LESSONS LEARNED FROM IMPLEMENTING A SCALABLE PAAS SERVICE BY USING SINGLE BOARD COMPUTERS

Christian Baun

Faculty of Computer Science and Engineering, Frankfurt University of Applied Sciences, Nibelungenplatz 1, 60318 Frankfurt am Main, Germany

ABSTRACT

When a Platform-as-a-Service is demanded and the cost for purchase and operation of servers, workstations or personal computers is a challenge, single board computers may be an option to build an inexpensive system. This paper describes the lessons learned from deploying the private cloud PaaS solution AppScale on single-node systems and clusters of single board computers.

KEYWORDS

Single Board Computers, Platform-as-a-Service, AppScale

1. INTRODUCTION

A Platform-as-a-Service (PaaS) implements a scalable application runtime environment for programing languages. Such a service allows scaling from a single service instance to many, depending on the actual demand [1]. Prominent instances of public cloud PaaS offerings are Google App Engine (see Figure 1), Microsoft Azure Platform, AWS Elastic Beanstalk and Engine Yard. In some cases, it might be desirable to avoid public cloud offerings for privacy or legal reasons for example. Advantageously, private cloud solutions exist. Examples are AppScale [2][3], Apache Stratos and OpenShift.

This work focuses on AppScale, because it is an open source re-implementation of the Google App Engine. This means in effect, that App Engine compatible applications can be executed with AppScale, which itself can be deployed inside an Infrastructure-as-a-Service, or inside a physical or virtual machine. AppScale is a unique selling point for the App Engine, because the existence of a compatible and free private cloud solution avoids a lock-in situation, which is a benefit from customer perspective.

Besides AppScale, another software solution, called TyphoonAE [4] exists, which re-implements the Google App Engine and aims to be API compatible. The project has not seen any further development since 2011 and the latest revision does not provide any cluster operation mode. For these reasons, TyphoonAE was not further investigated for this work.

AppScale implements the App Engine API by using the development server, which is a part of the App Engine SDK. It has support for Go, Java, PHP, and Python applications. Scalability is archived by using different open source components to implement the required infrastructure services. Some examples are the web server software nginx, the load balancer software HAProxy, the Python imaging library, the XMPP application server ejabberd and Apache Cassandra and Apache ZooKeeper, which are used to implement a persistent datastore and a blobstore storage.

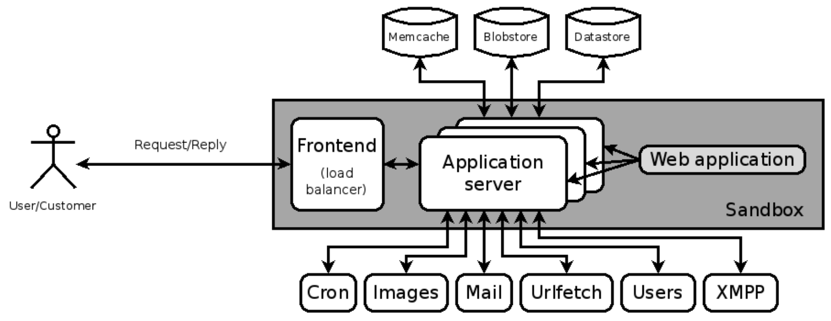


Figure 1. Web applications in the Google App Engine can use different infrastructure and storage services. AppScale needs to emulate theses services by using third-party software

For research projects, educational purposes and in environments with limited space and/or energy resources, single board computers may be a useful alternative to commodity hardware servers, because they require less purchase costs and operating costs. The drawback of single board computers are the limited resources, which cannot compete with the performance of higher-value systems [5]. One option to handle this issue is by building cluster systems of multiple single board computers [6].

Table 1 presents an overview about the hardware characteristics of the single board computers used for this work. Because main memory and compute power are critical factors, when running a PaaS, only single board computers with at least four CPU cores and not less than 1 GB of main memory were considered useful. Further devices, which meet these criteria, are the Banana Pi M2 and the Banana Pi M64. They have similar hardware characteristics compared with the Raspberry Pi 2/3 and the ODROID-U3. Because no fundamentally different results can be expected from these further devices, they have not been investigated in this work.

