IoT Resource-aware Orchestration Framework for Edge Computing

Niket Agrawal
Delft University of Technology
Delft, The Netherlands

Jan Rellermeyer
Delft University of Technology
Delft, The Netherlands

Aaron Yi Ding
Delft University of Technology
Delft, The Netherlands

ABSTRACT

Existing edge computing solutions in the Internet of Things (IoT) domain operate with the control plane residing in the cloud and edge as a slave that executes the workload deployed by the cloud. The growing diversity in the IoT applications requires the edge to be able to run multiple distinct workloads corresponding to the dedicated inputs it receives, each catering to a specific task. Achieving this with the current approach poses a limitation as the cloud lacks the local knowledge at the edge and sharing this knowledge regularly between the edge and the cloud will defeat the very purpose of edge computing, ie, low latency, less network congestion and data privacy. To solve this problem, we propose an orchestration framework for edge computing that enables the edge to actively initiate and orchestrate the workloads on request by using the local knowledge available in the form of IoT resources at the edge.

1 BACKGROUND AND MOTIVATION

The state of the art edge computing architectures for the IoT deploy, monitor and manage the applications from a central control plane in the cloud [7]. Applications are offloaded in form of lightweight packages on the edge nodes as per a specific predefined configuration supplied by the control plane. While this approach suffices for scenarios offering a particular service at the edge, it fails to address cases involving execution of multiple distinct workloads triggered by the IoT device layer which is not in direct contact with the cloud. We further highlight this through an example which also forms our target use case for this work. Vehicles contact the roadside infrastructure embedded with edge computing resources to request a specific information [3, 5], for example, road condition in the neighborhood. The type of requests may vary per vehicle or any client in general and so will the corresponding applications to be run on the edge to process those requests[2, 4]. Given the recent trends in connected autonomous driving, the data feed for these applications is generated by vehicles in vicinity which upload high definition maps and images to the nearby roadside infrastructure [3] where it is collectively processed and generated information is shared with other vehicles. This mobility of the IoT resource makes it challenging for the state of the art to fetch information regarding it's availability on the edge in order to allocate tasks adequately to serve on demand requests. It is also not possible to efficiently utilize intermediate results produced on the edge for serving future requests in this manner as it would require multiple round trips between the cloud and the edge for these details to be shared with cloud. Moreover, hosting fixed applications on the edge for requests that are initiated seldom, will result in those applications being idle often, consuming the already scarce computing resources on the edge. We identify this gap and propose an orchestration framework that is completely driven by edge. It is capable of utilizing the knowledge available in terms of IoT resource availability through collaboration of edge nodes and offload tasks dynamically to serve multiple on demand requests. [6] shares a similar idea as this work but lacks to provide details on how the edge nodes will interact and orchestrate tasks. Our proposed architecture can also enable the integration of Complex Event Processing (CEP) Engine with Edge computing[1]. CEP provides interesting patterns and can identify critical events by fusing data from multiple IoT devices. Edge being in the vicinity of the IoT devices and the CEP can initiate the deployment of emergency services and operate collaboratively to take necessary actions.

Our primary goal in this work concerns with exploring the feasibility of an edge driven IoT resource-aware orchestration architecture for practical IoT use cases. We aim to tackle two major research questions:

(1) *How to orchestrate workloads dynamically by utilizing the local knowledge available at the edge in form of IoT resources?*
(2) *What is the overhead due to this edge driven approach and feasibility for practical use cases?*

Our contributions are summarized as follows:

(1) Design an orchestration framework as described in Section 2 to utilize the knowledge available in form of IoT resources.
(2) Translation of the IoT resource awareness into orchestration operations and commands that can be performed by edge computing engines.

2 APPROACH

In Figure 1 we compare our proposed system architecture with the state of the art in order to exhibit the benefits our system can provide. Through this work we take the first step towards constructing and operating the edge computing infrastructure in bottom-up fashion rather than the conventional top-down strategy in the state of the art that overlooks the IoT device layer and the context information at the edge. The orchestrator is implemented as a software pipeline in the edge that is responsible for carrying out a series of tasks for each request that the edge receives. The description of the software components that make up this pipeline is as follows.

Library - It holds the information about the application packages corresponding to a particular request in form of key-value pairs with a key (request) mapping to one or more application packages that we assume are made available by a knowledgeable party.

Parser - It parses the inputs and fetches information about the corresponding application packages using the configuration library.

Frontend - The communication interface which enables reception of client requests, IoT resource and and sending of resource updates to other edge nodes.

Resource Manager - It maintains a catalog of the IoT resource availability at each node and is responsible for communicating and

