Read to Accelerate Instance Provisioning in Hybrid Cloud with Zero Capex

Hybrid IT provides on-demand creation of infrastructure either from public or private or both. The infrastructure management activities (like creation of instance, monitoring, consumption analysis etc) are typically performed using set of REST based services running from centralized place. The set of services often referred as **Hybrid ITaaS (IT as a service) portal** and it can be hosted either in public or private cloud. Hybrid IT has brought choices but with new set of challenges like mentioned below and many more!

1. On-demand fast provisioning of virtual infrastructure
2. Seamless deployment of workload in public cloud or private cloud or both
3. Migration of workload across cloud
4. Security when things are crossing boundaries of the cloud

The focus of this article is on with focus on other topics for future:

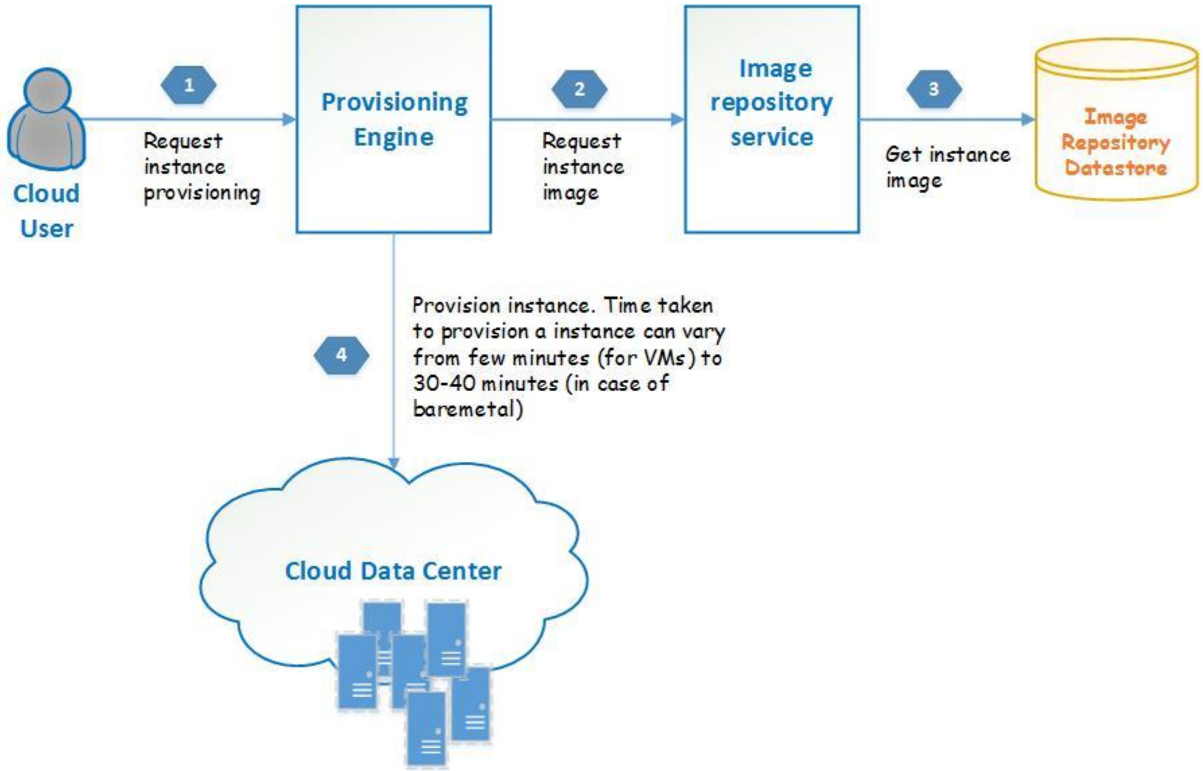
On-demand Accelerate provisioning of virtual infrastructure without involving any Capex

Problem

Let us spend some time describing the problem first. As mentioned above, Hybrid ITaaS portal is typically located in centralized place (public or private) typically. And, user does creation of workload using Hybrid ITaaS portal. Creation of each workload natively involves creation of compute instances (VM or baremetal). The traditional provisioning workflow involves the following steps:

1. Locating the cloud datacenter to host user instances.
2. Pushing the instance image to cloud, if required
3. Provisioning of instance

The infrastructure provisioning time can vary from **few minutes (for small VM instances) to 30-40 minutes or more for baremetal**. The provisioning turnaround time becomes more bothersome if there are concurrent request either because of cloud bursting or multiple users trying to create workload at the same time which is common pattern seen in cloud environment. The consumed time depends upon type of instance (baremetal or VM), OS image size (more for windows images), and type of cloud (e.g. AWS takes 6-7 minutes to create an instance whereas OpenStack cloud takes 3-4 minutes), compute processing time to exercise the provisioning workflow.


Cloud User
1
Request instance provisioning
Provisioning Engine
2
Request instance image
Image repository service
3
Get instance image
Image Repository Datastore
4
Provision instance. Time taken to provision a instance can vary from few minutes (for VMs) to 30-40 minutes (in case of baremetal)
Cloud Data Center