Table 1. The Single Board Computers, which were used to deploy AppScale

	ODROID-U3	**Raspberry Pi 2 B**	**Raspberry Pi 3 B**
CPU family	ARM Cortex A9	ARM Cortex A7	ARM Cortex A8
CPU cores	4	4	4
Clock rate	1700 MHz	900 MHz	1200 MHz
Main memory	2048 MB	1024 MB	1024 MB
Ethernet interface	10/100 Mbit	10/100 Mbit	10/100 Mbit
Storage interfaces	microSD	microSD	microSD

This paper is organized as follows. Section 2 contains a discussion of related work. In Section 3, single-node scenarios are explained. Section 4 presents two cluster scenarios with Raspbery Pi 2 and ODROID-U3 devices as nodes. In Section 5, the usability of the investigated scenarios is analyzed. Section 6 contains a calculation of the acquisition costs and Section 7 presents an analysis of the energy-efficiency of the investigated scenarios. Finally, Section 8 presents conclusions and directions for future work.

2. RELATED WORK

In the literature, several works propose using single board computers to build cluster systems.

Cox et al. [7] assembled in 2012 at the University of Southampton a cluster of 64 Raspberry Pi nodes with 256 MB main memory per node. The nodes were powered by using 64 individual 5 V power supplies. The power consumption of the cluster was not presented.

Balakrishnan [8] constructed in 2012 at the University of Edinburgh a cluster by using six PandaBoard single board computers and two Raspberry Pi nodes. The work provides the power consumption of the cluster, which is around 170 W in idle state and around 200 W during peak load.

Kiepert [9] assembled in 2013 at the Boise State University a cluster of 32 Raspberry Pi nodes with 512 MB main memory per node. To power the nodes, he used two standard PC power supplies, to power the nodes. The maximum total power usage of the cluster is 167 W.

Abrahamsson et al. [10] presented in 2013 a cluster, called MegaRPi, which was assembled at the Free University of Bozen-Bolzano. The cluster consists of 300 Raspberry Pi nodes with 512 MB main memory per node. To power the cluster, standard PC power supplies were used. The work identified several challenges and a number of opportunities. Unfortunately, no further power measurements were presented.

Sukaridhoto et al. [11] presented in 2013 a cluster of 16 Pandaboard nodes, which was assembled at the Electronics Engineering Polytechnics Institute Surabaya. The cluster used a single 200 W, 5 V, 40 A power supply to power the nodes. The power consumption of the entire cluster was not presented.

Ou et al. [12] compared in 2012 the performance, energy-efficiency and cost-efficiency of a single PandaBoard computer with an Intel x86 workstation for the three applications web server throughput, in-memory database and video transcoding.

Tso et al. [13] presented in 2013 the Raspberry Pi cloud, which was assembled at the University of Glasgow. This cluster is a scale model of a data center, composed of 56 Raspberry Pi Model B nodes, that emulates the layers of a cloud stack. The work compares the acquisition cost, electricity costs (196 W) and cooling requirements of the cluster of single board computers with a testbed of 56 commodity hardware servers.

Pfalzgraf and Driscoll [14] assembled in 2014 at the Bradley University a cluster of 25 Raspberry Pi nodes and used a single 600 W PC power supply to power the cluster nodes. This work does not provide any power measurements results.

These works show the potential of clusters of single board computers, but none of them evaluates the usability of single board computers for deploying a private cloud PaaS like AppScale.

3. SINGLE-NODE SCENARIOS

For testing and educational purposes, the single-node deployment of a PaaS is sufficient. For this reason, deployment attempts of Appscale have been carried out on different single board computers.

3.1. Raspberry Pi 2/3

The developers of AppScale only support a limited number of operating systems. The tested AppScale version 3.1 is designed to run per default on Ubuntu 14.04 LTS (Trusty Tahr), Ubuntu 12.04 LTS (Precise Pangolin) and Debian 7 (Wheezy) [15]. Because of unmet package dependencies, the installation of Ubuntu 16.04 LTS (Xenial Xerus) and Debian 8 (Jessie) failed.

