Adaptive Edge and Fog Computing Paradigm for Wide Area Video and Audio Surveillance

Ioannis Ledakis, Thanassis Bouras
Ubitech Ltd.
Athens, Greece
{gledakis, bouras}@ubitech.eu

G.Kioumourtzis and M.Skitsas
Advanced Integrated Technology Solutions & Services
ADITESS Ltd.
Nicosia, Cyprus
{research, mskitsas}@aditess.com

***Abstract*— Surveillance systems that capture video and audio in enterprise facilities and public places produce massive amounts of data while operating at a 24/7 mode. There is an increasing need to process, on the fly, such huge video and audio data streams to enable a quick summary of "interesting" events that are happening during a specified time frame in a particular location. Concepts like fog computing based on localisation of data processing will relax the need of existing cloud-based solutions from extensive bandwidth and processing needs at remote cloud resources, however, the abilities of data processing on the extreme edge are limited by the hardware capabilities of the devices. In this paper, we describe a novel, adaptive architecture and that builds on top of a distributed computing paradigm and is ideal for smart surveillance systems that can utilize resources at cloud, fog and edge. We provide the main architectural components, the hardware options and key software components of the system. The proposed architecture uses cloud, edge and fog computing concepts. Edge computing is realized by a camera embedded system, cloud computing with the usage of public accessible infrastructure for data processing and fog computing for the processing and data fusion of video streams in small areas.**

Keywords— surveillance system; edge computing; fog computing; adaptive architecture; distributed computing

I. INTRODUCTION

Current Surveillance systems in enterprise facilities and public places produce massive amounts of video and audio content while operating at a 24/7 mode. Typically, video clips are stored as compressed video files corresponding to video for several hours. Due to the prohibitively high requirements of surveillance video and the need for continuous monitoring, currently closed-circuit television (CCTV) systems store video on local devices. For example, the storage needs that are created within the duration of one hour for 1000 video streams of video surveillance, tens of terabytes of data are generated and stored[1]. In modern, smart surveillance systems, the massive amount of data generated from the camera (or other sensors) has to be transferred from Internet-of-Things (IoT) devices level to a centralized processing unit for analysis[2] of the footage and only then proceed with the storage of the incident footage. Even with usage of video encoding standards like MPEG-4 and H.264 to achieve high compression rate, video remains a very resource hungry type of multimedia, especially if taken into account that video analytics nowadays receive as an input HD or Full HD video, increasing substantially the strain on the hosting infrastructure for real time processing and provision. Often, the situation becomes more complicated when other types of data are involved in the processing (e.g. feeds from social media, other video sources like YouTube, Periscope, sensor systems, etc.)[1].

Therefore, there is an increasing need to create smart surveillance systems that can process, on the fly, such huge video data streams to enable a quick summary of "interesting" events that are happening during a specified time frame in a particular location. In the same time, new technological solutions and infrastructures are utilized in order to support the smarter and more efficient surveillance.

The remainder of this paper is organized as follows. Section II presents existing computing paradigms and solutions related to the data processing on smart surveillance systems. In section III, we present a distributed data processing paradigm for the case of wide area surveillance while in Section IV we discuss the architecture of a surveillance system that can utilize the aforementioned distributed processing paradigm. Details of the fog and edge nodes used for the implementation of this architecture are presented in Section V. Finally, we conclude our work in Section VI.

II. DISTRIBUTED PROCESSING PARADIGMS AND WIDE AREA VIDEO SURVAILANCE

In a dynamic IoT-based and Big Data driven environment like in the case of Wide Area Video and Audio Surveillance, there are still valid challenges with respect to efficient and reliable real-time processing. Several issues need to be addressed, including data migration over bandwidth limited and high latency communication networks, and the heterogeneous nature of data obtained from the surveillance systems[2]. Academic and commercially-successful platforms (Apache Hadoop, Apache Spark) with tremendous corporate backing (Amazon, Microsoft, Google) still present high barriers to entry [3] and, in fact, taking advantage of elasticity remains challenging for even sophisticated users, as the majority of these frameworks were designed to first target on-premise installations at large scale [4] and are not applicable for real-time processing due to their static nature [5]. Moreover, recent and dynamic technologies like STORM, offer a very efficient computational solution for real-time processing, however they don't exploit new decentralised

paradigms (e.g. distributed clouds, edge [6] and fog computing [7].

