Mapping Cross-Cloud Systems: Challenges and Opportunities

Yehia Elkhatib
School of Computing and Communications, Lancaster University, UK

Abstract

Recent years have seen significant growth in the cloud computing market, both in terms of provider competition (including private offerings) and customer adoption. However, the cloud computing world still lacks adopted standard programming interfaces, which has a knock-on effect on the costs associated with interoperability and severely limits the flexibility and portability of applications and virtual infrastructures. This has brought about an increasing number of cross-cloud architectures, *i.e.* systems that span across cloud provisioning boundaries. This position paper condenses discussions from the CrossCloud event series to outline the types of cross-cloud systems and their associated design decisions, and laments challenges and opportunities they create.

1 Introduction

Cloud applications are developed against a remote API that is independently managed by a third party, the cloud service provider (CSP). Instigated by changes (such as pricing), porting an application from consuming one set of API endpoints to another often requires a fair degree of re-engineering especially considering that even syntactically similar APIs could digress semantically [18]. As such, the increasing realisation of the inevitability of cross-cloud computing [23, 6, 7] led to various proposed solutions. As expected with such a nascent field, there is a certain degree of confusion arising from the use of non-convergent terminology: hybrid clouds, multi-clouds, meta-cloud, federated clouds, etc. The first contribution of this paper, thus, is to offer a coherent understanding of cross-cloud computing (§2) and the different terminology witnessed to date in this field. The second contribution is a classification to capture prominent efforts in this field (§3), describing their modus operandi and commenting on their suitability and limitations, and how they relate in terms of responsibility to the different stakeholders in cloud computing. The third and fourth contributions are a review of current challenges (§4) and an outlook on research opportunities (§5), respectively. These latter contributions are targeted towards mapping the future focus of application developers and researchers in this emerging field.

2 Why cross cloud boundaries?

A cross-cloud application is one that consumes more than one cloud API under a single version of the application. Let's consider a few examples drawn from real scenarios where developers are faced with the option to work with different APIs, *i.e.* cross cloud boundaries.

- Alan, an online service provider, finds that his user base is more fleeting than he planned for: web analytics indicates that a large proportion of users are accessing services through mobile devices and only for a few minutes (as opposed to hours as Alan originally envisioned). Alan decides to change how he manages his service infrastructure using ephemeral virtual machines (VMs) as opposed to dedicated long-life ones. He, thus, changes his business plan to employ a different CSP that charges by the minute rather than the hour, saving him hundreds of dollars each month in operational expenses.
- A company is consolidating some of its internal teams and, accordingly, their respective services will be unified into a single platform. Bella, the company's Chief Information Officer (CIO), is in charge of this task. Her objective is to keep all internal services operational and as as frictionless to use as possible during and after the transition. Bella finds that the teams to be consolidated have been using different public and private cloud infrastructures for different operations deep within their structure. This necessitates major changes to the underlying logic that handles task automation, service provisi-

oning, resource management, etc.

- An online gaming startup Casus is rapidly expanding its user base. The cloud allows Casus to consume an increasing amount of resources as and when required, which is extremely advantageous. However, the cloud does not necessarily aid in providing an optimized service to users who are not relatively close to any cloud datacenters, such as those in the Arabian Gulf region, western Africa, or central Asia. In order to cater to such users, Casus has to use innovative techniques to maintain high quality of experience (QoE). One such technique is to expand the housing of logic and data beyond any one CSP, but instead to be able to relocate on demand to local CSPs whilst maintaining service operation across the different infrastructure substrata.

A common thread to these scenarios is change to the predetermined plan relating to service provisioning, use, or management. Different parts of the application (virtualized infrastructure manager, load balancer, etc.) would need to be changed to call different APIs. Change is, of course, part of business. Hence, the need for cross-cloud systems grows greater as industries and societies increasingly use the cloud. Such change, however, entails fundamental changes to the communication behaviour to accommodate different semantics, charging model, and SLA terms. This is the core cross-cloud challenge.

Another commonality is the need to be free from long-term commitment. Many consumers choose the cloud for agility and elasticity. In the past few years, this was restricted to the boundaries of a single CSP but currently the trend is to transcend different CSPs. A recent survey [6] discovered that the "ability to move data from one service to another" ranked very highly as a concern raised by private sector SMEs and large organisations that use the cloud.