This made it impossible to deploy AppScale on the Raspberry Pi 3, because no operating system image of Ubuntu 14.04, 12.04 or Debian 7 was available for this single board computer. Because of the modest CPU performance gain of the Raspberry Pi 3 compared with its predecessor and due to the fact, that it provides also just one GB of main memory, the Raspberry Pi 3 was not considered further for this work.

The Raspberry Pi 2 used, was installed with an Ubuntu 14.04 image [16]. The biggest challenge when installing AppScale on a single Raspberry Pi 2 is the size of the main memory. The AppScale developers recommend for each node at least 4 GB of main memory [15]. Per default, the database system Apache Cassandra, used by AppScale, tries to occupy more main memory than available in hardware with the Raspberry Pi. Therefore it is mandatory to reduce the main memory utilization of Cassandra by assigning the value 200 MB to the variable `MAX_HEAP_SIZE` inside the configuration file `cassandra-env.sh`.

On Raspberry Pi computers, a part of the main memory is assigned to the graphics processing unit (GPU). A pure server does not need a GPU at all but the minimum value is 16 MB, which is more useful compared to the dafault value (64 MB). In order to free 48 MB of main memory, a new line "`gpu_mem=16`" was inserted inside the file `/boot/config.txt`. To free some additional main memory, the number of active virtual consoles was reduced from value six to value two inside the file `/etc/default/console-setup`. Furthermore, no graphical user interface was started. These steps resulted in 930 MB free main memory (see Figure 2).

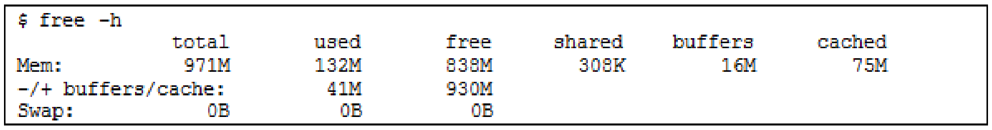

```
$ free -h
             total       used       free     shared    buffers     cached
Mem:          971M       132M       838M       308K        16M        75M
-/+ buffers/cache:        41M       930M
Swap:           0B         0B         0B
```

Figure 2. Main memory utilization of the Raspberry Pi 2 shortly after the operating system has been started

After AppScale has been started, the main memory is almost entirely utilized (see Figure 3).

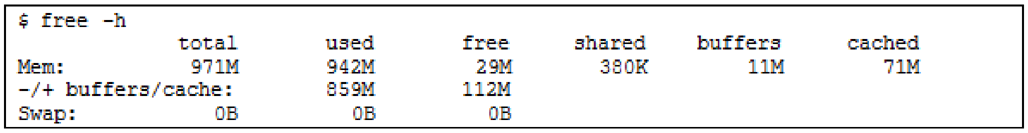

```
$ free -h
             total       used       free     shared    buffers     cached
Mem:          971M       942M        29M       380K        11M        71M
-/+ buffers/cache:       859M       112M
Swap:           0B         0B         0B
```

Figure 3. Main memory utilization of the Raspberry Pi 2 shortly after AppScale has been started

Because of this level of main memory utilization, the PaaS is not usable and it is even impossible to deploy one of the sample applications [17].

3.2. ODROID-U3

The ODROID single board computer was installed with a Debian 7 image [18]. Just like with the Raspberry Pi 2, the number of active virtual consoles was reduced inside the file `/etc/inittab` to free some additional main memory and no graphical user interface was started. Figure 4 presents the main memory utilization before AppScale is started.

```
# free -h
             total       used       free     shared    buffers     cached
Mem:          2.0G       122M       1.9G         0B        53M        32M
-/+ buffers/cache:        35M       1.9G
Swap:           0B         0B         0B
```

Figure 4. Main memory utilization of the ODROID-U3 shortly after the operating system has been started