Cloud computing is commonly used for real-time data-intensive applications due to the theoretically boundless scalability power. Nevertheless, there are arguments that highlight as a major potential downside of cloud computing the slower-than-desirable performance due to networks' bandwidth limitations [8], and also on commercial cloud platforms, users have to take ahead of time many decisions for the instance type, cluster size, pricing model, programming model, and task granularity [4]. The problem of how to process efficiently on the cloud is becoming more acute as more and more objects become "smart," or able to sense their environments, connect to the Internet, and even receive commands remotely. Modern 3G and 4G cellular networks simply aren't fast enough to transmit data from devices to the cloud at the pace it is generated. The above facts led the research community to new concepts on fog computing since they store and process the torrent of data being generated by the Internet of Things (IoT) on the "things" themselves, or on devices that sit between the "things" and the Internet. According to Cisco [7], the fog extends the cloud to be closer to the things that produce and act on IoT data. These devices, called fog nodes, can be deployed anywhere with a network connection: on a factory floor, on top of a power pole, alongside a railway track, in a vehicle, or on an oil rig. Any device with computing, storage, and network connectivity can be a fog node. Examples include industrial controllers, switches, routers, embedded servers, and video surveillance cameras. IDC estimates that the amount of data analysed on devices that are physically close to the Internet of Things is currently approaching 40 percent [9]. This is quite reasonable, since latency reduction for quality of service (QoS) and edge analytics/stream mining, result in superior user-experience [10] and redundancy in case of failure ([11], [12]).

An approach that is related to fog computing and distributed computing in general is cyber foraging. Cyber foraging term coined by M. Satyanarayanan in his 2001 paper entitled "Pervasive Computing: Vision and Challenges"[13], and described the pervasive computing technique where resource poor, mobile devices offload some of their heavy work to stronger surrogate machines in the vicinity. According to cyber foraging vision, mobile devices offload computation to servers called surrogates, which are located in close proximity to the mobile device. To facilitate this, M. Satyanarayanan stated that it was necessary to develop systems to partition applications between local and remote resources and to manage and secure the surrogates, data, and devices needed for offloading computation [14]. The discussion of cyber foraging has been active mainly in research field and the original vision still remains an important part of pervasive computing, but compelling applications are missing that will make cyber foraging a widely deployed technology [15].

Only few attempts research attempts have been made to develop scalable and lightweight distributed surveillance systems that make use of cyber foraging as part of integrated framework for Internet-of-Things (IoT) and cloud computing. In specific, A. Ali et al. [2] proposed the concept of bringing cloud closer to the IoT, and in order to provide real-time on-site object detection, it is needed to instantiate a cloudlet on resource-rich nearby infrastructure which is connected to the IoT devices for distributed retrieval and processing of critical and sensitive data.

Driven by the latest developments, the analysis presented above and the facts in wide area video surveillance, in this paper we present a novel surveillance system based on adaptable usage of cloud, fog and edge computing. Our solution involves small computing devices at the sensor level (camera) that provide a first level of processing based on "light" video and audio analytics. The various video streams from the same location (e.g. building) are fused to fog nodes where a second level processing is taking place. A private cloud infrastructure with higher computing and processing capacity can be utilised to cope with unpredicted workload of a diverse number of video streams. Therefore, we actually extend the cyber foraging approach with allowing the effortless offloading of computational work not only to surrogates, but also to edge devices with close proximity and limited processing capabilities.

III. Distributed DAta Processing Paradigm On Wide Area Video ANd Audio Surveillance Systems

As described in the introductory section, the creation of smart CCTV technology is a measure for increasing security, as with current architectures where many human factors are involved in the chain from event detection to Police notification, the response to security threats is low and from operational perspective, it is hard for security personnel to monitor many security cameras simultaneously. In the same time the need to physically transport video surveillance footage is a bandwidth hungry application area.

To address the identified gaps and make existing CCTV surveillance systems more efficient, a novel paradigm is proposed that can be used for distributed data processing. We also present how this paradigm can be grounded for the creation of smart surveillance systems that are efficient, adaptive and highly scalable.

