A Deployable Containerized 5G Core Solution for Time Critical Communication in Smart Grid

Van-Giang Nguyen, Karl-Johan Grinnemo, Javid Taheri, and Anna Brunstrom
Department of Computer Science
Karlstad University, Karlstad 651 88, Sweden
Email: (giang.nguyen, karl-johan.grinnemo, javid.taheri, anna.brunstrom)@kau.se

***Abstract*—Communication within substatation automation systems in a smart grid environment is often time-critical. The time required for exchanging information between intelligent devices should be within a few milliseconds. The International Electrotechnical Commission (IEC) 61850 standard has proposed the Generic Object Oriented Substation Event (GOOSE) protocol to achieve this goal. However, the transmission of GOOSE messages is often limited to a small Local Area Network (LAN). In this demo, we demonstrate the feasibility of using 5G for GOOSE-based time critical communication in a large-scale smart-grid environment, and present a deployable 5G core solution using container-based virtualization technology. The radio part of the demo is emulated. The demo also shows that the delay introduced by the core network is in the order of sub milliseconds, while the one-way delay without a real radio access network is less than 1 ms, which is well below the total delay budget for GOOSE.**

I. Introduction

A smart-grid substation automation system is a system which provides protection, control, monitoring, and automation within a power grid system. These tasks strictly rely on the underlying communication network, where related information is reported from or is exchanged between so-called Intelligent Electronic Devices (IEDs), i.e., microprocessor-based sensors and actuators in electric grids. Such communication often has special performance requirements, especially very low latency. The IEC 61850 standard [1] defined the GOOSE protocol for transferring time-critical information, such as control commands and alarms, between IEDs with millisecond transfer delays. As depicted in Fig. 1, there are two different versions of GOOSE: an Ethernet- and an IP-based version. The Ethernet-based version (IEC 61850-8-1) is the oldest of the two versions. In this version, GOOSE messages are mapped directly to Ethernet data frames. The IP-based version (IEC 61850-90-5) is a more recent version that enables GOOSE messages to be routed over IP-based networks. In order to increase the range of the protection scheme, mobile cellular technologies like 4G have been considered as one of the technology candidates [2], [3]. However, the one-way delay reported from these works is in the range of 15 to 20 ms. Currently, the fifth generation mobile networks (5G) is being developed. With the adoption of advanced technologies and radical changes in the architecture, 5G will support a variety of services including critical Machine-Type Communication (cMTC) [4] in which 5G is promised to provide a latency of

978-1-7281-5127-4/20/$31.00

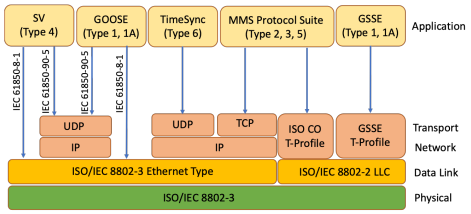

Fig. 1. IEC 61850 Protocol Stacks [1]

1 ms between machines or Internet of Things (IoT) devices. In view of this, this demo shows how 5G can be used to transport GOOSE messages within the GOOSE delay budget of 4 ms [1]; especially, we show that our solution in comparison to the proposals in [5], requires no modification of the mobile network protocol stack, thus making it deployable with a minimum effort. Fig. 2 shows the mapping between 5G Evolved Packet Core (EPC) and GOOSE protocol stacks in the data plane. To be able to take advantage of its inherent multicasting support and low protocol overhead, this paper uses the Ethernet-based GOOSE version. We propose the use of GOOSE gateways to carry GOOSE frames through 5G using the Generic Routing Encapsulation (GRE) tunneling protocol (as shown in Fig. 2). Many approaches have been proposed to help further improve the performance of the core part of the mobile network. Virtualizing the core network architecture using Network Function Virtualization (NFV) technology [6] is one of the main approaches, especially for the development of the 5G core network. Virtual machine- and container-based deployments are the two most popular deployment solutions of NFV. However, due to the anticipated benefits offered, such as being a more lightweight solution with a small memory footprint, containerizing the 5G core network functions promises to not only save resources but also improve the overall performance. In this demo, we use the Open5GCore platform [7] which provides standards compliant components to build an evolved packet core network for 5G. We run the Open5GCore instances on Docker containers. The Rapid61850 tool [8] is used to emulate the GOOSE traffic. For the demo purpose, we run all 5G core components and the GOOSE gateways as containers inside one server.

The remainder of the paper is organized as follows. Our demo setup is described in Section II, and Section III presents the demo workflow. Finally, we conclude the paper in Section IV.