The main memory utilization after AppScale is started is shown in Figure 5. The PaaS was tested with several sample applications [17] and turned out to be usable for testing and educational purposes.

```
# free -h
             total       used       free     shared    buffers     cached
Mem:          2.0G       1.9G       120M         0B        58M       109M
-/+ buffers/cache:       1.7G       288M
Swap:           0B         0B         0B
```

Figure 5. Main memory utilization of the ODROID-U3 shortly after AppScale has been started

4. Cluster Scenarios

A possible option to cope with the main memory problems, when using a single Raspberry Pi 2 or an ODROID-U3, is to use more nodes to build up cluster systems. A further benefit of a cluster system is the potential to provide a better level of reliability. In order to construct cluster systems, the micro SD cards used by the single computers need to be duplicated and the hostnames and IP addresses in the configuration files `/etc/network/interfaces` and `/etc/hostname` modified.

4.1. Raspberry Pi 2

For this work, a cluster of four Raspberry Pi 2 has been constructed (see Figure 6). In the `AppScalefile`, which is used to specify the configuration of the PaaS, the roles (AppScale components), are assigned to the single nodes. The used configuration is presented in Figure 7. This configuration distributes the load, caused by the AppScale components to the available nodes, but it cannot provide any failure resiliency, because no backup services are available.

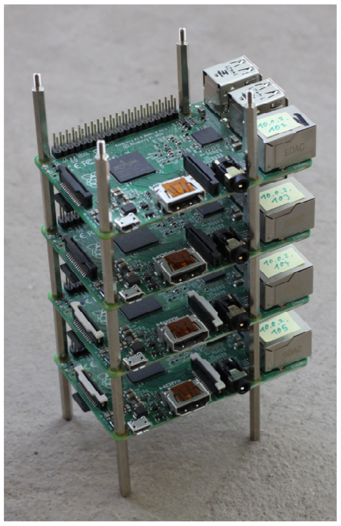
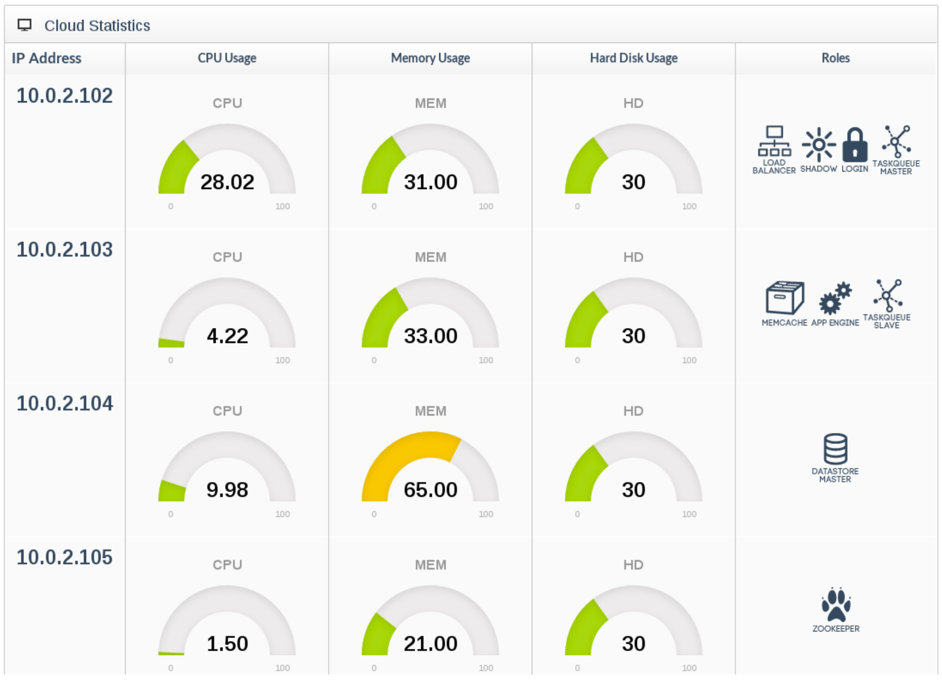


Figure 6. The Cluster of four Raspberry Pi 2 nodes

```
ips_layout :
 master : 10.0.2.102
 appengine : 10.0.2.103
 database : 10.0.2.104
 zookeeper : 10.0.2.105
```

Figure 7. Assignment of roles to the nodes of the Raspberry Pi cluster in the AppScalefile

The first node operates as master and hosts among others the load balancer. The second node hosts among others the App Engine and the Memcache services. The remaining nodes are dedicated nodes for Apache Cassandra and Apache Zookeeper. The node which hosts Cassandra has the highest level of main memory utilization (see Figure 8).



