to D1 are now forwarded to BS3 (originally BS1); (2) at BS3, a new pair of SDN rule (downlink/uplink) is installed for D1 so that packets arrive at BS1 will be forwarded to D1; (3) in-flight packets that are heading to BS1 are forwarded to BS3 by a triangle path set up between BS1 and BS3. The MC then updates the table with the new RI (i.e., RI3, assigned by the MF at BS3) for D1 (step 6). This completes the handover procedure.

IV. Prototype Implementation

SDN-enabled base station: Figure 6 shows our base station architecture. We implemented the SDN-enabled base station (S-BS) by combining a node running a refactored OpenAirInterface (OAI) eNodeB implementation [24], with a node running Open vSwitch (OVS) [25]. We reused OAI radio stack and radio control plane (i.e., RRC, PDCP, RLC, MAC/PHY) and modified the relay function in the eNodeB to forward SIMECA's radio traffic to the OVS via raw sockets.

Mobility Function (MF) and Mobility SDN controller (MC): We implemented the MF component by refactoring an OpenEPC [26] MME implementation. We implemented the Mobility SDN controller using the Ryu controller. For intra-region routing we used shortest path forwarding protocol matching on an IP prefix. For inter-region forwarding we used GRE encapsulation. We used IP addresses for SIMECA device identity.

V. Evaluation

We used the PhantomNet testbed for our evaluation [27]. Figure 5 shows the evaluation topology. It consists of two regions each mimicking an edge cloud. The two regions are connected via two gateway switches (GS1, GS2.) We run SIMECA's SDR eNodeB using USRP B210 hardware platform. There are two controllers (MCs) each per region and a shared MF. The location tracking table is implemented using a MySQL database. To mimic regional latency, we set all link latency to sub-millisecond one-way.

We compared SIMECA with an unmodified EPC network instance (OpenEPC [26].) To mimic a realistic LTE/EPC deployment, total end-to-end latency for legacy EPC core is around 20ms [13]. We choose a non-critical health monitoring service with a large number of sensors for our evaluation. We implemented a simple CoAP [28] client using the CoAPthon library [29]. There are 2000 clients generating CoAP requests mimicking body temperature sensor devices sending 2.4 bps traffic with 1.5B packets.

Control plane latency improvement: Figure 7 shows SIMECA's control plane latency for Attach Request procedures. As shown, for intra-region events (*SIMECA*), SIMECA processing time is 76% lower for device attach. For inter-region requests (e.g., device in a region attaches to another device in another region, denoted as *SIMECA-I*), SIMECA is 55% faster (Attach Procedure). The performance improvement of SIMECA is because distributed network functions (i.e., MF,MC) are now closer to users which reduces network latency (black bars.)

End-to-end data plane latency improvement: Figure 8 shows CDF of data plane RTT between the Nexus 5 and a service in SIMECA's edge cloud (i.e., C2S) and between Nexus 5 and another client (i.e., P2P). Average RTT for C2S is 12ms and for P2P 24ms. Note that the end-to-end latency is dominated by radio latency of the OAI. E.g., removing the radio latency, end-to-end latency incurred by SIMECA is less than 1ms (C2S-core and P2P-Core.)

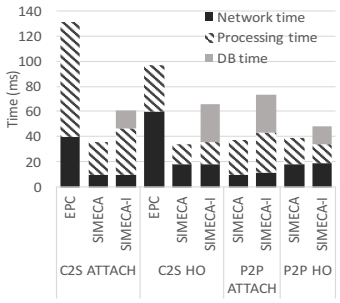


Fig. 7: Control plane time: SIMECA, LTE/EPC

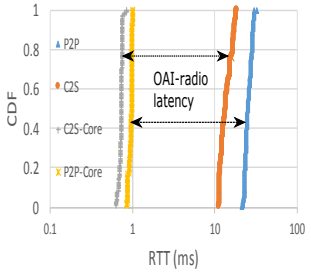


Fig. 8: End-to-end latency between Nexus 5 and service via OAI eNodeB

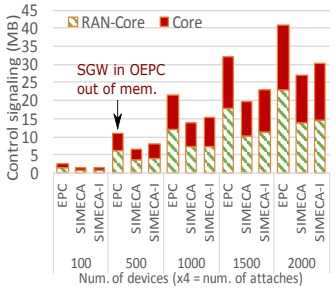


Fig. 9: SIMECA and EPC control overhead: Attach/HO

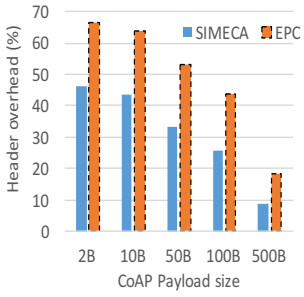


Fig. 10: Packet header overhead: SIMECA and EPC

Control plane overhead improvement: Figure 9 shows the amount of control traffic in SIMECA and EPC as a function of the number of devices. SIMECA has approximately 37% less control traffic for 1000 sensors for intra-region scenario compared to LTE/EPC. The control signaling consists of 2 parts: *RAN-Core* is the signaling between base stations and the SDN edge network and *Core* is the signaling within the SIMECA components. As shown in the figure, most of the reduction in control plane traffic is from the Ran-core part which is the simplified control plane proposed by SIMECA.