As such, a number of works in academia and industry have attempted to tackle this challenge using different strategies. Before attempting to categorize these works, it is perhaps important to point out the obvious: This is *not* a thesis for a universally uniform provisioning system. First, such "uber cloud" is unrealistic given the commercial nature of the market. Second, we believe it to be healthy to have a diverse cloud market where each provider brings a unique mix of specialized services that cater to a certain niche of the market.

3 Cross-cloud Dictionary

We identify four types of cross-cloud systems (Table 3) based on the prevalence in the literature and from experience running the CrossCloud[1] workshop series. We aim to provide some clarity about the terminology, and to use this moving forwards when discussing challenges.

3.1 Hybrid Clouds

Hybrid clouds are inherently made up of dissimilar parts. It is the responsibility of the system developer amalgamate these sub-clouds for the purposes of their application. This involves some logic to determine which sub-cloud is to be employed and when. Such policy is coupled to the rest of the application to some degree, and could be implemented as a separate module within the application or as a self-standing proxy between the application and the underlying clouds. In all cases, however, the policy enforcing element of the application will eventually interface with a finite set of cloud APIs. Such hardwiring is costly if the said APIs are altered after application development. Another hidden cost seldom discussed emerges from the forward design incurred to predetermine the exact sub-clouds to be employed. This is a crucial step to minimize the chance of changing the set of sub-clouds after application deployment.

Although hybrid cloud architectures are abundant, their highly customized nature does not necessarily lend them to reuse by others.

3.2 Multi-Clouds

Like hybrid clouds, *multi-cloud systems* build applications by employing different autonomous systems. However, in this model management is achieved via abstraction which relieves some of the cross-cloud responsibility from the end system developer. It also introduces a level of portability, a shortcoming of hybrid systems.

There are different forms of multi-clouds such as common programming models and API translation libraries. In either case, agreement on a least common denominator API between the different cloud systems supported by such abstraction environments does mean loss of some specialized services. Any need outside the unified abstract API[2] (*e.g.,* billing) hence needs to be addressed individually by the developer.

A notable example is the mOSAIC programming model [22]. Most of the traction multi-cloud engineering has had, however, came through the use of open source API translation libraries such as Libcloud, a Python toolkit that provides abstraction to 35+ compute and data service offerings. Alternatives include jClouds (Java), Fog (Ruby), DeltaCloud (Ruby), and pkgcloud (node.js).

Multi-cloud solutions could also operate at the networking level, creating an illusion of a single address space across multiple sub-infrastructures. An example is Neti[3] developed by Instagram to ease migration from Amazon EC2 to Facebook data centers.

Table 1: The different types of cross-cloud systems.

	Hybrid Cloud	Multi-Cloud	Meta-Cloud	Federated Cloud
Similarity of sub-clouds	Dissimilar	Similar	Similar	Very similar, with agreement
Abstraction	None	Some	High	Very high
Provisioning Means	Bespoke logic	Least common denominator API via a common programming model or a translation library	Delegating infrastructure instrumentation	Common API
Responsible Party	System developer	CPM or library developer	Third party brokers	Component cloud service providers
Examples	(domain specific; *e.g.,* [11])	Libcloud, jClouds, Fog, MODAClouds IDE	OPTIMIS, Jamcracker, Dell Cloud Marketplace	CIMI, OCCI, TOSCA, CDMI

3.3 Meta-Clouds (or *inter-clouds*)

Meta-clouds shift the responsibility even further away from the application developer, offering both abstraction and delegation. Meta-clouds are typically deployed through third party brokers that provide loosely-coupled interaction as a managed service [24]. Such brokers handle the discovery of suitable resources and subsequent life cycle management.

Some of these brokers evolved from meta-schedulers in grid computing where fine-grained infrastructure instrumentation is out-sourced to a centralized scheduler. An example is the work by Tordsson et al. [26] on a broker that optimizes placement and management of VMs across infrastructures. Other efforts approach brokerage by being more engrained into the application development process. The Optimis Toolkit prototype [12] constructs services as a configuration of *core elements* using a programming model and runtime. Each element is attached to a set of functional and non-functional requirements that are to be honoured by a central deployment manager that negotiates suitable resources.

3.4 Cloud Federations

A common condition in all of the above models is the lack of agreement amongst the sub-clouds. *Federated clouds*, on the other hand, achieve distribution through agreement that could be manifested as compliance to standard interfaces and/or data formats.