Cloud Statistics

IP Address	CPU Usage	Memory Usage	Hard Disk Usage	Roles
10.0.2.102	CPU 28.02	MEM 31.00	HD 30	LOAD BALANCER, SHADOW, LOGIN, TASKQUEUE MASTER
10.0.2.103	CPU 4.22	MEM 33.00	HD 30	MEMCACHE, APP ENGINE, TASKQUEUE SLAVE
10.0.2.104	CPU 9.98	MEM 65.00	HD 30	DATASTORE MASTER
10.0.2.105	CPU 1.50	MEM 21.00	HD 30	ZOOKEEPER

Figure 8. Resource utilization of the Raspberry Pi cluster, shown in the AppScale web user interface, shortly after AppScale has been started

4.2. ODROID-U3

The second cluster, which has been constructed for this work, consists of eight ODROID-U3 nodes (see Figure 9). The assignment of the roles to the single nodes in the `AppScalefile` is presented in Figure 10.

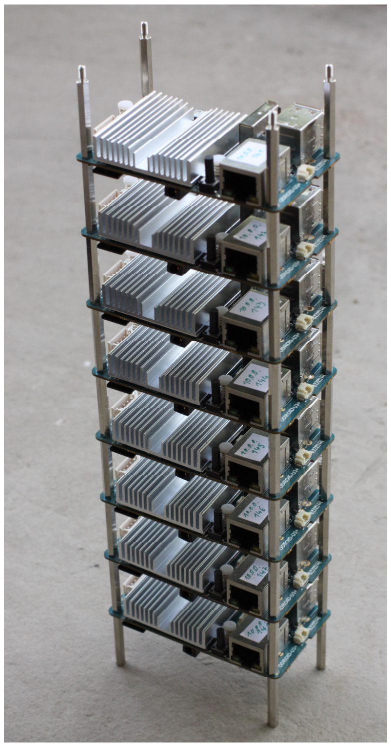
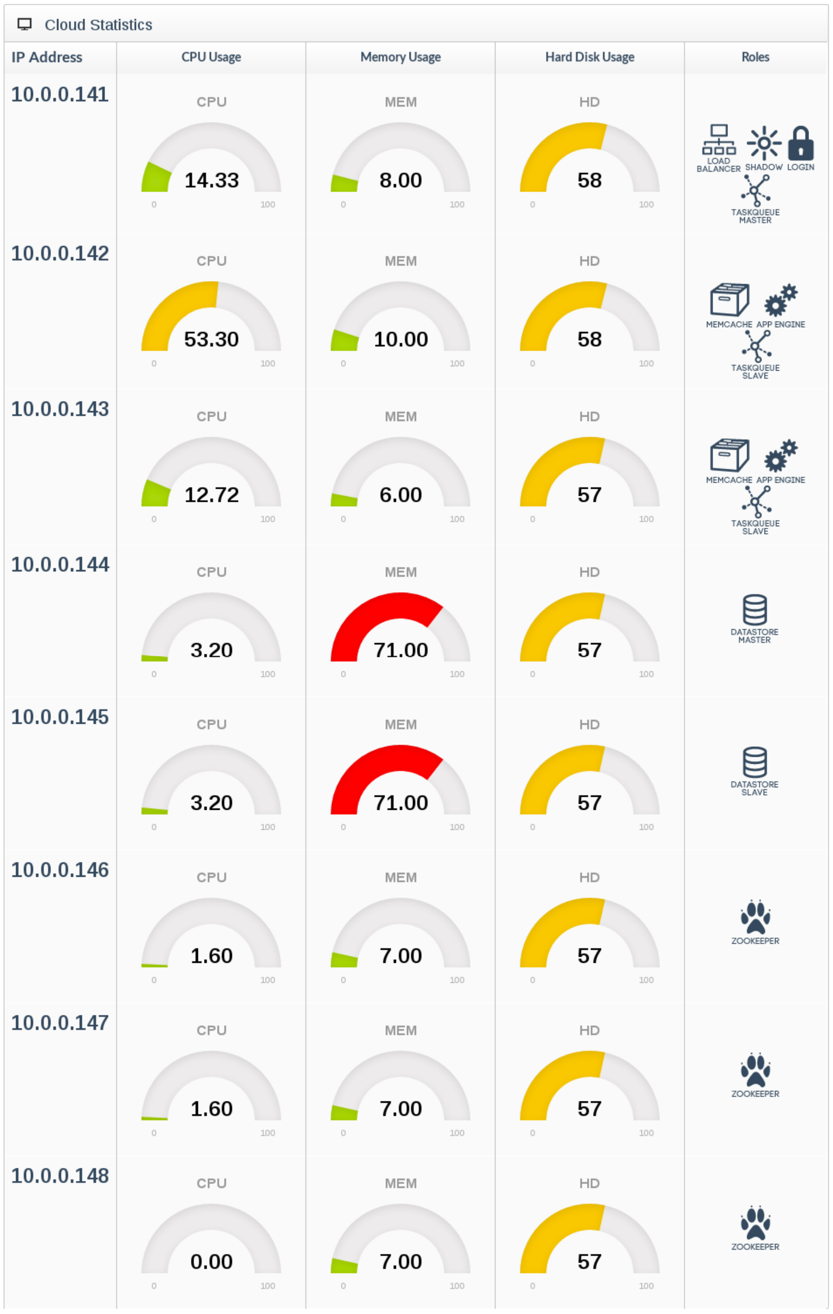