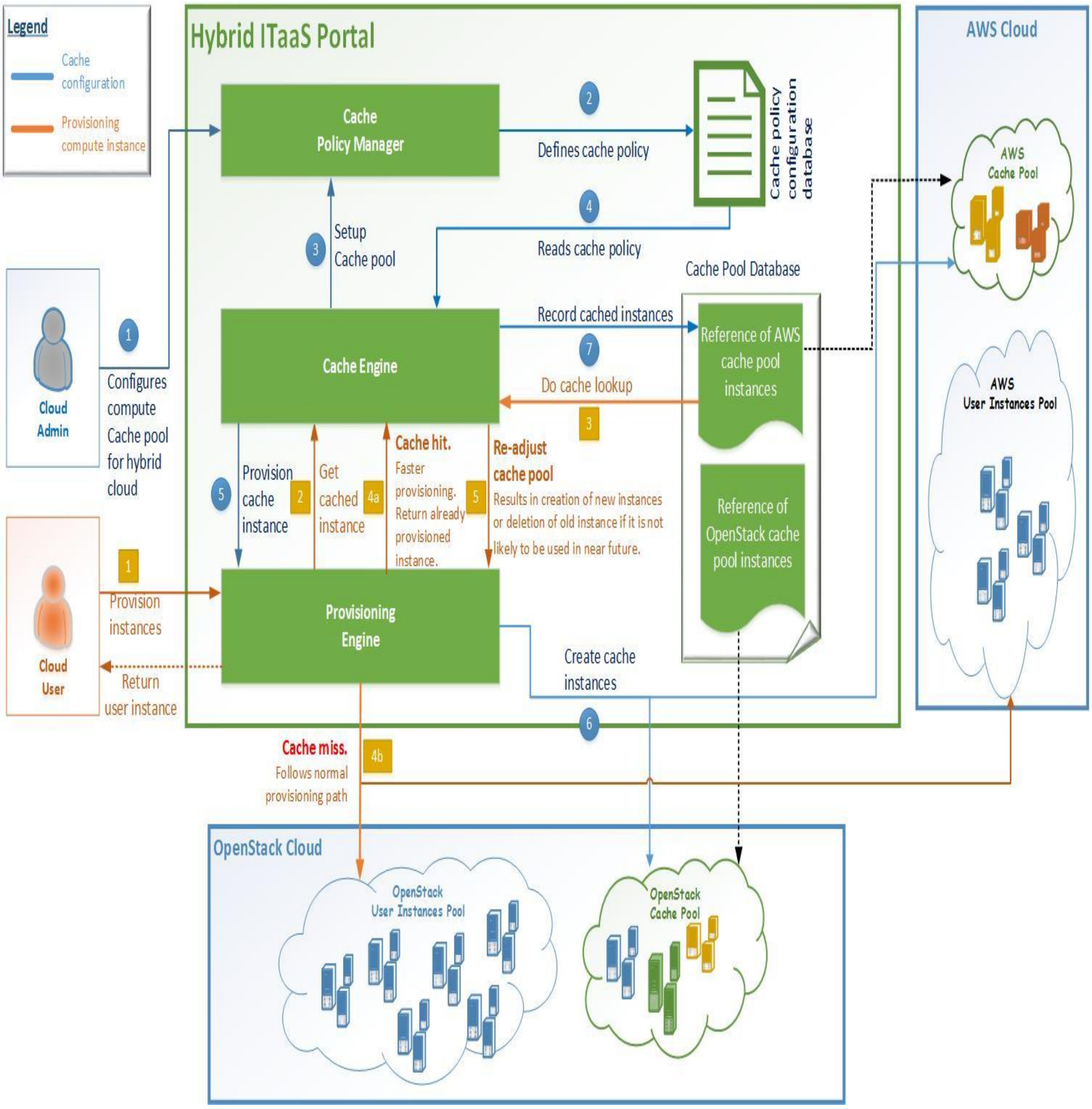
Homogeneity among heterogeneity: do you see it?

Natively, it is apparent that heterogeneity is inherent property of cloud. In other words, kind of cloud application can be anything under the horizon like MySQL, apache, mongodb, Jenkin, gerrit, wordpress etc. But, if we double click the infrastructure being used underneath by applications we do find that heterogeneity is not as diverse as it appears from the surface. Each of the above mentioned application typically runs on compute instances (VM or baremetal) of pre-defined attributes like memory, disk size or CPU count. **The compute attributes is expressed in terms of instance flavors.** For VM, it is like tiny instance, small instance, medium instance etc. For baremetal, it can be DL380, HPE Synergy s480, SL 4520 etc. Because of this commonality, we can device a caching solution for compute instances.

Solution

The solution is to leverage above mentioned homogeneity amidst of heterogenitty by having a **Hybrid Cache Pool** which maintains already provisioned virtual instances for each cloud data center. As customer requests provisioning of compute instance, the provisioning engine (running in Hybrid ITaaS portal) looks for instances in cache pool. If requested instance is already present in cache pool (**often termed as cache hit**) then the provisioning engine returns the already provisioned instances. **It effectively reduces instance provisioning time significantly to few seconds as only tasks needed to do is of some book keeping (in cache pool)** instead of time consuming tasks like uploading image, going through each steps of provisioning workflow etc. If cache pool doesn't contain the requested flavor of instances **(often termed as cache miss),** the provisioning engine follows the normal path and henceforth takes default time to serve the request. At the same time, cache implementation logic keeps track of kind of instances are being provisioned so that it can take advantage of temporal coherence (the most recently provisioned instance is likely to be provisioned in near future) and spatial coherence (the cloud used to launch instance is likely to be used again). Or, it can apply logistic regression, a statistical analysis method used to predict a data value based on prior observations of a data set. The dynamic adaption of cache pool helps in increased probability of cache hit and hence better utilization of solution.

Don't trust in its realization? Here is high level design to convince you further!



Still asking yourself: what you will get?

1. Faster provisioning of instances
2. Almost constant provisioning time irrespective of location and type (AWS, OpenStack) of cloud

3. Improved hybrid cloud experience for end user especially if instances are of baremetal type
4. Higher probability of meeting SLA for provisioning turnaround time

Many times, life is about doing same things but in different way! Here is one opportunity.

Are you ready to for Acceleration?