As opposed to "normal" OpenEPC, SIMECA's SGW also did not run out of memory when handling a large number of sensors.

Data plane overhead improvement: We measured the packet overhead (i.e., $\frac{payload}{payload+header}$) of SIMECA and EPC with 500 sensors sending packets with a size from 2B to 500B. Figure 10 shows that SIMECA has about 20% less overhead than EPC for CoAP POST requests of less than 100B.

VI. Related work

Leveraging SDN programmability to improve flexibility in cellular networks has been proposed in [5], [6], [30]. SoftMow proposed a scalable recursive SDN control plane for cellular networks. A scalable SDN-based control and data plane to support fine-grained policies has been proposed in SoftCell [6], [30]. Separation of the control and data plane using SDN has been proposed [31], [32]. SIMECA similarly applies the SDN concept for programmability as in SoftCell. However, SIMECA proposes a new connectivity service abstraction (i.e., P2P) realized by light-weight protocol and data plane designed specific for IoT. SoftCell's recursive control plane could be applied to scale SIMECA's control plane. Unlike [31], [32] SIMECA proposes removing GTP tunnels in the data plane.

Offloading of data plane [33], [34], flat cellular network architecture [35], efforts to scale EPC control plane [8], wireless/backhaul resource management to support large number of devices [36], and industrial proposals for edge cloud [37], [22], [21] were proposed to solve the control signaling overload problem in LTE/EPC. SIMECA targets the same problem but with protocol/architectural changes to reduce control/data plane overhead while supporting mobility. SIMECA is also the first to propose an end-to-end design integrating NFV/SDN and edge cloud to support IoT services in LTE networks.

Separating location identity and routing identity to support mobility was proposed in future architectures such as MobilityFirst [38], XIA [39], and LISP [40]. SIMECA applies the same technique to enable mobility but using SDN and was the first to realize it in LTE networks.

VII. Conclusions

We presented SIMECA, a distributed mobile edge cloud architecture that enables a new network service abstraction aiming to suit IoT devices communication models better compared to the LTE/EPC architecture. SIMECA realizes the abstraction by a lightweight control and data planes that significantly reduce signaling and packet header overhead while support seamless mobility. Through evaluations with pre-commercial EPC software, SIMECA shows promising improvements that support large numbers of IoT devices in cellular networks.

REFERENCES

[1] Binh Nguyen. SIMECA: SDN-based IoT mobile edge cloud architecture - PhantomNet tutorial. https://wiki.phantomnet.org/wiki/phantomnet/simeca-sdn-based-iot-mobile-edge-cloud-architecture/.

[2] Gartner. (2013) Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020. http://www.gartner.com/newsroom/id/2636073.

[3] 5GPPP. (2015) 5G Vision - The 5G Infrastructure Public Private Partnership: the next generation of communication networks and services. https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf.

[4] J. Andrews, S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. Soong, and J. Zhang, "What will 5g be?" *Selected Areas in Communications, IEEE Journal on*, 2014.

[5] M. Moradi, L. E. Li, and Z. M. Mao, "Softmow: A dynamic and scalable software defined architecture for cellular wans," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14, 2014.

[6] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "Softcell: Scalable and flexible cellular core network architecture," in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, 2013, pp. 163–174.

[7] A. Banerjee, B. Nguyen, V. Gopalakrishnan, S. K. Kasera, S. Lee, and J. K. V. der Merwe, "Efficient, adaptive and scalable device activation for M2M communications," in *IEEE SECON*, 2015.

[8] A. Banerjee, R. Mahindra, K. Sundaresan, S. Kasera, J. Van Der Merwe, and S. Rangarajan, "Scaling the LTE Control Plane for Future Mobile Access," in *ACM CoNext*, 2015.

[9] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying nfv and sdn to lte mobile core gateways, the functions placement problem," in *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, & Challenges*, ser. AllThingsCellular '14, 2014.

[10] Signaling is growing 50% faster than data traffic. http://tinyurl.com/zl5ckda.

[11] David Nowoswiat. Managing LTE core network signaling traffic. https://techzine.alcatel-lucent.com/managing-lte-core-network-signaling-traffic.

[12] M. Olsson, S. Sultana, S. Rommer, and L. Frid, *SAE and the Evolved Packet Core*. Elsevier, 2009.

[13] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An in-depth study of lte: Effect of network protocol and application behavior on performance," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 363–374.