Figure 9. The Cluster of eight ODROID-U3 nodes

```
ips_layout:
  master : 10.0.0.141
  appengine:
   - 10.0.0.142
   - 10.0.0.143
  database:
   - 10.0.0.144
   - 10.0.0.145
  zookeeper:
   - 10.0.0.146
   - 10.0.0.147
   - 10.0.0.148
```

Figure 10. Assignment of roles to the nodes of the ODROID cluster in the AppScalefile

In contrast to the Raspberry Pi cluster, the presented configuration of eight ODROID-U3 nodes provides some failure resiliency because backup services do exist for the services, which are used to provide the App Engine and storage functionality (see Figure 11). Critical for the operation of the entire cluster is the first node, which operates as the master node. If this node fails, the PaaS does not work any longer. Also in this scenario the nodes which have the highest level of main memory utilization, are the one, which host Cassandra.



Cloud Statistics

IP Address	CPU Usage	Memory Usage	Hard Disk Usage	Roles
10.0.0.141	CPU 14.33	MEM 8.00	HD 58	LOAD BALANCER, SHADOW, LOGIN, TASKQUEUE MASTER
10.0.0.142	CPU 53.30	MEM 10.00	HD 58	MEMCACHE, APP ENGINE, TASKQUEUE SLAVE
10.0.0.143	CPU 12.72	MEM 6.00	HD 57	MEMCACHE, APP ENGINE, TASKQUEUE SLAVE
10.0.0.144	CPU 3.20	MEM 71.00	HD 57	DATASTORE MASTER
10.0.0.145	CPU 3.20	MEM 71.00	HD 57	DATASTORE SLAVE
10.0.0.146	CPU 1.60	MEM 7.00	HD 57	ZOOKEEPER
10.0.0.147	CPU 1.60	MEM 7.00	HD 57	ZOOKEEPER
10.0.0.148	CPU 0.00	MEM 7.00	HD 57	ZOOKEEPER

Figure 11. Resource utilization of the ODROID-U3 cluster, shown in the AppScale web user interface, shortly after AppScale has been started

5. Analysis of the Usability

Even after several memory tweaks have been applied to the operating system and Apache Cassandra, a single Raspberry Pi 2 provides just enough main memory to start the AppScale, but not to work with it. Because of the lack of free main memory, it was impossible to deploy web applications inside the running PaaS. Starting AppScale in this scenario requires approximately 16 minutes.

