Migrating SimCloud to HP Helion

Francisco Oblea, Annabelle Eseo Votaw, Sunil Kothari, Jun Zeng

HP Laboratories
HPL-2014-36

Keyword(s):
Cloud deployment; HP Helion; OpenStack; Software as a Service

Abstract:
This is our submission to HP Helion Tech Champs Contest. SimCloud is a Simulation as a Service (SaaS) created by HP Labs/Print and Content Delivery Lab driven by the need from PPS/Graphics business for a recommendation tool that can quantitatively differentiate amongst different equipment configuration for commercial and industrial printing factories - a key growth area for HP's printing business for both hardware and consumables. As an always-on cloud service running inside HP Labs' private cloud, SimCloud has helped to standardize the designs of both hardware configuration and workflow through HP's solution architects worldwide. The migration journey of SimCloud to HP Cloud Service was triggered at the end of last year when a customer put forward a request to HP/Indigo senior management for SimCloud access which required to bring SimCloud from behind the firewall to a public cloud platform. Significant research and learning has gained by the team during the process of migrating the architecture and code-base into HPCS OpenStack (v12.12), and by May of this year, HP Helion (v13.5). Today SimCloud has been successfully running on Helion for over a month. In this technical report we describe Simcloud's migration journey. While we are telling a SimCloud's story, our experiences should resonate with many cloud computing professionals who are contemplating to embark on their own HP Helion journey.
Migrating SimCloud to HP Helion
Francisco Oblea\textsuperscript{2}, Annabelle Eseo Votaw\textsuperscript{1}, Sunil Kothari\textsuperscript{1}, Jun Zeng\textsuperscript{1}

\textsuperscript{1}HP Labs, \textsuperscript{2}GIT/GBS

\{francisco.oblea; annabelle.eseo; sunil.kothari; jun.zeng\}@hp.com

Abstract

This is our submission to HP Helion Tech Champs Contest\textsuperscript{1}.

SimCloud \cite{1} is a Simulation as a Service (SaaS) created by HP Labs/Print and Content Delivery Lab driven by the need from PPS/Graphics business for a recommendation tool that can quantitatively differentiate amongst different equipment configuration for commercial and industrial printing factories (Fig. 1) – a key growth area for HP’s printing business for both hardware and consumables. As an always-on cloud service running inside HP Labs’ private cloud, SimCloud has helped to standardize the designs of both hardware configuration and workflow through HP’s solution architects worldwide.

The migration journey of SimCloud to HP Cloud Service was triggered at the end of last year when a customer put forward a request to HP/Indigo senior management for SimCloud access which required to bring SimCloud from behind the firewall to a public cloud platform. Significant research and learning has gained by the team during the process of migrating the architecture and code-base into HPCS OpenStack (v12.12), and by May of this year, HP Helion (v13.5). Today SimCloud has been successfully running on Helion for over a month.

In this technical report we describe Simcloud’s migration journey. While we are telling a SimCloud’s story, our experiences should resonate with many cloud computing professionals who are contemplating to embark on their own HP Helion journey.

Introduction

Fig. 2 shows SimCloud architecture. The Scenario Factory module in SimCloud Server uses shared vocabulary databases including production plan catalog, printing process catalog and equipment catalog to translate the interview data (stored in a private database with

\footnote{1 https://ent301.sharepoint.hp.com/teams/heliontechchamps/}
access control) into factory schematics that serve as simulation [2] inputs. It populates the interview data into multiple \( (N) \) scenarios covering all possible meaningful combinations of demand trends and resource compositions. The Cluster Executor module simulates and analyzes all \( (N) \) scenarios concurrently by grouping them into a simulation cluster (Fig. 3) which is composed of \( (2N+1) \) virtual machines (VMs). One VM serves as the cluster’s web-based access gateway. Each scenario includes a VM that executes simulation and another VM hosts simulation input and output databases. The Cluster Executor includes scripts to provision necessary VMs on demand based on specifications provided by Scenario Factory, construct the cluster following the interconnection topology as shown in Fig. 3, monitor the simulation progress and VM health, dynamically terminate and release VMs once the simulations are completed, and transmit data between VMs and the Scalable Persistent Store. Because of the observed uncertainty in producing a healthy VM, we have incorporated additional failure detection and recovery features into our deployment scripts. For instance, we use an over-provision factor in VM acquisition. Instead of terminating all idled VMs, we implement server pooling and recycling to retain a small number of freshly idled VMs to supplement the creation of the next simulation cluster. Example failure modes we have encountered include time taken to generate a new VM varies sizably, and failures to acquire proper public/private IP addresses or to open necessary listener ports for Tomcat application servers and database servers. Since the deployment of this SaaS application in HP Labs’ private cloud, it has attracted attentions of solutions architects of multiple divisions covering wide spectrum of commercial and industrial printing. HP Labs’ management estimates this SaaS will result in incremental HP revenue of tens of millions over a 5 year time horizon.

RPI is one of the largest make-on-demand manufacturers of personalized consumer print products; it is one of the top 20 producers of HP/Indigo impressions worldwide. Last December RPI put forward a request through HP/Indigo’s senior management for SimCloud access. To enable that we will need to deploy SimCloud to a public cloud platform; this triggered Simcloud’s journey of migration to HP Cloud Services. We first deployed SimCloud at HPCS OpenStack (v12.12). In May of this year, HPCS announced Helion (v13.5) and the subsequent retirement of v12.12. We continued the migration into Helion. Today SimCloud has been running on Helion for over a month.

From system architecture perspective, SimCloud is a typical SaaS application. Migrating SimCloud from a private cloud running on Eucalyptus to HPCS OpenStack public cloud, and eventually HP Helion, was not without challenge. This report describes our experiences and lessons. We hope it will resonate with system architects that are embarking on HP Helion journey.

