Responses to Reviewers

Thank you all very much for the useful comments and inputs related to the FluidCloud paper. Below are the measures that we took to address each reviewers inputs. We hope they satisfy each of you and of course are alway open for more discussion.

Reviewer #45A

Comments for author

Very good coverage of a solution to the lock-in that happens in a cloud environment by providing mobility of the service instance, adapting the instance to the new cloud platform, and the handling of the associated data.

Can you provide guidance on how FluidCloud can work within regulatory compliance guidelines? Have you considered extensions to ensure that those compliance guidelines are not broken? Consider Tier-1 applications under PCI, HIPAA, or FERPA compliance.

Authors: Yes - this would be related to service owners supplying some form of rule based policy to the Migrator which would form part of the decision process shared between the migrator and broker. This could be one of those interesting topic which come in the aftermath of this Proof of Concepts. Certainly work from the Cloud Security Alliance (CSA) work on A6 might be interesting.

CLARIFY: How do you deal with changes to the cloud platform? Does FluidCloud have the ability to dynamically update the Service Instance Adaptation component?

Authors: What sort of changes in the platform are you referring to? Currently dynamic updates while the migration is in-process is not present. It would be great to understand the use case that you might have in mind.

The architecture is well thought out and addresses most of the questions and requests that arise from enterprise cloud customers that want flexibility to move between cloud providers based on cost, performance, availability, and other requirements.

Reviewer #45B

Comments for author

While I agree with the argument that relocation of service instances between different cloud providers brings more flexibility to both cloud service owners and their instances, I am not fully convinced the proposed framework, FluidCloud, is well-thought out to accomplish this goal.

First, the framework is not comprehensive enough to handle a variety of relocation scenarios. The paper mentions that FluidCloud supports relocation of IaaS-based, Paas-based, and Iaas-to-PaaS services, whereas Section 4 only discusses the IaaS case. It would be good to at least conceptually discuss the other two cases. For the IaaS case, have you considered the heterogeneity of underlying hypervisors (e.g., KVM, Xen, VMware, and Hyper V) in different clouds? Relocation between clouds with different hypervisors will pose non-trivial system-level challenges.

Authors: IaaS is the current target of exploration however we have run through informal scenarios regarding PaaS-PaaS migration and IaaS->PaaS. Yes for the IaaS case there are challenges in VM formats and descriptions however there are many conversions tools as well as description schemas (e.g. OVF) that resolve many issues. Naturally if the migration cannot happen then the service should simply say so after pre-migration checks.

It seems that FluidCloud needs to provide a group of interfaces for each cloud stack to operate VMs inside the cloud stack, such as pausing, booting, copying, and transferring VMs. Existing RESTful Cloud APIs might be insufficient to handle all such operations. For example, in section 4, how does SmartOS “re-connect” to the object storage created by OpenStack via the current RESTful APIs?

Authors: Yes and FluidCloud would seek to leverage and contribute to existing work in this area such as jClouds and Apache libcloud.

Furthermore, I don’t think it’s reasonable and practical to share the same storage resource between different cloud stacks, which would cause problems such as complexity of management, (in)compatibility of cloud stacks and (in)security of data. However, if the storage resource cannot be shared between cloud stacks, how would FluidCloud relocate the object storage?

Authors: Agreed with sharing for resources between cloud platforms and this is not our intention either. If the storage resource in question is the object storage then in our view this is not a resource but a service and one that is a separate concern, which is perfectly good at managing multiple read/writers (e.g. S3, Swift), albeit with the constraints of eventual consistency.

The description of the migration process in Section 4 is confusing. Since “Both machines are then booted using a prepared Ubuntu iso image”, why do you need to boot the VM again “After copying the virtual machine on the SmartOS platform”?

Authors: We need to boot with ubuntu to do the dd operation - the passage has been

extended to more clearly work out this point.

As described in Section 3, one of Broker's functions is to inspect the service instances. That is inconsistent with Figure 2, where CloudConduit is responsible for the inspection.

Authors: This has been fixed in the camera-ready version

Reviewer #45C

Comments for author

I'm voting to accept this paper because the topic is interesting and would require significant work to make it possible, some of which is described and partially evaluated in this paper. Despite the concerns I list below, this is a topic that should cause lots of discussion and is therefore a good one for HotCloud.

While I sympathize with the goal of making it possible to relocate cloud services at will, I'm not sure I believe it is entirely achievable or that if achieved it will actually result in the benefits the authors describe. For instance, if I understand the evaluation section, it seems unlikely in practice that cloud services with a lot of storage (a growing number) will end up relocating the storage across providers on a regular basis. It's not clear that the overall usage will therefore justify all this work. It would be nice if the authors could say something more about the number and type of services that would find this practical. While the paper mentions that FluidCloud is intended for IaaS and PaaS services, I don't see much about PaaS. Has it just not been addressed yet?

Authors: Yes it has not been fully addressed at this point in time and this has now been hopefully clarified in the camera-ready version of the paper

I think this work might also be jeopardized if cloud providers end up doing anything to restructure their fees to penalize frequent relocation, which they might.

Authors: A fair point and something that is not unique to IaaS/PaaS cloud services. For example Microsoft has continually locked implementers out from the once MSN messenger service. Nonetheless this spurred many on to realise and argue for more open access to such protocols.

I think that overall there is a lot more work that would need to be done to implement this broadly enough, and that a lot of the struggle really will be in the details. But it is a good hot topic for the workshop and the community would benefit from this.

Stylistically, the cloud liberation freedom front verbiage is perhaps a little overdone in the first half of the paper. For instance the bit about they "should have right of movement for those services" could perhaps be "should have the option of movement for those services. I somehow don't see this as a right equivalent to property rights or freedom from slavery...

Authors: Fixed - Call us idealists, or heaven forbid futurists however we certainly don't see it as an equivalent to freedom from true real-word slavery or tyranny. In that case we're realists.

"ad hoc"should not be hyphenated.

Authors: Fixed

Should "consequentially" be "consequently"?

Authors: Fixed

I'm not sure I understand the "Regulatory" bit in the last paragraph of section 2.1. I think of Regulation as being about governments or overseeing bodies setting up rules, but that doesn't seem like what this means. This seems to be about recognizing with a service is at risk? Why is this regulatory?

Authors: there are some regulatory obligations, especially here in Europe, that cloud service providers need to observe. These mainly relate to operating in certain legal jurisdictions, which essentially means geo-location of where services are provisioned. As you rightly note, it is about managing the risk and should a violation occur the respective tooling can be provided by FluidCloud.

I'm still confused about the amount of work that needs to be done to make this work, even for the IaaS case. Won't there be incompatibilities between hypervisors? How do you address this or am I confused and you don't need to somehow?

Authors: Yes for the IaaS case there are challenges in VM formats and descriptions however there are many conversions tools as well as description schemas (e.g. OVF) that resolve many issues. The migrators in the architecture are the conceptual elements that are responsible for any potential conversions needed.

I know that "re-contextualized" refers to previous work, but a brief definition where it appears in this paper would be helpful.

Authors: Fixed

Figure 1 and even 2 are too small for the fonts to be easy to read.

Authors: Fixed

URLs show up sometimes in the text and other times in footnotes, but it would be nice if there were consistency.

Authors: Fixed

If you're going to define PaaS then it makes more sense to do so when it first appears rather than when it last appears.

Authors: Fixed

There are a bunch of typos in the references. For instance, "Jr, S. O." should be "Ortiz Jr., S." or just Sixto Ortiz Jr. as some of the other references are spell out that way.

Authors: Fixed