A. Proposed Solution

For creating a distributed data processing solution suitable for wide area surveillance systems, we suggest the usage of serverless execution of stateless functions in order to create a fundamentally elastic, distributed data processing system with functions that can be executed at cloud, fog or edge resources.

Serverless computing is a new execution model that is currently emerging to transform the design and development of modern scalable applications. By design serverless is an event-driven computing model where underlying infrastructure (including physical and virtual hosts, virtual machines, containers as well as the operating systems) is abstracted from the developer and the service consumer. Applications run in stateless containers that are spawned based on triggering of events. In this model, a function is deployed once and is

invoked repeatedly whenever new inputs arrive and elastically scales with input size[2].

The evolution of serverless computing is reinforced by the continuous advances in container-based technology together with the consolidation of cloud computing platforms[4]. Recently cloud providers developed infrastructure to run event-driven, stateless functions, with AWS Lambda [16] being the first serverless computing service to appear, followed by Google Cloud Functions [17], Microsoft Azure Functions [18], and the open source platforms of OpenLambda[19] and Apache OpenWhisk [20].

In 2016 Villamizar et al.[21], completed a cost comparison research by comparing a service built with a) monolithic architecture, b) microservices architecture, and c) utilizing AWS Lambda serverless functions. The results of this research shown that developing an application using AWS Lambda reduces infrastructure costs more than 70% and guarantees the same performance and response times as the number of users increases.

With these findings in mind, we developed a distributed development paradigm and architecture that mimics serverless computing in order to achieve the transparent scalability and distribution, while adapting to the need for reliable, continuous, real-time, processing of big data streams as close as possible to the edge.

The key insight in the suggested parading is to create functions that execute common computation intensive tasks like transcoding on data coming from data streams. To achieve this data processing system that inherits the elasticity of the serverless model while addressing the simplicity for end users, functions are executed using Java Parallel Processing Framework, (JPPF) [22] and Operating-system-level virtualization (containers).

JPPF is a software solution enabling the use of grid computing in applications, demanding minimum code modification. Simply put, JPPF enables applications with large processing power requirements to be run on any number of computers, in order to dramatically reduce their processing time [23]. This is done by splitting an application into smaller parts that can be executed simultaneously on different machines.

Dividing an application into smaller parts that can be executed independently and in parallel. JPPF can easily and seamlessly scale from a standalone application on a single machine, to a massively distributed grid of machines comprising thousands of compute nodes. For the case of the Wide Area Video Surveillance needs, we use JPPF in order to implement the fragmentation mechanism facilitating the distribution of processing tasks based on functions of the application and also to enable the onloading or offloading of processing to edge, fog and cloud resources. This way that the creation of the functions to be provided as a service can be implemented.

For the execution of the JPPF agents, we use containers and for each of the processing nodes that is instantiated or terminated, tasks are possible to be executed.

IV. Adaptable Wide Area Video Surveillance System Architecture

In the addition to the distributed computing paradigm, a novel architecture is proposed with a focus on the transparent use of fog and edge computing resources. With this new architecture, surveillance systems can cover more efficiently all sensitive areas in small towns and larger cities including public administration buildings, bus stations, shopping malls, main squares and any other sensitive area. The application logic of the new architecture is to gather video streams from the cameras, store and process video clips in the case of identified threats, for instance:

- Perimeter protection
- No trespassing of security areas
- Gunshots / Screaming

The primary detection of these types of security threats is based on a camera-built embedded system with computing and storage capacity. For more efficient handling and processing, the embedded systems are connected to Fog Nodes installed in central locations (e.g. building, shop malls, train stations etc.) that will provide additional computing and storage resources. The Fog Nodes can be connected to a public or private cloud, when needed, to make use of additional computing resources to cope with unpredicted workload at the Fog Node level and provide costumer-tailored business intelligence.

Control rooms in each one central location are equipped with video walls where security personnel can watch on real time the video streams coming from the cameras. In the case of an identified threat, an alarm is triggered and security personnel can notify the Police for response.

Therefore, we can distinguish the following processing layers, as follows:

- Layer-1: Camera-built embedded system

At this level, all camera-built embedded systems under the same or different domains will be synchronised based on localisation of data processing.