To highlight one example from academia, Celesti et al. [5] envisioned that CSPs will federate horizontally to gain market share and to further improve their capabilities by harnessing economies of scale. The Cross-Cloud Federation Manager (CCFM) is the solution they put forward to bridge the gaps between different providers, and is based on three stages: discovery, match-making and authentication. The solution is to be adopted by an CSP as a middleware to enable federation with other CSPs.

Standardisation efforts have aimed at creating a unified interface for working with resources and services across vendors. Examples include: the Cloud Infrastructure Management Interface (CIMI) [8], the Open Cloud Computing Interface (OCCI) [19], the Topology and Orchestration Specification for Cloud Applications (TOSCA) [3], and the Cloud Data Management Interface (CDMI) [1]. Other efforts have attempted to develop minimalistic APIs to support the most common interactions; *e.g.,* [9, 15]. Despite the presence of cloud standardisation groups, their efforts are yet to pay off in terms of wide market adoption [21]. The major CSPs probably see standards as giving customers the opportunity to effortlessly switch to competitors.

4 Challenges

We have elucidated how building cross-cloud systems does not come without cost. Until the rise of cost-effective brokerage services that a developer can offload the cross-cloud burden to, there will remain a significant development effort required to enable applications to seamlessly cross cloud boundaries. Additional costs and aftermaths are discussed here.

4.1 Developer Preferences

The choice between cross-cloud solutions is a trade off, in which the application at hand plays an obviously big role. The development trade off boils down to the amount of flexibility required and the extent of time available. Hybrid clouds provide the most flexibility as the developer can build a custom tailored infrastructure using native APIs from any provider as and when required. On the other hand, this is a relatively significant development effort for some projects. Multi-cloud solutions are favourable in terms of saving some of the developer's time when handling different APIs. The disadvantage, however, is the restriction to the least common denominator API. This is suitable for most applications, as is evident in the use of libraries such as Libcloud, but does

not go the full distance for others looking for deep control of their infrastructure. Another factor to bear in mind is the need to keep such libraries up to date with the latest APIs. Meta-clouds provide even more abstraction to the developer at the cost of less control, but these solutions are currently lacking in the market as will be discussed in §4.3. Finally, infrastructure management through cloud federation is perhaps the easiest for a developer, but are yet to be widely adopted by the major CSPs.

4.2 Value-added Services

By definition, building a cross-cloud application using any of the solutions discussed in §3 results in a system that is somewhat unhinged from the underlying infrastructure. To the developer, this incurs additional overhead in terms of manually managing infrastructure resources. Using high level added-value CSP services (such as AWS ElastiCache or Google DataFlow) become less of an option as they tether the system to a certain ecosystem. It is exactly this sort of binding that unintentionally grows with time, building a gravity towards a certain CSP above others.

This cost is the reason why to many building cross-cloud applications might be great as a concept but is in fact unlikely. In a sense, it dissolves the golden promise of the cloud computing paradigm as proffered by CSPs: "don't worry about the hardware, we'll take care of it; you just focus on your application".

4.3 Mind the Broker Gap

Another unsavoury reality is the state of the brokerage market. Despite the huge appeal for customers and the seemingly incessant forecasts about its potential (*cf.* [2, 14]), the market is in fact hopelessly slim. This could be attributed to a number of reasons, not least of which is entrusting a seemingly opaque third party (broker) to have huge influence about how the customer deals with a larger and even more opaque body (CSP). However, we find that this is not the core problem as evident by abundance of brokers in other markets that portray similar settings (*e.g.,* airline ticketing). The 'key disabler' here is expertise. A business that possesses the in-house expertise to manage different cloud infrastructures in a hybrid cloud setup (*e.g.,* Netflix) does not need to spend additionally on brokerage provided by a third party. In fact, an in-house built system would naturally be much more informed, customized and flexible.

This argument then suggests that companies that do not have such expertise internally (*e.g.,* young startups) are perhaps the prime consumers of cloud brokerage services. Alas, such SMEs are primarily focusing on market definition and horizontal expansion, and thus might not have cross-cloud computing as a priority.

Consequently, the brokerage business model is difficult leaving this market quite narrow [4]. Additional added value is required for the brokers to exist. We discuss one such added value in §5.1.