Our Challenges

Prior to HPCS migration, SimCloud was deployed at HP Labs private cloud managed by Eucalyptus (v1.62). To migrate this SaaS to HPCS, we faced two major challenges.

1. **Addressing computing resources.**

Cloud Connector is a library in Cluster Executor layer (Fig. 2) that provides a set of APIs to enable creation, termination, access and monitoring of the virtual machines and persistent data store in the cloud. This library, written in Python, heavily leveraged Eucalyptus command line interface (euca2ools, [3]) to interact
with the cloud. The compute logic (for instance, fault tolerant features) was convoluted with euca2ools calls. Cloud Connector as is could not interface with OpenStack.

2. Service access authentication and authorization.

HP Labs private cloud is behind the firewall. When deployed in such a trusted environment, SimCloud allowed service access with little to no user authentication due to the fact that we used a network based authentication. This is certainly not adequate when deployed to public open cloud environment such as HPCS.

Our Solution


Neither HPCS OpenStack (v12.12) nor HP Helion (v13.5) supports euca2ools. We decided to not increment existing Cloud Connector codebase but start new because of the positive outcome of our experimentation with Apache JClouds.

JClouds is an open source library that provides API abstractions to interface a large list of public cloud service providers and enables the SaaS solution to be portable across the cloud platforms it supports. JClouds comes with Java APIs that enable managing instances and key-value stores provided by the cloud platform. Our experimentation with JClouds has shown four major advantages:

1. decouple the compute logic with addressing computing resources in Cloud Connector;
2. serve both the Eucalyptus cloud and HP OpenStack cloud with the same Cloud Connector codebase;
3. accelerate the development and minimize the maintenance cost and risk by building on top of widely adopted open source library; and
4. obtain cloud platform extendibility without any extra effort.

During the process of rewriting the Cloud Connector based on JClouds library, we put extra emphasis on separating SimCloud logic from addressing underlying cloud resources to allow minimal effort for any future cloud migrations. A thin layer of utility classes were implemented to map SimCloud compute workflow: from submitting resource requests to the cloud provider, acquiring and preparing VMs, transferring data to VMs through SSH, monitoring VMs and simulation execution, transferring results from VMs to persistent key-value store, and terminating VMs when tasks are completed (or aborted). These classes include:

- **VirtualMachine**: manages a VM. It contains all necessary VM information, for instance, VM id, IP address, security groups’ information, and possible operations such as restart, shutdown and terminate.
- **VirtualMachineManager**: manages all currently active VMs, monitors their status, and provides methods to acquire new VMs from the cloud provider. Once the cloud provider returns the new VM id, it hands this id over to a new VirtualMachine object for accessing all the VM properties.
- **Simulation**: manages a simulation instance. It implements listeners to VM events. A VM registers with the Simulation object by dispatching a VIRTUALMACHINE_READY event.
- **SimulationSession**: manages a simulation cluster. It computes the resources required (e.g., how many VMs) to establish this simulation cluster, and monitors each simulation’s progress and health, and determines when VMs can be safely disposed.
- **SimulationSessionManager**: manages all currently active and non-active (finished) simulation sessions. It manages the lifecycle of each SimulationSession: starting a new simulation cluster launch sequence and reports any anomalies, monitor the launch and simulation progress, and retire the simulation cluster by turning in all computing resources but persistently storing its meta-data.

JClouds is fully compatible with HPCS OpenStack (v12.12). It is compatible with HP Helion with one exception. HP Helion requires the use of the KeyStone API v2 which is not yet enabled by default in
JClouds. In order to enable the use of the KeyStone we need to add a couple of overrides to the connection settings:

- `overrides.setProperty("jclouds.api-version","2");`
- `overrides.setProperty("jclouds.keystone.tenant-id","<TennantId>");` <> denotes a variable.

It took us several trials and conversation with HPCS support engineering to pinpoint this issue and the subsequent fixes.

2. Tiered access, authentication and authorization

As a security feature, HP Helion restricts user interaction with VMs. User interactions with computing resources are defined by security groups. To be compatible to this requirement, tiered security groups (“roles”) were created in SimCloud to enable different classes of interactions with the cloud resource. For instance, VMs that own a TomCat instance (including SimCloud system server) are part of the Servers group. This group has port 80 (HTTP), 8080 (Alt-HTTP) and 22 (SSH) opened for external connections. VMs that have either the simulator engine or the database engine are part of the Simulators group. This group only has port 22 (SSH) opened to allow data transfer in and out of them. This way Simcloud built a line of defense against outside unplanned connections.

Access to the simulation data is explicitly granted. User authentication is performed using encrypted user/password pair stored inside the system database. The authorization is validated (for instance, is the requestor the owner of this simulation) prior to allowing the access to the simulation data. To guarantee a secure environment, we used a session-based authentication in addition to an SSH tunneling for accessing simulation data. SimCloud Server communicates with cloud resources via SSH tunnel. Such communication was implemented using JSch [5]. JSch is a Java secure channel library implementing SSH2. It allowed smooth integration with SimCloud Server codes to enable connections to a SSH server, port forwarding, X11 forwarding, file transfer, and more.

Summary

SimCloud has been successfully running on Helion for over a month. As illustrated by Fig. 2, SimCloud is a typical SaaS application. Even though this technical report describes our migration journey into HP Helion, we hope it provides a user story and a reference point to other software engineering professionals should they contemplate their own HP Helion journey.

We’d like to thank Jun Li (HP Labs), Gary Dispoto (HP Labs) and Narayan Krishnan (HP Labs) for their help and support for this work.

References


