Demonstration of Cloud-Based Streaming Telemetry Processing for Optical Network Monitoring

Jesse E. Simsarian(1), Gurudutt Hosangadi(1), Wolfgang Van Raemdonck(2), Jurgen Gripp(3), Matthew Nance Hall(4), Jiakai Yu(5), and Theodore Sizer(1)

(1) Nokia Bell Labs, Murray Hill, New Jersey, USA, jesse.simsarian@nokia-bell-labs.com,
(2) Nokia Bell Labs, Antwerp, Belgium,
(3) Nokia ION Optics, Murray Hill, New Jersey, USA,
(4) University of Oregon, Computer and Information Science, Eugene, OR, USA,
(5) College of Optical Sciences, University of Arizona, Tucson, AZ, USA

Abstract *We present a demonstration of the processing of streaming telemetry data from an optical network and machine-learning based video analytics on a cloud-based stream-processing platform. Real-time processing enhances network security and reliability by combining information from diverse sources.*

Introduction

As we become ever more reliant on communications infrastructures for human-to-human and emerging machine-to-machine[1] interactions, maintaining high reliability in optical networks is crucial. By using coherent transponders that detect the field transmitted through deployed optical fibers, there are new opportunities to sense the environment[2] and monitor the network by classifying events[3]. Furthermore, gNMI/gRPC[4] protocols have been introduced to efficiently[5] stream monitoring data from optical network elements. Previously, telemetry streams have been stored in a database and utilized in a software defined networking controller of the optical network elements[6]. However, with the monitoring data being generated as streams of information, it becomes advantageous to process the telemetry data as event streams for flexible and real-time analysis and alarm generation[7],[8].

With the advent of 5G networks, a massive number of internet of things (IoT) devices such as environmental sensors can be supported by the wireless network infrastructure. By monitoring the physical network infrastructure with IoT devices and correlating that with telemetry streams from the optical network elements (see Fig. 1), additional insight on network operations can be gained.

In this paper and demonstration, we provide information on the methods utilized and demonstrate the operation of the stream processing of streaming telemetry data from an optical network combined with video monitoring of the network infrastructure with machine-learning based person identification[8]. The alarm correlations can be used to improve anomaly detection and to prevent network outages from intentional[9] or unin-

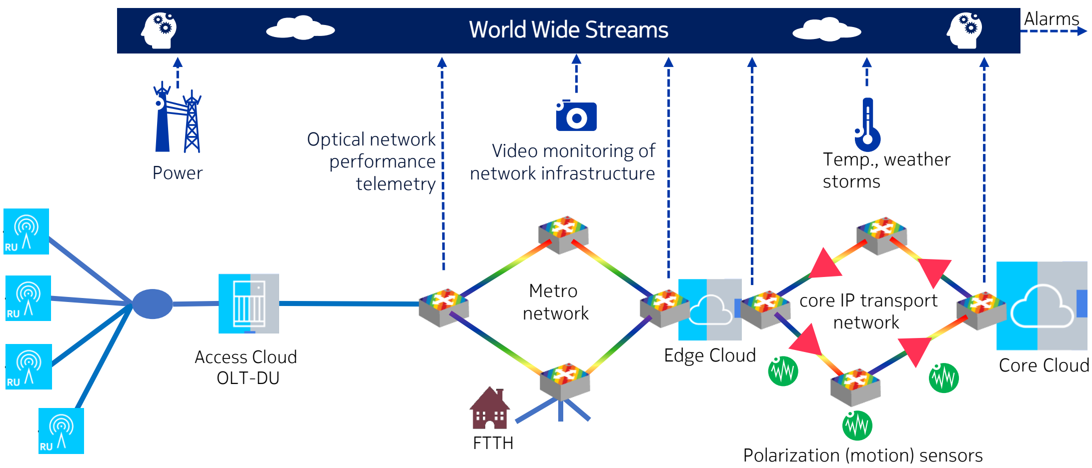

Fig. 1: Monitoring 5G networks with diverse sensor data including streaming telemetry from the optical network.