5 Opportunities

In light of the highlighted trends and challenges, we now discuss some of the main opportunities and associated open questions. We exclude the subject of standard adoption as it has been well covered by others (*cf.* [25]).

5.1 Strategic Decision Making

Customers moving between CSPs (and indeed those entering the market) need assistance in choosing the right services for their requirements and constraints. This creates an opportunity for non-partisan consulting services that provide automated recommendations about which cloud services to use, and when and where to migrate.

Recent advancements in machine learning provide great opportunities here. Such sophisticated decision making has been until very recently hindered by the cost of benchmarking data. However, cloud performance metrics are now relatively inexpensive: *e.g.,* NewRelic[4], ThousandEyes[5], and CloudHarmony[6]. The evidence-based consulting frameworks envisaged here would enable additional services such as arbitrage and life cycle management, improving the brokerage market prospects.

- How to harness performance metrics to gain knowledge in different domains? *i.e.* between different CSPs, between different customer applications.
- What is the best way of quantifying the cost of migrating between CSPs?
- How to provide strategic consultation for a large number of customers while avoiding/minimizing thrashing and maximizing overall utility?

5.2 Crowd-sourced Cloud Knowledge

Companies such as NewRelic and ThousandEyes (mentioned in the previous subsection) make a business out of finding user-reported cloud performance metrics. CloudHarmony perturbed the market by openly disseminating benchmarking data they collect. In a cross-cloud capable world, such information is transformative as it allows users to vote "with their feet", switching providers whenever they do not receive the service they are paying for.

Furthermore, cloud metrics are becoming even more accessible and comparable using tools such as Google's open source PerfKit[7]. This allows customers to quantify and compare experiences about CSP usage, potentially

creating a crowd-sourced knowledge base. Further, Microsoft is leading the way in obtaining billing information via an API[8].

The next logical step is to provide means of verifying whether SLAs are breached or not. Indeed, researchers' attention has started to turn towards means of programmatically verifying SLAs [16]. Coupled with crowd-sourced cloud knowledge, this would empower customers and help them make informed choices about competing services. It also has the potential to change the market by putting pressure on CSPs and increase competitiveness.

- How can we accumulate knowledge of how end users consume resources and the QoS they experience? Could this knowledge be matched with application profiles and usage context to provide a fine-grained insight into cloud performance?
- Could such community knowledge be used to verify SLAs in an automated manner?
- What would the effect of such evidence-based culture be on the distribution of market share between the different major and small CSPs? How would it affect their service offerings and pricing, if at all?

5.3 Edge Market Places

The need for cross-cloud computing towards the edge of the network is increasingly important as more resources are available nearer to the end users [27, 13]. This is especially true for locations where users are geographically distant from data centers [10], *e.g.,* emerging economies with rapidly growing consumer markets but with no close data centers. This puts such promising markets at a disadvantage. Major CSPs are racing to set up data centres in such areas (*cf.* [20]). However, such expansion is a highly expensive, long term investment. Moreover, it cannot reach all users due to the sheer speed of growth of certain markets. Hence, there is a need for deploying cloud applications towards the edge.

This has been discussed for a few years under different names including 'cloudlets' and 'offloading'. However, previous efforts have been primarily motivated by the convergence of mobile and cloud computing and as such were focused on housing particular application architectures (*e.g.,* backend for mobile services) in an environment with certain stakeholders, such as ISPs. What we discuss here is fundamentally different. Cross-cloud computing at the edge is growing from the bottom up where the application could practically be of any type or scale, and the stakeholders are largely the consumers themselves. In such ecosystem, there is room for forming 'community clouds' as well as for entrepreneurs to provide 'mini-clouds' to serve small communities. This, in essence, is the rise of fog computing: the diminishing reliance on the cloud data centers and the growth of heterogeneous computing infrastructures nearer to the end users.

- How to piece together dissimilar cloud blocks to form a larger pool of resources without barriers?
- What solutions are needed to enable applications to easily straddle infrastructures in a hybrid fashion?
- How to minimize gravity to any infrastructure and ease difficulties of moving between infrastructures?
- Could major CSPs enter the fog market by providing services outside of their mega-data centers?