The cluster of four Raspberry Pi 2 nodes required approximately 29 minutes to start AppScale. The system is more usable, compared with the single node scenario, but because of frequent service crashes during this evaluation and the fact, that all user interaction caused much delay, the Raspberry Pi 2 cannot be recommended for hosting AppScale at all, not even in cluster operation mode.

Because the ODROID-U3 computer provides double the amount of main memory, compared to the Raspberry Pi 2, it is possible to operate AppScale for non-production purposes on a single device. Starting the AppScale in this scenario requires approximately 4 minutes and 30 seconds. The system operated stable over several hours.

The cluster of eight ODROID-U3 nodes required approximately 8 minutes and 30 seconds to start AppScale. The system operated in a stable way over several hours. Such a system may provide lesser performance, compared with a higher-value system like a physical or virtual server, but it provides a better level of reliability because of the existence of backup services.

6. ANALYSIS OF THE ACQUISITION COSTS

The acquisition costs for the scenarios, which are analyzed in this work, are presented in Table 2. The prices of all components were checked in March 2017 in Germany and they may vary on the free market. The price to implement the single node scenarios can be reduced by using another power supply because there is no need to acquire a 10 port power supply for a single device. The calculations in Table 2 do not include the price for a network switch because for this work, the network infrastructure was considered already existing and it is also not taken into account in the analysis of the energy-efficiency.

Table 2. Components used for this work and their acquisition costs

Component	**Single node Ras. Pi 2**	**Single node ODROID-U3**	**Cluster of 4 Ras. Pi 2**	**Cluster of 8 ODROID-U3**
Single board computer	40 €	70 €	160 €	560 €
microSD card (16 GB)	10 €	10 €	40 €	80 €
USB cable	1.25 €	1.25 €	5 €	10 €
USB power supply	35 €	35 €	35 €	35 €
Network cable CAT 5e	1.25 €	1.25 €	5 €	10 €
Price for the entire system	87.5 €	117.5 €	245 €	695 €

7. ANALYSIS OF THE ENERGY-EFFICIENCY

For the four investigated scenarios the overall power consumption has been evaluated, when AppScale is running (see Table 3). With each scenario, an Anker PowerPort 10 (Anker Model 2133 60 W, 5 V, 12 A) USB power supply was used. The results are influenced by the waste of electric energy, caused by the power supply, and they vary over time for approx. 10 %, depending of the load level and number of I/O operations.

Table 3. Power consumption of the investigated scenarios

Number of Devices	**System Architecture**	**Single Board Computer**	**Power Consumption**
1	Single node	Raspberry Pi 2	3 W
4	Cluster	Raspberry Pi 2	7 W
1	Single node	ODROID-U3	4 W
8	Cluster	ODROID-U3	18 W

The energy costs per year C_Y for a 24/7 usage for a specific power consumption in kW during operation E can be calculated with equation 1. In the equation, energy costs of 0.30 € per kWh are assumed.

$$C_Y = E \times 24 \frac{hours}{day} \times 365.25 \frac{days}{year} \times 0.30 \frac{€}{kWh} \qquad (1)$$

In a scenario where the cluster of eight ODROID-U3 nodes and the 10-port USB power supply runs all the time, the energy cost per year for 24/7 usage is just approx. 47.34 € (if the system consumes 18 W all the time).

8. Conclusions and Future Work

Constructing a private cloud PaaS with AppScale on a Cluster of single board computers may be an option for projects, which are focused on testing or educational purposes. It may also be useful in environments with limited space and/or energy resources. The evaluated cluster of eight nodes provides a good level of reliability for less than 700 € acquisition cost and less than 50 € cost for electric energy per year, but the size of the main memory limits the usability.

Next steps are the evaluation of a replacement solution for Apache Cassandra, which consumes lesser main memory resources.