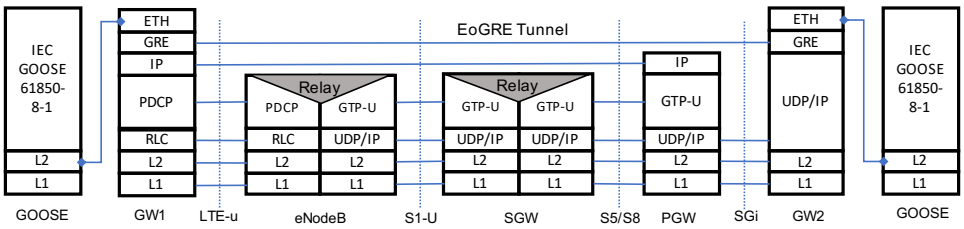

Fig. 2. LTE/5G user plane protocol and GOOSE mapping

II. Demo Setup

In this section, we describe the our demo setup which includes the 5G core network components, the GOOSE components, and the monitoring utilities. The overall system architecture is shown in Fig. 3.

A. 5G Core Components

The 5G core network is emulated using the Open5GCore platform, which is the most standards-compliant implementation of a 5G Evolved Packet Core (EPC) system available. It includes an eNodeB, a MME-SGWC-PGWC, a HSS, and a SGWU-PGWU. The eNodeB entity is a software implementation of a LTE base station, and is responsible for providing communication between user devices and the mobile network. The MME-SGWC-PGWC entity is an implementation of a Mobility Management Entity (MME), the control functions of a Serving Gateway (SGW) and a Packet Data Network Gateway (PGW). This entity is mainly responsible for handling control signaling traffic such as attachment requests sent over the S1-C interface from the eNodeB. The HSS entity runs a Home Subscriber Server (HSS), which is a database storing user subscription information. The SGWU-PGWU entity is used to route the user plane traffic between the user and the external networks (e.g., the Internet) sent over the S1-U interface where the user packets are encapsulated using General packet radio service Tunneling Protocol (GTP). All the 5G core network functions are containerized using a Docker engine on a Ubuntu 18.10 server. The networks between containers are created using Macvlan Docker networking [9].

B. GOOSE Components

The GOOSE components include a GOOSE publisher, a GOOSE subscriber, and two GOOSE gateways. The publisher runs the Rapid61850 software [8] which allows us to emulate different types of IEC61850 traffic including Ethernet-based GOOSE. It connects to the 5G core system from the eNodeB side. The transmission schema is defined in a so-called Substation Configuration Description (SCD) file which is then used to generate GOOSE messages to a list of subscribers according to a traffic pattern. GOOSE messages are mapped directly to Ethernet data frames on a predefined VLAN and and with a pre-specified priority level (e.g., VLAN 3, priority 4). The subscriber is located at the other end of the 5G network. It listens to GOOSE messages on the same VLAN as the publisher (i.e., VLAN 3). Both the publisher and subscriber run on Raspberry Pi devices as shown in Fig. 3. In order to send Ethernet-based GOOSE messages over the 5G network, which is IP-based, two GRE-enabled GOOSE gateways are required, as shown in Fig. 3. The first gateway is located between the publisher and the 5G network. It has the same protocol stack as a User Equipment (UE), and is able to subscribe to the 5G network using its International Mobile Subscriber Identity (IMSI). It connects to the eNodeB using the emulated LTE-U interface, which does not include the implementation of a real radio stack. The other gateway is located between the 5G network (the SGWU-PGWU entity) and the subscriber on the SGi interface. Finally, an Ethernet-over-GRE (EoGRE) tunneling is established between these two gateways, thus enabling the transmission of GOOSE messages encapsulated within Ethernet frames over the 5G network. For the demo purpose, the GOOSE gateways are also containerized in the same Docker server as the 5G core components.

C. Monitoring Components

To monitor the performance of the system, we use a set of tools, namely Metricbeat [10], InfluxDB [11], and Grafana [12]. Metricbeat is a lightweight monitoring tool, collecting system information such CPU, memory, and network, as well as application information. In our system, Metricbeat is running as an agent inside each container, as shown in Fig. 3. The collected information is stored in InfluxDB, which is an open source, time series database. Lastly, Grafana is used to visualize the statistics stored in InfluxDB. As follows from Fig. 3, InfluxDB, and Grafana are installed in the same container, the Monitor container. This container will also have information about the data plane reported from the flowmon agent running on each Open5GCore's component. To monitor the one-way delay of a GOOSE transmission in the core network, we have a tool which captures GOOSE messages at the ingress and egress interfaces of the 5G Docker server using tshark [13]. The main window of the tool is shown in Fig. 5. This tool extracts timestamps from arrived and departed GOOSE frames, calculates the frame sojourn time, and plots the sojourn times in real time.

