Next Generation Single Board Clusters

Jeremy Singer[*¶], Herry Herry[*], Philip J. Basford[†], Wajdi Hajji[‡], Colin S. Perkins[*], Fung Po Tso[‡], Dimitrios Pezaros[*], Robert D. Mullins[§], Eiko Yoneki[§], Simon J. Cox[†], Steven J. Johnston[†],

[*]School of Computing Science, University of Glasgow, Glasgow, G12 8QQ, United Kingdom
[†]Faculty of Engineering and the Environment, University of Southampton, Southampton, SO16 7QF, United Kingdom
[‡]Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, United Kingdom
[§]Computer Laboratory, University of Cambridge, Cambridge, CB3 0FD, United Kingdom
[¶]jeremy.singer@glasgow.ac.uk

Abstract—**Until recently, cluster computing was too expensive and too complex for commodity users. However the phenomenal popularity of single board computers like the Raspberry Pi has caused the emergence of the single board computer cluster. This demonstration will present a cheap, practical and portable Raspberry Pi cluster called Pi Stack. We will show pragmatic custom solutions to hardware issues, such as power distribution, and software issues, such as remote updating. We also sketch potential use cases for Pi Stack and other commodity single board computer cluster architectures.**

I. Introduction

15 million Raspberry Pi single board computer (SBC) devices have been sold to date[1]. The pressing question is: how are people deploying such SBCs? One appealing answer is to network several boards together to produce a micro-cluster environment. This is highly feasible, since each SBC runs a full Linux stack and has conventional ethernet interconnect, effectively enabling the creation of a Beowulf cluster.

The key problem is that per-node resource availability is highly constrained, in terms of processing capability, RAM, storage capacity and network bandwidth. For this reason, conventional cluster workloads do not perform particularly well on SBC-based micro-clusters. However, a number of compelling use cases have emerged for this new technology, see Section IV.

Our project involves building federated clusters of single board computer (SBC) nodes. During our work, we have encountered a number of practical challenges in terms of both hardware (Section II) and software (Section III). This demo (described in Section V) will identify these challenges and showcase our working solutions.

II. Pi Stack

Creating a cluster of Single Board Computers (SBCs) poses challenges around mounting and cable management. The extremely small form factor of the SBC means requires novel mounting mechanisms and the density available means many more cables are present when compared to a server cluster. The approach used by Iridis-Pi [1] was to use Lego to create a rack to hold the SBCs, in this case Raspberry Pi version 1. Whilst

the approach allowed the first Pi cluster to be produced and benchmarked it used separate USB power supplies for each Pi meaning fine grained power consumption data is not available and the power distribution hardware was bulkier than the main cluster. Since then there have been other approaches taken to producing Pi Clusters such as using Power-Over-Ethernet[2], or 12volt DC power distribution and local power supplies[3].

An alternative solution to either of these is the Pi Stack [2] shown in Figure 1a. This board sits between 2 Raspberry Pi SBC (or any other SBC board using the same B+ footprint[4]) and provides individual power control and monitoring for each SBC. The boards use the metal stand-offs that form the mechanical mounting to carry the power and control signals within the stack of boards. Each Pi Stack board is given a unique ID using DIP switches mounted on the board. This allows each board to be addressed within the bus, there is a theoretical maximum of 16 Pi-Stack boards per stack, however, practical considerations may mean that this cannot be reached. Multiple stacks can be combined into a single cluster. In order to reduce the current flowing through the stand-offs each Pi Stack board has a local 5volt power supply for the connected Pis. Whilst the 2 Pis on a single Pi Stack board share a 5v PSU, each pi can be power controlled individually, as well as having individual voltage and current monitoring. The Pi-Stack also uses GPIO lines to communicate with the Pi. One GPIO is used to enable either device to indicate a shut-down, this means the Pi Stack can turn the power off once the Pi has shut-down, or the Pi can initiate a clean shut-down when notified that a power off is imminent. A second pin is used as a heartbeat pin, enabling the Pi Stack to detect if the Pi has crashed and to trigger a power cycle.
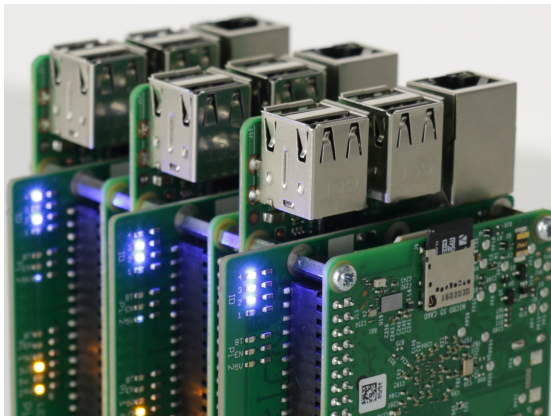
III. Managing Federated Pi Stack Clusters