- Layer-2: Fog Node

At this level, all Fog Nodes will operate under a common monitoring service where computing and network resources will be shared amongst them for higher performance.

- Layer-3: Public or Private Cloud

At this level service migration between Fog Nodes and the cloud will be realised when Fog Nodes are not able to handle sudden picks in the workload.

An overview of the proposed technology for the new surveillance system and how the resources can be utilized by specific functions is depicted in Fig. 1. We can distinguish three major elements a) the camera-built embedded system, creating the Extreme Edge Layer, b) the Fog Nodes that create the Edge Layer, and c) the public or private cloud, creating the Cloud Deployment Layer.

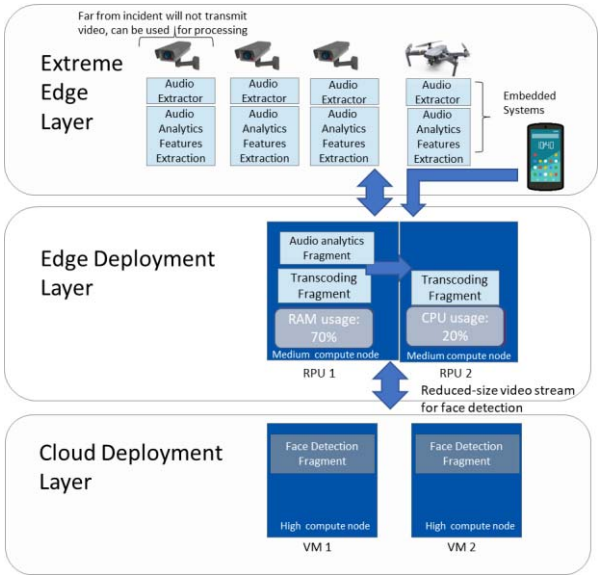

Fig. 1. Adaptable Survaillance system architecture.

A. *Distributed Task Execution and Orchestration*

In order to achieve the execution distributed computational tasks on edge devices the usage of JPPF framework has been selected. However, in order to accomplish the deployment and adaptation workflow that is needed, a complete orchestration mechanism that supports both edge and cloud resources has to be designed and implemented. For achieving the orchestration, the following architecture has been devised.

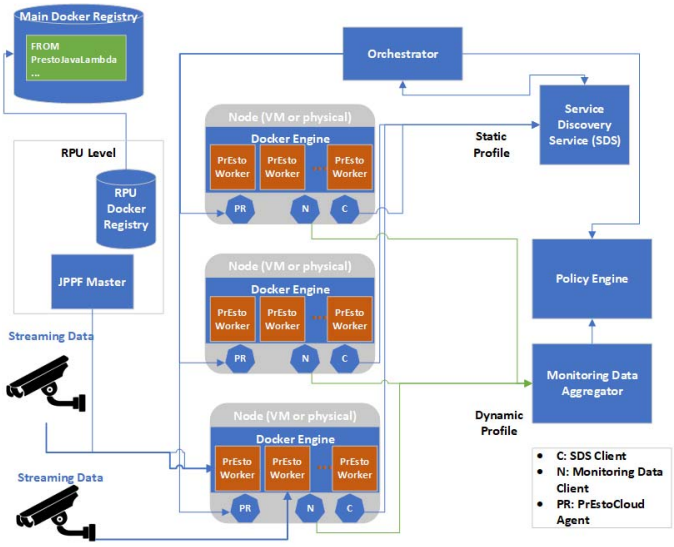

Fig. 2. Orchestrating components of an adaptable survaillance system architecture

For achieving the distributed processing of video stream, in each node of the system, either it is a Virtual Machine (VM) or a physical machine acting as surrogate, or an edge device, a container is deployed using one of the pre-existing container images. These images contain the JPPF based PrEsto Worker that includes the logic for the distributed processing for specific task, similarly with functions of the serverless computing paradigm.

To provide real time adaptation, the Orchestrator logic is supported by a monitoring mechanism that collects, aggregates and analyses monitoring data, and a policy engine that is responsible for enforcing adaptation and scaling rules when a policy is violated.