III. Demo Workflow

In order to perform the transmission of GOOSE over 5G, a data plane route or a data path must be setup in advance. The first GOOSE gateway (GOOSE GW1) sends an attachment

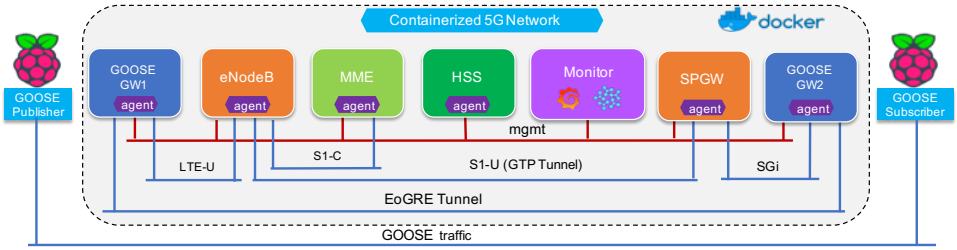

Fig. 3. Demo setup diagram

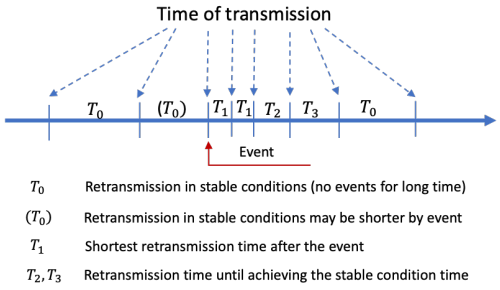

Fig. 4. GOOSE transmission pattern

request to the network through the LTE-U interface. This request contains an IMSI information which will be compared to the subscriber information stored in the HSS container. Once the attachment is successful, an IP address is allocated to this gateway. This IP address is used to establish the EoGRE tunnel with the other GOOSE gateway, whose IP address is also known. Once the EoGRE tunnel is up, the publisher can start generating GOOSE traffic. According to the IEC 61580 standards, GOOSE operates in two different states: a stationary and a critical state. In the stationary state, each IED reports its status every T_0 seconds via identical GOOSE messages to neighboring IEDs. In the critical state, the transmission interval is shortened to milliseconds every time a new event occurs, to ensure the timeliness of the message delivery [5]. In this demo, we show the operation of GOOSE in both states by varying the transmission interval, as shown in Fig. 4. In an experiment, where we use a GOOSE emulation machine for running both the publisher and the subscriber, we evaluated the performance of GOOSE transmission for message sizes of 172 bytes with different transmission intervals such as 10, 20, and 50 milliseconds. The average one-way delay between the publisher to the subscriber is less than 0.6 ms, while the average Open5GCore delay is about 0.3 ms, as shown in Fig. 5. Thus, with the maximum delay budget for a 5G radio access network of 1 ms [14], the total end-to-end delay is well below the maximum permitted delay for GOOSE of 4 ms. In other words, our experiment suggests reasonable to believe that time-critical GOOSE traffic could indeed be transferred over 5G.

IV. Conclusion

In this demo, we show an easy-to-deploy solution for time-critical communication using an Ethernet-based GOOSE protocol over a containerized 5G network. We validate and evaluate the performance of the system in terms of one-way delay targeting the 5G core network. In spite of no real radio access network, which is the limitation of our demo, the obtained results still indicate the feasibility of in using 5G as a transport medium for GOOSE transmission in a smart grid.

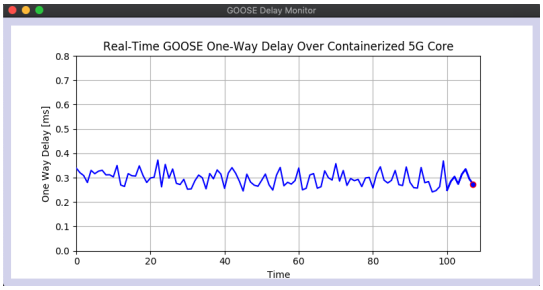

Fig. 5. Real-time GOOSE delay monitor

Acknowledgment

The work was supported by the High Quality Networked Services in a Mobile World (HITS) project funded by the Knowledge Foundation of Sweden and the 5GENESIS project funded by the European union.

References

[1] IEC, "Communication networks and systems for power utility automation," https://www.iec.ch/smartgrid/standards/.
[2] C. Kalalas *et al.*, "Enabling IEC 61850 communication services over public LTE infrastructure," in *IEEE ICC*. IEEE, 2016.
[3] G. Bag *et al.*, "Performance evaluation of IEC 61850-90-5 over a latency optimized 3GPP LTE network," in *IEEE SmartGridComm*. IEEE, 2018.
[4] V.-G. Nguyen *et al.*, "5G mobile networks: Requirements, enabling technologies, and research activities," in *Comprehensive Guide to 5G Security*, M. Liyanage, *et al.*, Eds. John Willey & Sons Ltd., 2018.
[5] J. Cheng, B. Kovács, and M. Darula, "Proposal for IEC GOOSE transport in 5G networks," 2018, Technical Report.
[6] V.-G. Nguyen *et al.*, "SDN/NFV-based mobile packet core network architectures: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.
[7] Open5GCore. http://www.open5gcore.org/.
[8] Rapid61850. https://github.com/stevenblair/rapid61850.
[9] Macvlan. https://docs.docker.com/network/macvlan/.
[10] Metricbeat. https://www.elastic.co/products/beats/metricbeat.
[11] InfluxDB. https://www.influxdata.com/.
[12] Grafana. https://grafana.com/.
[13] "Tshark," [Online] Available: https://linux.die.net/man/1/tshark.
[14] ITU-R, "Minium requirements related to technical performance for IMT-2020 radio interface(s)," Nov. 2017.