Most SBCs are capable of running a range of mainstream OS variants. However, there are some challenges on making this OS to be more robust on hardware problems. One of them is that SD card, a common storage for SBC, could be easily corrupted. A read-only root partition with ramdisk overlays

---

[1]https://betanews.com/2017/07/19/raspberry-pi-eben-upton-qa/

[2]https://blog.mythic-beasts.com/wp-content/uploads/2017/03/raspberry-pi-cloud-final.pdf
[3]https://resin.io/blog/good-better-beast-week-2/
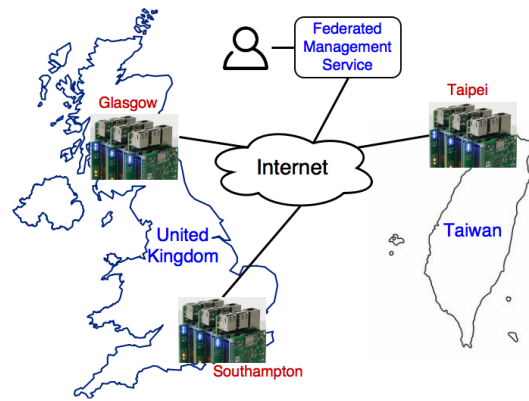[4]https://www.raspberrypi.org/blog/introducing-raspberry-pi-model-b-plus/

(a) Pi Stack cluster shown without network cables.

(b) Federated Pi Stack clusters.

Fig. 1: Architecture of the demo system.

[3] can mitigate this. Alternatively, SBCs can be set to use network booting enabling them to be diskless nodes, although this requires more complex setup.

Linux, as the most popular SBC OS, is fragmented because it requires specific patches that are only available on vendor repositories. Moreover, popular distributions such as Raspbian are bloated with unnecessary software making them fragile to security attacks as well as increasing the size and complexity of updates for a large cluster. These issues could be addressed using a customized OS generator [4] to build a complete, lean Linux image targetting multiple hardwares. Also, applying over-the-air update techniques [5] can shorten the deployment times and support for rollback.

SBCs are commonly used to build sensor and control systems, and to provide smart edge compute nodes. Such systems are usually deployed in inaccessible locations, be mobile, or be in private residence. These raise a new challenge for updating the system since they only have limited network access due to the presence of firewalls, NATs, or intermittent connectivity. This limited network connectivity issue requires management tools to evolve to allow node management via indirect, peer-to-peer connections that traverse firewalls and NATs. A peer-to-peer approach, as suggested in [6], could be a solution of this problem. It also eliminates the need of a central server, increasing the robustness and scalability of the system.

Typical application frameworks can run on SBCs, e.g. Hadoop and Spark. However their performance is limited, due to the per-node resource constraints [7]. More promising lightweight frameworks include Apache Edgent and Tensorflow Lite, which are specifically targeted for edge environments.

IV. SINGLE BOARD CLUSTER APPLICATIONS

SBCs have inspired researchers to build novel hardware configurations for a range of uses cases such as *edge compute* [8], [9] *expendable comput*e [10], [11], and *portable clusters* which are difficult to implement using traditional hardware.