978-1-6654-3868-1/21/$31.00 ©2021 IEEE

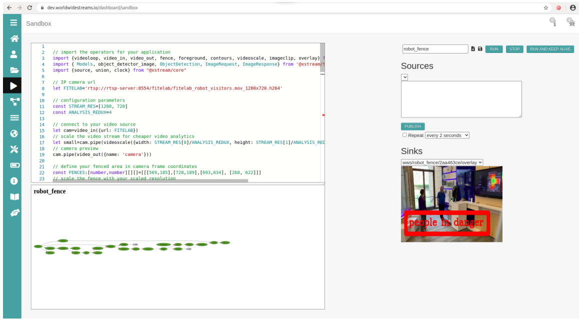

Fig. 2: Online tool based WWS application creation

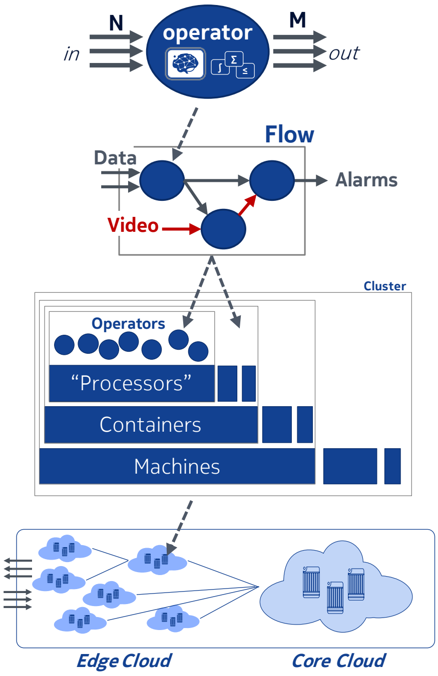

Fig. 3: Stream processing platform

tentional events caused by human intervention. The cloud-based World Wide Streams[10] platform processes the data in real time for alarm generation and correlation as shown in Fig. 1.

Stream Processing Platform

World Wide Streams (WWS)[10] is a distributed stream processing platform. WWS offers two key features: (1) creation of dataflow-based applications in an online authoring tool, see Fig. 2, and (2) automated distribution of the dataflow over a wide-area network. An application is composed of a set of operators, each implementing a

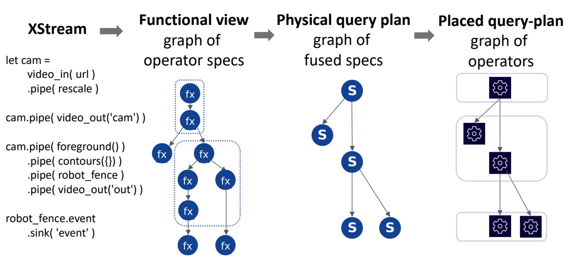

Fig. 4: Deployment process

stream-processing function. These functions can be diverse, from a basic time-window function implemented in JavaScript to an object classifier for video streams running on top of a machine learning (ML) framework like TensorFlow. This allows the creation of services that can combine both event stream as well multimedia stream processing.

The goal of WWS is to deploy services over distributed runtimes, called processors. A WWS processor groups similar operators that share implementation technology or runtime characteristics. For example, a processor can be a NodeJS[11] based event-loop optimized to run large groups of light-weight lambda functions. At the other end of the spectrum, another processor schedules each operator on an OS process, running a GStreamer[12] pipeline for stateful media processing. The processor abstraction also allows for optimizations for long-lived stateful transformations versus using a worker model to schedule scalable stateless tasks.

As illustrated in Fig. 3, the processor itself is deployed as a container on a cluster of machines. Each cluster can be deployed at different locations in the network like the edge or core cloud. WWS will interconnect these clusters into one large distributed processing platform.

WWS applications are written in XStream[13], a typed domain-specific functional programming language embedded in TypeScript[14]. XStream helps the developer in selecting and combining the right operators into more complex stream processing services. The XStream compiler generates a graph of operator specifications and optimizes this graph by fusing operators sharing the same technology that can coexist on the same processor (Fig. 4). The WWS deployer places each fused operator on a runtime in the network and the WWS platform takes care of the data streaming between the operators within one cluster or handles the data forwarding to connect operators running across different clusters. Operators can have direct communication with each other over sockets or shared memory e.g. if they run in the same processor. Otherwise operators communicate via a broker, in this case the broker offers the fan out towards multiple consumers and provides loose coupling between operators. WWS uses RabbitMQ[15] as the event broker and a RTMP[16] and WebRTC[17] media server for media streams.