[14] R. Saunders, J. Cho, A. Banerjee, F. Rocha, and J. Van der Merwe, "P2p offloading in mobile networks using sdn."

[15] F. Ghavimi and H.-H. Chen, "M2m communications in 3gpp lte/lte-a networks: Architectures, service requirements, challenges, and applications," *Communications Surveys & Tutorials, IEEE*, vol. 17, no. 2, pp. 525–549, 2015.

[16] M. Balakrishnan, I. Mohomed, and V. Ramasubramanian, "Where's that phone?: geolocating ip addresses on 3g networks," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 2009, pp. 294–300.

[17] 3GPP TS23.203: Policy and charging control architecture. http://www.qtc.jp/3GPP/Specs/23203-890.pdf.

[18] B.-J. J. Kim and P. S. Henry, "Directions for future cellular mobile network architecture," *First Monday*, vol. 17, no. 12, 2012.

[19] 3GPP. (2015) 3GPP TS 23.401 - General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access. http://www.etsi.org/deliver/etsi_ts/123400_123499/123401/08.14.00_60/ts_123401v081400p.pdf.

[20] N. Alliance, "5g white paper-executive version," *White Paper, December*, 2014.

[21] M. Patel, Y. Hu, P. Hede, J. Joubert, C. Thornton, B. Naughton, J. R. Ramos, C. Chan, V. Young, S. J. Tan, D. Lynch, N. Sprecher, T. Musiol, C. Manzanares, U. Rauschenbach, S. Abeta, L. Chen, K. Shimizu, A. Neal, P. Cosimini, A. Pollard, and G. Klas. (2014) ETSI - Mobile-Edge Computing. www.etsi.org.

[22] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal *et al.*, "Mobile-edge computing introductory technical white paper," *White Paper, Mobile-edge Computing (MEC) industry initiative*, 2014.

[23] 3GPP. Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS). http://www.3gpp.org/DynaReport/24301.htm.

[24] OpenAirInterface. http://www.openairinterface.org.

[25] Open vSwitch. http://openvswitch.org.

[26] OpenEPC. http://www.openepc.com/.

[27] A. Banerjee, J. Cho, E. Eide, J. Duerig, B. Nguyen, R. Ricci, J. Van der Merwe, K. Webb, and G. Wong, "Phantomnet: Research infrastructure for mobile networking, cloud computing and software-defined networking," *GetMobile: Mobile Computing and Communications*, vol. 19, no. 2, pp. 28–33, 2015.

[28] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," 2014.

[29] G. Tanganelli, C. Vallati, and E. Mingozzi, "Coapthon: Easy development of coap-based iot applications with python," in *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*. IEEE, 2015, pp. 63–68.

[30] L. E. Li, Z. M. Mao, and J. Rexford, "Toward software-defined cellular networks," in *Software Defined Networking (EWSDN), 2012 European Workshop on*. IEEE, 2012, pp. 7–12.

[31] M. Sama, L. Contreras, J. Kaippallimalil, I. Akiyoshi, H. Qian, and H. Ni, "Software-defined control of the virtualized mobile packet core," *Communications Magazine, IEEE*, 2015.

[32] K. Pentikousis, Y. Wang, and W. Hu, "Mobileflow: Toward software-defined mobile networks," *Communications Magazine, IEEE*, vol. 51, no. 7, pp. 44–53, 2013.

[33] K. Nagaraj and S. Katti, "Procel: Smart traffic handling for a scalable software epc," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14, 2014.

[34] J. Cho, B. Nguyen, A. Banerjee, R. Ricci, J. Van der Merwe, and K. Webb, "Smore: software-defined networking mobile offloading architecture," in *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*. ACM, 2014, pp. 21–26.

[35] P. Bosch, L. Samuel, S. Mullender, P. Polakos, and G. Rittenhouse, "Flat cellular (umts) networks," in *Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE*. IEEE, 2007, pp. 3861–3866.

[36] M. I. Sanchez, A. Asadi, M. Draxler, R. Gupta, V. Mancuso, A. Morelli, A. De la Oliva, and V. Sciancalepore, "Tackling the increased density of 5g networks: The crowd approach," in *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*. IEEE, 2015, pp. 1–5.

[37] Cisco Fog Computing. http://www.cisco.com/c/en/us/solutions/internet-of-things/iot-fog-computing.html.

[38] MobilityFirst Project. http://mobilityfirst.winlab.rutgers.edu.

[39] A. Anand, F. Dogar, D. Han, B. Li, H. Lim, M. Machado, W. Wu, A. Akella, D. G. Andersen, J. W. Byers *et al.*, "Xia: An architecture for an evolvable and trustworthy internet," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*. ACM, 2011, p. 2.

[40] D. Farinacci, D. Lewis, D. Meyer, and V. Fuller, "The locator/id separation protocol (lisp)," 2013.