RaDiCS: Distributed Computing Service over Raspberry Pis with Unikernels

Keith Collister
University of Cambridge
kc506@cam.ac.uk

Eiko Yoneki
University of Cambridge
eiko.yoneki@cl.cam.ac.uk

CCS CONCEPTS

• **Software and its engineering** → *Software system structures*; • **Computer systems organization** → *Distributed architectures*; • **Networks** → *Network architectures*;

1 INTRODUCTION

Cloud computing is currently performed mostly by large server farms: these are expensive to build, operate, and maintain. Additionally, these centres of computation may be far away from where data is sourced from, increasing the latency of processing due to transportation overheads. We present RaDiCS, distributed data processing over Raspberry Pi based cluster computing environment, where Unikernels [2] are exploited to improve data processing performance. RaDiCS builds a framework for decentralising computation by using many smaller nodes spread out over a large area. Client programs can then run on these nodes, performing computation and transmitting data to other connected nodes. Processes can be migrated to connected nodes in order to balance demand on the system, or move a data processor closer to the source of its data, to reduce latency.

2 USE CASES

There will be many possible use cases. One of our of target applications is an image classification based on machine learning. We have built a fast, lightweight, and fully parallelisable convolutional neural network library (PiCNN) designed for the Raspberry Pi, which improves its limited computational capabilities compared to conventional desktop computers. Together with compact Unikernels, it enables practical machine learning in RaspberryPi programming platforms, without the hassle, size, and computational requirements of installing and running larger machine learning frameworks. The detail of PiCNN is out of scope of this paper. In general, RaDiCS supports distributed computing functions, such as MapReduce or web servers' load balancing, by migrating requests.

3 ARCHITECTURE

We use Rumprun [1] as our Unikernel framework, which allows for standard C++ code to be written, compiled, and baked into a small unikernel image (usually around 4MB) that can be executed on any x86 platform (or emulator). It also has support for other languages, such as Java, Python, and Go. RaDiCS uses Raspberry Pis as physical nodes, and runs client programs (virtual nodes) in virtual machines to enforce isolation and allow for easier process migration. An implementation of the Delay Tolerant Networking Bundle Protocol (RFC 5050) is used to communicate between physical nodes. Client programs take the form of Unikernels, extremely small operating system kernels designed to perform a single task. The client code interfaces with an API provided by RaDiCS that handles both communication with other processes and the management of this process. The client code and the RaDiCS libraries are then put through a build process that creates the unikernel, which is extremely fast to boot and is then run as a virtual node. The RaDiCS framework itself is made of two layers (Figure 1):

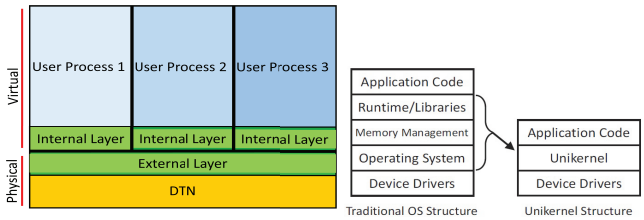

Figure 1: RaDiCS Node Structure

- Internal layer with which the client process interacts. This is compiled into the unikernel as well, and provides services to the client process by communicating with the external layer.
- External layer which interfaces with the internal layer, the DTN implementation, and the virtual machine process to provide useful functionality.

This two-layer setup allows the client program to transparently get access to features including:

- Communication between virtual nodes on the same or different physical nodes.
- Saving a virtual node to a disk to be resumed later.
- Migrating a running virtual node to a different physical node (e.g. to place it closer to a data source).

ACKNOWLEDGMENTS

This research is part-funded by the EPSRC (EP/P004024/1) and RAEng FoESF1718/4/3.

REFERENCES

[1] Kantee, A., and Cormack, J. Rump kernels: No os? no problem!
[2] Madhavapeddy, A., and Scott, D. Unikernels: Rise of the virtual library operating system.

MobiSys '18, June 10–15, 2018, Munich, Germany

ACM ISBN 978-1-4503-5720-3/18/06.
https://doi.org/10.1145/3210240.3210824