Finally, for the communication of all nodes that are in the edge and cloud on a Wide-Area surveillance system, it might be needed to have an inter-site network overlay that emulates this flat IP space from the application perspective. Overlay network techniques [24] allow the creation of a virtual network topology on top of a physical one. For the context of the surveillance system described in this paper, the overlay is materialized on each device by a specific network interface featuring an IP address in a dedicated (and large enough) private IP address subnet. The orchestrator described in the previous sections shall take care of adding specific routing rules and possibly dedicated machines to build the overlay.

V. Sample Resources for Implementing an Adaptable Wide Area Video and Audio Surveillane

The 1st Layer of the smart wide area surveillance system includes the Embedded System (ES) with the Camera. The ES has designed and developed in order to handle:

- Sensor management tasks. Connectivity and information retrieval from different cameras and microphones.
- Video processing in order to achieve unified video format.
- Lightweight video analytics (e.g. intrusion detection).
- Intelligent management of Lightweight analytics and embedded system operations (storage efficiency, network adaptation etc.).

The embedded system is capable of interconnecting various types of cameras (USB, Network cameras etc.). It is very important to mention that the embedded system has two modes: i) smart operation, and ii) traditional CCTV. As the 1st layer of the proposed system the embedded system has the capability of performing lightweight identification of various events. The ES can be built using low cost and small size computing devices. For our testbed, we used Raspberry Pi Model 3B single-board computer, that is cheap but powerful enough in order to execute distributed tasks. It uses a Quad Core 1.2GHz Broadcom BCM2837 64bit CPU that is combined with 1GB RAM. On the connectivity side, this device is provided with onboard fast Ethernet connection of 1 Gbps, WiFi and Bluetooth and four USB-2.0 ports that enables not only the use of USB based sensors (e.g. cameras) but also the utilization of USB WIFI dongles for connecting to wireless mesh networks. In our upcoming experiments we will utilize WiPi Mesh cards for the setup of mesh deployment of the

Embedded Systems in the field and utilize this mesh network for the communication of the computation nodes.

The ES can be utilized for running video and audio analytics based on end-user needs. The ES can provide a first notification of any abnormal detection at a local level. When any type of the aforementioned security threats is detected, then video is streamed back from the camera to the respective fog node for further identification and verification.

For the fog level, we have used small NUC computer that is deployed at each floor of the building and collects the input from the extreme edge layer. This standalone server could be easily replaced by Cloudlets deployed on the edge.

In order to achieve the optimal scalability, the designed system is supported by a public cloud. In our case it was OpenStack based cloud, but due to the usage of containerized environments on top of normal VMs, any Cloud provider is supported and also due to the Inter-site Network Virtualization, usage of multiple cloud providers is the same time is possible.

VI. Conclusions and Future Work

We presented in this paper a novel paradigm and architecture for surveillance systems based on edge and fog computing concept. The suggested parading combines the cyber foraging approach for offloading of computation tasks to machines in the vicinity, and the serverless execution model for the definition of function that can be executed on cloud, fog and edge devices. We defined the architectural and software components in three different processing layers named i) the embedded system, ii) the Fog node, and iii) the cloud. The final choice of the embedded system flavor is based on the end-user need and the application area.

In our future work, we plan to evaluate the suggested parading with trials for the whole system under a real environment where the performance of all different layers from the edge to fog and the cloud back-end will be evaluated with a number of test cases.

Acknowledgment

This work is partly funded by the European Union's Horizon 2020 research and innovation program under grant agreement No 732339. Project PrEstoCloud - Proactive Cloud Resources Management at the Edge for Efficient Real-Time Big Data Processing (http://prestocloud-project.eu).

References

[1] G. Kioumourtzis, M. Skitsas, N. Zotos and A. Sideris, "Wide area video surveillane based on edge and fog computing concept," 2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA), Larnaca, 2017, doi: 10.1109/IISA.2017.8316451

[2] A. M. M. Ali, N. M. Ahmad and A. H. M. Amin, "Cloudlet-based cyber foraging framework for distributed video surveillance provisioning," 2014 4th World Congress on Information and Communication Technologies (WICT 2014), Bandar Hilir, 2014, pp. 199-204. doi: 10.1109/WICT.2014.7076905

[3] Eric Jonas, Qifan Pu, Shivaram Venkataraman, Ion Stoica, and Benjamin Recht. 2017. "Occupy the cloud: distributed computing for the 99%". In Proceedings of the 2017 Symposium on Cloud Computing (SoCC '17). ACM, New York, NY, USA, 445-451.