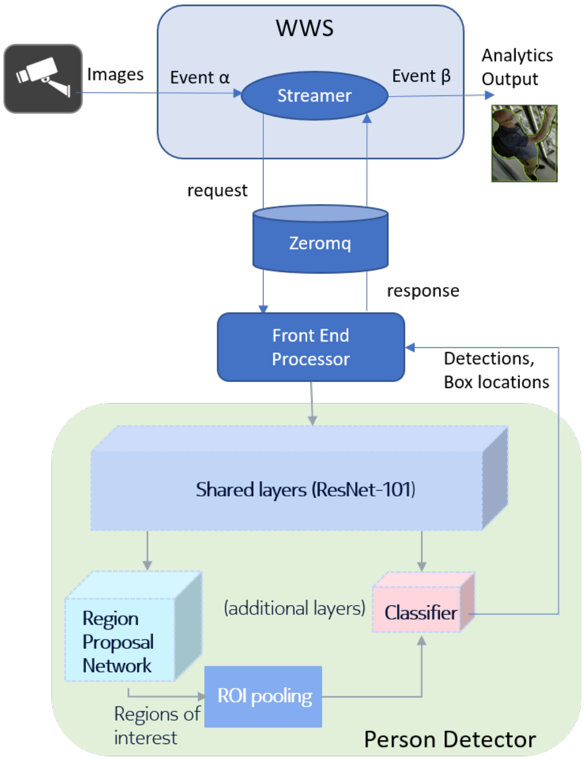

Fig. 5: Processing pipeline for person detection

Video Stream Processing

The role of the video stream processing pipeline is to process sensed images from the IP camera overlooking the monitored area, detect persons in the scene at any point in time and then report these events back for use by the overall system.

We rely on the WWS platform to manage the streaming between camera, video analytics and the rest of the system. The details of the video stream processing pipeline are shown in Figure 5. Specifically, once WWS receives a camera image, it uses a request-reply protocol based on ZeroMQ to request analysis on the image. The information sent comprises of the image and a metadata template for the analytics pipeline to fill in case of a person detection. The analytics pipeline uses a two-stage deep learning network (mask-rcnn-resnet-101)[18]. The first stage (region proposal network) of this machine learning network proposes regions in the image where objects of interest are likely to be found. This is followed by a second stage (classifier) where a region is classified as "person" if the features extracted for this region matched apriori learned "person" features. This classification leads to a "person detection" event which is relayed back to WWS via a reply message after filling in information such as bounding box and contour locations of the person in the metadata template that was previously received along with the image. Note that both stages share a common block (labeled "Shared Layers" in Figure 5) for feature extraction.

We implemented the ML in Python, Keras and TensorFlow using the code at[19] as a starting point. This is Dockerized[20] and run on a GPU to provide a person-detection microservice to WWS via a ZeroMQ socket as described in previous work[8].

Demonstration

We present a demonstration using a network testbed located at Bell Labs in New Jersey consisting of six commercial Nokia 1830 PSS optical nodes with flexgrid reconfigurable optical add drop multiplexers and 2200 km of fiber[8]. We monitor the performance metrics of seven Nokia flexible line cards with variable bitrate operated at 200 Gb/s with 8QAM modulation format using gNMI with 10 s period. By attenuating the optical power of one of the wavelengths, the resulting signal-to-noise ratio degradation causes the preFEC bit error ratio (BER) to rise. We simultaneously monitor one of the terminating nodes of the wavelength with a camera that utilizes the machine learning based person identification process running on WWS. We detect point anomalies of the BER by using stream processing of the optical telemetry data stream using a time-window detection method on the WWS platform. The mean, μ, and standard deviation, σ, of the previous 29 data points are calculated over the window, and $z(t)$ of the most recent data point, $d(t)$, is given by $z(t) = (d(t) - \mu)/\sigma$. Anomalies in the BER are alarmed when $z(t)$ crosses a threshold, z_{th}.