Acknowledgements

This work was funded by the Hessian Ministry of Higher Education, Research, and the Arts ('*Hessisches Ministerium für Wissenschaft und Kunst*') in the framework of research for practice - ('*Forschung für die Praxis*').

Many thanks to Rosa-Maria Spanou, Henry-Norbert Cocos, Tobias Rohrer and Katrin Baun for their assistance in improving the quality of this paper.

References

[1] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M. (2009) "Above the Clouds: A Berkeley View of Cloud Computing", Technical report.

[2] Bunch, C., Chohan, N., Krintz, C., Chohan, J., Kupferman, J., Lakhina, P., Li, Y., Nomura, Y. (2010) "An Evaluation of Distributed Datastores Using the AppScale Cloud Platform", IEEE Cloud '10: Proceedings of the 3rd International Conference on Cloud Computing, pp 305-311

[3] Chohan, N., Bunch, C., Pang, S., Krintz, C., Mostafa, N., Soman, S., Wolski, R. (2009) "AppScale: Scalable and Open AppEngine Application Development and Deployment", CloudComp '09: Proceedings of the 1st International Conference on Cloud Computing, pp 57-70

[4] https://github.com/fajoy/typhoonae

[5] Baun, C. (2016) "Performance and Energy-Efficiency Aspects of Clusters of Single Board Computers", International Journal of Distributed and Parallel Systems, Vol. 7, No. 2/3/4, pp 13-22.

[6] Baun, C. (2016) "Mobile clusters of single board computers: an option for providing resources to student projects and researchers", SpringerPlus 5:360.

[7] Cox, S. & Cox, J. & Boardman, R. & Johnston, S. & Scott, M. & O'Brien, N (2009) "Iridis-pi: a low-cost, compact demonstration cluster", Cluster Computing, Vol. 17, No. 2, pp 349-358.

[8] Balakrishnan, N. (2012) "Building and benchmarking a low power ARM cluster". Master's thesis, University of Edinburgh.

[9] Kiepert, J. (2013) "Creating a Raspberry Pi-Based Beowulf Cluster", Technical report, Boise State University.

[10] Abrahamsson, P. & Helmer, S. & Phaphoom, N. & Nicolodi, L. & Preda, N. & Miori, L. & Angriman, M. & Rikkila, J. & Wang, X. & Hamily, K. & Bugoloni, S. (2013) "Affordable and energy-efficient cloud computing clusters: The bolzano raspberry pi cloud cluster experiment", Cloud Computing Technology and Science, IEEE 5th International Conference on CloudCom 2013, Vol. 2, pp 170–175.

[11] Sukaridhoto, S. & KHalilullah, A. S. & Pramadihanto, D. (2013) "Further Investigation on Building and Benchmarking A Low Power Embedded Cluster for Education", 4th International Seminar on Applied Technology, Science, and Art (APTECS).

[12] Ou, Z. & Pang, B. & Deng, Y. & Nurminen, J. K. & Ylä-Jääski, A. & Hui, P. (2012) "Energy- and Cost-Efficiency Analysis of ARM-Based Clusters", 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp 115–123.

[13] Tso, F.P. & White, D.R. & Jouet, S. & Singer, J. & Pezaros, D.P. (2013) "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing infrastructures", 1st International Workshop on Resource Management of Cloud Computing (CCRM).

[14] Pfalzgraf, A. M. & Driscoll, J. A. (2014) "A Low-Cost Computer Cluster for High-Performance Computing Education" IEEE International Conference on Electro/Information Technology, Milwaukee, pp 362-366.

[15] AppScale FAQs (2017) https://github.com/AppScale/appscale/wiki/FAQs

[16] http://www.finnie.org/software/raspberrypi/2015-04-06-ubuntu-trusty.zip

[17] https://github.com/AppScale/sample-apps

[18] http://odroid.us/odroid/odroidu2/debian/debian-wheezy-base-7.1.0.img.xz

AUTHORS

Dr. Christian Baun is working as Professor at the faculty of Computer Science and Engineering of the Frankfurt University of Applied Sciences in Frankfurt am Main, Germany. His research interest includes operating systems, distributed systems and computer networks.