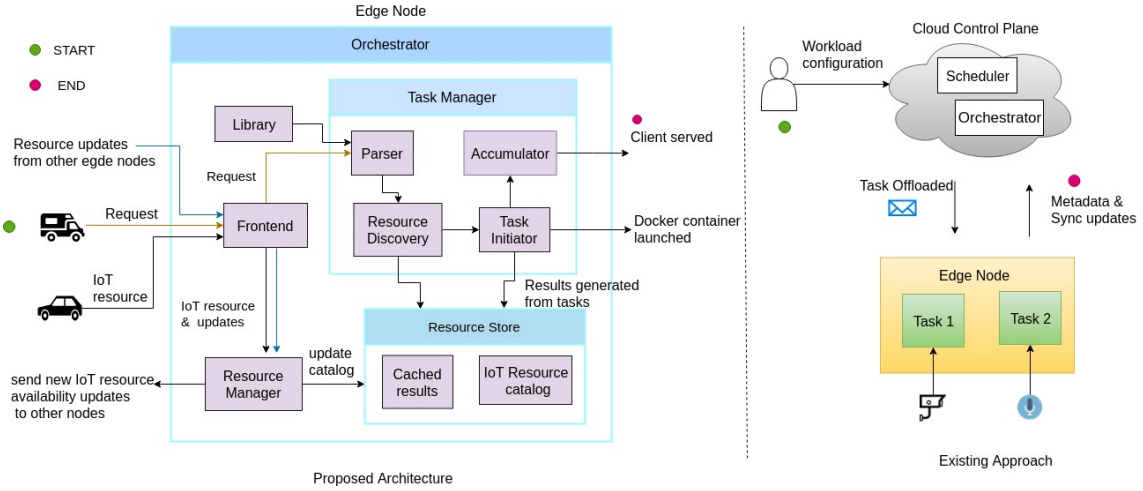

Figure 1: Proposed system Architecture vs Existing approach

receiving updates regarding the same with other edge nodes each time an IoT resource is uploaded on an edge node.

Resource Discovery - It determines the availability of the IoT resource needed by a workload and its location on the edge cluster via the resource manager. It also monitors any updates to the resource itself and notifies the availability of a newer version to the task manager to take appropriate actions, for example, restarting task with new version as input. It further communicates with the underlying container orchestration engine to fetch information regarding the workloads running on the cluster to identify if results from them can be utilized for future workloads.

Task Initiator - Translates the information from the above module into commands for the underlying container orchestration engine to launch the containerized tasks.

Accumulator - It accumulates results from all the application packages executed for a particular request and marks the end of the processing pipeline. The results can be either be shared with the client directly using appropriate wireless connectivity means or with the cloud in case further processing is desired, the details of which do not form the scope of this work.

To realize this framework, we foresee the following challenge.

The above modules work concurrently and in a distributed fashion, ie, while the edge nodes receives a new request and runs it through the pipeline, the IoT resource from a vehicle is being received and associated updates are being shared across the edge nodes at the same time. Our system lacks a leader node and any node can read and write to the distributed data store. The rationale behind this decision comes from the point that any node can ideally receive an IoT resource or a client request and hence the need for a flat hierarchy. Absence of a leader will lead to potential conflict, synchronization and consensus issues and hence the overhead that comes from the frequent inter node communication to solve them.

3 PRELIMINARY RESULTS

We plan to evaluate the system based on the overhead it generates to service each client request with increasing number of applications and requests. Implementation of a proof of concept using Docker Swarm as baseline is in progress. Docker only orchestrates the workloads based on computing resource availability that are predefined in a configuration file . We extend this with our edge processing pipeline to also make it IoT resource-aware. To this point, we are able to run the framework distributed on two separate instances on a single host emulating two edge nodes. There is no noticeable delay in resource management and resource discovery across the two instances. Implementation of a communication link with Docker to launch containerized tasks and access snapshots is in progress. The framework will be fully evaluated on physical nodes like Intel NUC and Raspberry Pi.

4 CONCLUSION

In this work we address the need for and develop an edge driven orchestration framework for edge computing to handle evolving use cases in the IoT domain. These use cases involve IoT context driven services, mobile IoT resources and on demand client requests which are infeasible to support using the existing cloud driven approach. We aim to study the feasibility of such an architecture for practical use cases and reflect on the trade-off it offers.

REFERENCES

[1] T. Sung P. Wang E. Jou C. Y. Chen, J. H. Fu and M. Feng. 2014. Complex event processing for the Internet of Things and its applications. *IEEE International Conference on Automation Science and Engineering (CASE)* (2014), 1144–1149.

[2] N. Zhang X. Yang H. Zhang W. Zhao J. Lin, W. Yu. 2018. A survey on Internet of things: Architecture enabling technologies security and privacy and applications. *IEEE Access* 6 (2018), 6900–6919.

[3] T. Braud P. Hui P. Zhou, W. Zhang and J. Kangasharju. 2018. ARVE: Augmented Reality Applications in Vehicle to Edge Networks. *Proceedings of the 2018 Workshop on Mobile Edge Communications, New York, NY, USA* (2018), 25–30.

[4] J. Pan and J. McElhannon. 2018. Future Edge Cloud and Edge Computing for Internet of Things Applications. *IEEE Internet of Things Journal* 5, 1 (2018), 439–449.

[5] A. Y. Ding V. Cozzolino and J. Ott. 2019. Edge Chaining Framework for Black Ice Road Fingerprinting. *Proceedings of the 2Nd International Workshop on Edge Systems, Analytics and Networking, EdgeSys '19, Dresden, Germany* (2019), 42–47.

[6] S. Zhang Y. Sahni, J. Cao and L. Yang. 2017. Edge Mesh: A New Paradigm to Enable Distributed Intelligence in Internet of Things. *IEEE Access* 5 (2017), 1028–1031.

[7] L. Xing Y. Xiong, Y. Sun and Y. Huang. 2018. Extend Cloud to Edge with KubeEdge. *2018 IEEE/ACM Symposium on Edge Computing (SEC)* (2018), 373–377.