We correlate the person detection alarm with BER anomaly detection processes on wavelengths that terminate at the monitored node. Due to the different periods of the person detection measurements and optical telemetry stream updates, we resample the streams before generating the correlation alarm[8].

Conclusions

We demonstrate a scalable and cloud-based stream processing for the flexible generation and correlation of optical alarms with ML-based person detection to improve network awareness and give insight to operators when optical networks are impaired. Employing IoT devices external to the transport system broadens our view of the network physical environment beyond information available from the optical network elements.

References

[1] N. Benzaoui, “Deterministic latency networks for 5G applications”, in *Proc. ECOC’20*, Brussels, Belgium, Dec. 2020.

[2] Z. Zhan, M. Cantono, V. Kamalov, A. Mecozzi, R. Müller, S. Yin, and J. C. Castellanos, “Optical polarization-based seismic and water wave sensing on transoceanic cables”, *Science*, vol. 371, no. 6532, pp. 931–936, 2021.

[3] K. Guan, J. Simsarian, F. Boitier, D. Kilper, J. Pesic, and M. Sherman, “Efficient classification of polarization events based on field measurements”, in *Proc. OFC’20*, San Diego, CA, Mar. 2020, Th3D.7.

[4] (2021), [Online]. Available: `https://www.grpc.io/`.

[5] R. Vilalta, N. Yoshikane, R. Casellas, R. Martínez, S. Beppu, D. Soma, S. Sumita, T. Tsuritani, I. Morita, and R. Muñoz, “GRPC-based SDN control and telemetry for soft-failure detection of spectral/spacial superchannels”, in *Proc. ECOC’19*, Dublin, Ireland, Sep. 2019, pp. 1–4.

[6] F. Paolucci, A. Sgambelluri, F. Cugini, and P. Castoldi, “Network telemetry streaming services in SDN-based disaggregated optical networks”, *J. Lightwave Technol.*, vol. 36, no. 15, pp. 3142–3149, 2018.

[7] A. Sadasivarao, S. Syed, D. Panda, P. Gomes, R. Rao, J. Buset, and L. Paraschis, “Demonstration of extensible threshold-based streaming telemetry for open DWDM analytics and verification”, in *Proc. OFC’20*, San Diego, CA, Mar. 2020, M3Z.5.

[8] J. E. Simsarian, M. N. Hall, G. Hosangadi, J. Gripp, W. V. Raemdonck, J. Yu, and T. Sizer, “Stream processing for optical network monitoring with streaming telemetry and video analytics”, in *Proc. ECOC’20*, Brussels, Belgium, Dec. 2020, Tu2K–1.

[9] T. Hughes, *Attacks show fiber optic internet cables vulnerable*, `https://www.usatoday.com/story/news/2015/09/16/attacks-show-fiber-optic-internet-cables-vulnerable/32502785/`, 2015.

[10] (2021), [Online]. Available: `https://www.worldwidestreams.io`.

[11] (2021), [Online]. Available: `https://nodejs.org/en/`.

[12] (2021), [Online]. Available: `https://gstreamer.freedesktop.org/features/`.

[13] (2017), [Online]. Available: `http://soft.vub.ac.be/~tvcutsem/talks/presentations/XStream_ifip17.pdf`.

[14] (2021), [Online]. Available: `https://www.typescriptlang.org/`.

[15] (2021), [Online]. Available: `https://www.rabbitmq.com/`.

[16] (2012), [Online]. Available: `https://wwwimages2.adobe.com/content/dam/acom/en/devnet/rtmp/pdf/rtmp_specification_1.0.pdf`.

[17] (2021), [Online]. Available: `https://webrtc.org/`.

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn”, in *Proc. IEEE International Conference on Computer Vision (ICCV’17)*, Venice, Italy, Oct. 2017, pp. 2980–2988.

[19] W. Abdulla, *Mask r-cnn for object detection and instance segmentation on keras and tensorflow*, `https://github.com/matterport/Mask_RCNN`, 2017.

[20] (2021), [Online]. Available: `https://www.docker.com/`.