[4] Alfonso Pérez, Germán Moltó, Miguel Caballer, Amanda Calatrava,Serverless computing for container-based architectures,Future Generation Computer Systems,Volume 83,2018,Pages 50-59,ISSN 0167-739X,https://doi.org/10.1016/j.future.2018.01.022. (http://www.sciencedirect.com/science/article/pii/S0167739X17316485)

[5] Burns, E., "Hadoop still too slow for real-time analysis applications" In TeckTarker. Available online at: http://searchbusinessanalytics.techtarget.com/feature/Hadoop-still-too-slow-for-real-time-analysis-applications, 2013.

[6] Cuomo, G., Martin, B.,K., Smith, K.,B., Ims, S., Rehn, H., Haberkorn, M., Parikh, J. (2013). Developing Edge Computing Applications. IBM White Paper, Avaialable online at: http://www.ibm.com/developerworks/cn/websphere/download/pdf/OnDemandEdgeComputing.pdf

[7] Cisco White Paper: Available online at: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf, 2015.

[8] Mims, M.,. "Forget 'the Cloud'; 'the Fog' Is Tech's Future". In The Wall Street Journal. Available online at: http://www.wsj.com/articles/SB10001424052702304908304579566662320279406, 2014.

[9] IDC (2014). FutureScape: Worldwide Internet of Things 2015 Predictions. In International Data Corporation web conference. Available online at: https://www.idc.com/getdoc.jsp?containerId=prUS25291514

[10] Cisco (2013) RFP-2013-078. Fog Computing, Ecosystem, Architecture and Applications. Available online at: http://research.cisco.com/research

[11] Perry, T., (2013)What Comes After the Cloud? How About the Fog? In IEEE Spectrum. Available online at: http://spectrum.ieee.org/tech-talk/computing/networks/what-comes-after-the-cloud-how-about-the-fog

[12] Gatto, M., E., (2014) New Solutions on the Horizon—"Fog" or "Edge" Computing? In the National Law Review. Available online at: http://www.natlawreview.com/article/new-solutions-horizon-fog-or-edge-computing

[13] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges," IEEE Personal Comm., vol. 8, no. 4, 2001, pp. 10–17.

[14] R. K. Balan and J. Flinn, "Cyber Foraging: Fifteen Years Later," in IEEE Pervasive Computing, vol. 16, no. 3, pp. 24-30, 2017. doi: 10.1109/MPRV.2017.2940972

[15] M. R. Ebling and R. Want, "Pervasive Computing Revisited," in IEEE Pervasive Computing, vol. 16, no. 3, pp. 17-19, 2017.

[16] Amazon, Amazon Lambda (AWS Lambda). https://aws.amazon.com/lambda/.(Online; accessed 20 May 2018).

[17] Google, Google Cloud Functions. https://cloud.google.com/functions/. (Online; accessed 20 May 2018).

[18] Microsoft. Microsoft Azure Functions. https://azure.microsoft.com/en-in/services/functions/. (Online; accessed 20 May 2018).

[19] OpenLambda. http://open-lambda.org/ .(Online; accessed 20 May 2018).

[20] The Apache Software Foundation. Apache Openwhisk. http://openwhisk.org/. (Online; accessed 20 May 2018).

[21] M. Villamizar et al., "Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures," 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), Cartagena, 2016, pp. 179-182. doi: 10.1109/CCGrid.2016.37

[22] Java Parallel Processing framework. Available online at: http://jppf.org

[23] A. Tsagkaropoulos, Y. Verginadis, D. Apostolou and G. Mentzas, "Seamless task offloading on multi-clouds and edge resources: An experiment," 2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA), Larnaca, 2017, pp. 1-5. doi: 10.1109/IISA.2017.8316411

[24] Cisco, Data Center Overlay Technologies, https : / / www . cisco . com / c / en / us / products / collateral /switches / nexus - 9000 - series - switches / white - paper - c11 - 730116 . html, [Online; accessed 20-May-2018], 2013