5.4 Migration Vehicles

Undoubtedly, virtualization plays a fundamental role in cloud computing, enabling the dynamic provisioning of bespoke environments. However, VMs as packaged memory and disk state are too expensive for cross-cloud purposes, both in network and operation terms. Containers (*e.g.,* Docker[9] and rkt[10]) and unikernels [17] provide lightweight alternatives that offer a lot of promise. These are much more appropriate as portable vehicles to (re)deploy applications. VMs provide immutable infrastructure which is cumbersome to modify and manage once the VMs are 'baked'. In contrast, containers and unikernels have good exposure to the developer and are thus much more easily controllable. They are also well integrated with different infrastructure management tools like Puppet and Vagrant. In addition, containers and unikernels are much more efficient in sharing resources at the OS level compared to hardware virtualization by hypervisors. In effect, this opens up migration possibilities with more hosting destinations.

- Which of these technologies is the best vehicle for migrating different workloads?
- How to create *stateful* containers to transfer operational workloads across with ease and still maintain the low network cost of current containers?
- What does the life cycle management of containers and unikernels across disparate host infrastructures look like? Could we learn from solutions developed for managing VMs?

6 Concluding Remarks

We gave an overview of cross-cloud computing, putting forward a dictionary to formalize the differences between the solutions proposed in the literature. We also discussed challenges and opportunities unravelled by the emergence of cross-cloud solutions. Cloud computing began and remains an industry-driven domain, but there is plenty of room for work done by system engineers not necessarily working with the big cloud players.

References

[1] Cloud data management interface (CDMI). Tech. Rep. CDMI V1.1.1, Storage Networking Industry Association (SNIA), 2015.

[2] BADGER, L., BOHN, R., CHU, S., HOGAN, M., LIU, F., KAUFMANN, V., MAO, J., MESSINA, J., MILLS, K., SOKOL, A., TONG, J., WHITESIDE, F., AND LEAF, D. US government cloud computing technology roadmap. Tech. Rep. 500-293, National Institute of Standards and Technology, Nov 2011.

[3] BINZ, T., BREITER, G., LEYMAN, F., AND SPATZIER, T. Portable cloud services using TOSCA. *IEEE Internet Computing 16*, 3 (2012), 80–85.

[4] CARTLIDGE, J., AND CLAMP, P. Correcting a financial brokerage model for cloud computing: closing the window of opportunity for commercialisation. *Journal of Cloud Computing 3*, 1 (2014), 1–20.

[5] CELESTI, A., TUSA, F., VILLARI, M., AND PULIAFITO, A. How to enhance cloud architectures to enable cross-federation. In *IEEE Cloud* (July 2010), pp. 337–345.

[6] CLOUD STANDARDS COORDINATION (PHASE 2). Cloud computing users needs - analysis, conclusions and recommendations from a public survey. Tech. Rep. SR 003 381 V1.0.0, The European Telecommunications Standards Institute (ETSI), June 2015.

[7] COADY, Y., HOHLFELD, O., KEMPF, J., MCGEER, R., AND SCHMID, S. Distributed cloud computing: Applications, status quo, and challenges. *SIGCOMM Comput. Commun. Rev. 45*, 2 (Apr 2015), 38–43.

[8] DAVIS, D., AND PILZ, G. E. Cloud infrastructure management interface (CIMI) model and RESTful HTTP-based protocol - an interface for managing cloud infrastructure. Tech. Rep. DSP0263 V1.0.1, Distributed Management Task Force (DMTF), Sep 2012.

[9] DODDA, R., SMITH, C., AND VAN MOORSEL, A. An architecture for cross-cloud system management. In *Contemporary Computing*, S. Ranka, S. Aluru, R. Buyya, Y.-C. Chung, S. Dua, A. Grama, S. Gupta, R. Kumar, and V. Phoha, Eds., vol. 40 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, 2009, pp. 556–567.

[10] ELKHATIB, Y. Building cloud applications for challenged networks. In *Embracing Global Computing in Emerging Economies*, R. Horne, Ed., vol. 514 of *Communications in Computer and Information Science*. Springer, Nov 2015, pp. 1–10.

[11] ELKHATIB, Y., BLAIR, G. S., AND SURAJBALI, B. Experiences of using a hybrid cloud to construct an environmental virtual observatory. In *The 3rd Workshop on Cloud Data and Platforms* (April 2013), pp. 13–18.