On the other hand, *educational* micro-scale datacenters built from SBCs [1], [12] have helped students to learn how to deal with a software/hardware stack that is similar (but not identical) to real commercial large-scale datacenters. It is likely that next-generation datacenters will feature ARM processing cores. Since the majority of SBCs are Arm-based, a micro-cluster can be an ideal environment for *prototyping future datacenter* applications[5].

V. DEMONSTRATION

We propose an integrated hardware/software demo of a federated Pi Stack clusters illustrated in figure 1b, running our custom peer-to-peer secure update and management infrastructure. In this interactive demo, we will: 1) setup bare metal Pi Stack hardware with FruitOS[6] to join our federated management infrastructure; 2) use the peer-to-peer secure update to run Kubernetes [13] system on the Pi Stack cluster; and 3) deploy containerised applications on Kubernetes to prove that the updates are applied properly.

This demo will serve to highlight the portability and flexibility of the Pi Stack as a platform for *edge compute* and *portable clusters*. This also demonstrates that clusters of edge devices can be managed in the presence of NATs and firewalls, where there is not necessarily direct access from a management node to the devices being updated.

REFERENCES

[1] S. J. Cox, J. T. Cox, R. P. Boardman, S. J. Johnston, M. Scott, and N. S. O 'brien, "Iridis-pi: a low-cost, compact demonstration cluster," *Cluster Computing*, 2013.
[2] P. Basford and S. Johnston, "Pi Stack PCB," 2017. [Online]. Available: https://eprints.soton.ac.uk/416906/
[3] "Resin os," https://resinos.io, 2017.
[4] "Yocto project," https://yoctoproject.org, 2017, Accessed: 2017-12-19.

[5]http://zd.net/2zQmQow

[6]https://github.com/fruit-testbed/fruitos

[5] "Mender," http://mender.io, 2017, Accessed: 2017-12-19.

[6] E. Band, H. Herry, C. Perkins, and J. Singer, "Peer-to-peer secure update for heterogeneous edge devices," 2018, submitted to DOMINO Workshop.

[7] W. Hajji and F. P. Tso, "Understanding the performance of low power Raspberry Pi Cloud for big data," *Electronics*, vol. 5, no. 2, p. 29, 2016.

[8] C. Pahl, S. Helmer, L. Miori, J. Sanin, and B. Lee, "A container-based edge cloud paas architecture based on raspberry pi clusters," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. IEEE, Aug. 2016, pp. 117–124.

[9] S. Helmer, S. Salam, A. M. Sharear, A. Ventura, P. Abrahamsson, T. D. Oyetoyan, C. Pahl, J. Sanin, L. Miori, S. Brocanelli, F. Cardano, D. Gadler, D. Morandini, and A. Piccoli, "Bringing the Cloud to Rural and Remote Areas via Cloudlets," in *Proceedings of the 7th Annual Symposium on Computing for Development - ACM DEV '16*. New York, New York, USA: ACM Press, 2016, pp. 1–10.

[10] E. Yoneki, "Demo: RasPiNET: decentralised communication and sensing platform with satellite connectivity," in *Proceedings of the 9th ACM MobiCom workshop on Challenged networks - CHANTS '14*. New York, New York, USA: ACM Press, 2014, pp. 81–84.

[11] D. Moure, P. Torres, B. Casas, D. Toma, M. Blanco, J. Del Río, and A. Manuel, "Use of Low-Cost Acquisition Systems with an Embedded Linux Device for Volcanic Monitoring," *Sensors*, vol. 15, no. 8, pp. 20 436–20 462, Aug. 2015.

[12] F. P. Tso, D. R. White, S. Jouet, J. Singer, and D. P. Pezaros, "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures," in *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*. IEEE, jul 2013, pp. 108–112.

[13] "Kubernetes: Production-Grade Container Orchestration," https://kubernetes.io, 2018, Accessed: 2018-02-16.