[12] FERRER, A. J., HERNÁNDEZ, F., TORDSSON, J., ELMROTH, E., ALI-ELDIN, A., ZSIGRI, C., SIRVENT, R., GUITART, J., BADIA, R. M., DJEMAME, K., ZIEGLER, W., DIMITRAKOS, T., NAIR, S. K., KOUSIOURIS, G., KONSTANTELI, K., VARVARIGOU, T., HUDZIA, B., KIPP, A., WESNER, S., CORRALES, M., FORGÓ, N., SHARIF, T., AND SHERIDAN, C. Optimis: A holistic approach to cloud service provisioning. *Future Generation Computer Systems 28*, 1 (2012), 66–77.

[13] GARCIA LOPEZ, P., MONTRESOR, A., EPEMA, D., DATTA, A., HIGASHINO, T., IAMNITCHI, A., BARCELLOS, M., FELBER, P., AND RIVIÈRE, E. Edge-centric computing: Vision and challenges. *SIGCOMM Comput. Commun. Rev. 45*, 5 (Sep 2015), 37–42.

[14] JENNINGS, B., AND STADLER, R. Resource management in clouds: Survey and research challenges. *Journal of Network and Systems Management* (2014), 1–53.

[15] JOHNSTON, S. Simple workload & application portability (SWAP). In *First International Workshop on CrossCloud Computing, IEEE INFOCOM* (April 2014), pp. 37–42.

[16] LANGO, J. Toward software-defined SLAs. *Queue 11*, 11 (Nov 2013), 20–31.

[17] MADHAVAPEDDY, A., MORTIER, R., ROTSOS, C., SCOTT, D., SINGH, B., GAZAGNAIRE, T., SMITH, S., HAND, S., AND CROWCROFT, J. Unikernels: Library operating systems for the cloud. In *ACM SIGPLAN Notices* (2013), vol. 48, pp. 461–472.

[18] MARTINO, B. D. Applications portability and services interoperability among multiple clouds. *IEEE Cloud Computing 1*, 1 (2014), 74–77.

[19] METSCH, T., EDMONDS, A., AND (EDITORS), R. N. Open cloud computing interface - core. Tech. Rep. GWD-R, Open Grid Forum (OGF), Nov 2010.

[20] MSV, J. Amazon follows microsoft's footsteps, announces India region. http://www.forbes.com/sites/janakirammsv/2015/07/02/amazon-follows-microsofts-footsteps-announces-india-region/, Jul 2015. Accessed 23-Feb-2016.

[21] ORTIZ JR, S. The problem with cloud-computing standardization. *IEEE Computer 44*, 7 (2011), 13–16.

[22] PETCU, D., MACARIU, G., PANICA, S., AND CRĂCIUN, C. Portable cloud applications–from theory to practice. *Future Generation Computer Systems 29*, 6 (2013), 1417–1430.

[23] RIGHTSCALE. 2015 state of the cloud survey. Tech. rep., 2015.

[24] SATZGER, B., HUMMER, W., INZINGER, C., LEITNER, P., AND DUSTDAR, S. Winds of change: From vendor lock-in to the meta cloud. *IEEE Internet Computing 17*, 1 (2013), 69–73.

[25] SILL, A. The role of communities in developing cloud standards. *IEEE Cloud Computing 1*, 2 (July 2014), 16–19.

[26] TORDSSON, J., MONTERO, R. S., MORENO-VOZMEDIANO, R., AND LLORENTE, I. M. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Generation Computer Systems 28*, 2 (2012), 358–367.

[27] VAQUERO, L. M., AND RODERO-MERINO, L. Finding your way in the fog: Towards a comprehensive definition of fog computing. *SIGCOMM Comput. Commun. Rev. 44*, 5 (Oct 2014), 27–32.

Notes

[1] `http://bit.ly/CrossCloud`

[2] "One Interface To Rule Them All", as described on the Libcloud webpage: `https://libcloud.apache.org/`

[3] `http://instagram-engineering.tumblr.com/post/89992572022/migrating-aws-fb`

[4] `http://newrelic.com/`

[5] `https://thousandeyes.com/`

[6] `https://cloudharmony.com/`

[7] `https://github.com/GoogleCloudPlatform/PerfKitBenchmarker`

[8] `https://msdn.microsoft.com/library/azure/1ea5b323-54bb-423d-916f-190de96c6a3c`

[9] `https://docker.com/`

[10] `https://coreos.com/rkt`