THESIS APPROVAL

GRADUATE SCHOOL, KASETSART UNIVERSITY

Master of Science (Information Technology)

DEGREE

Information Technology — **FIELD**

Computer Engineering — **DEPARTMENT**

TITLE: The Development of a Disaster Recovery Software using Cloud Technology

NAME: Mr. Nutrapong Jariyathanoi

THIS THESIS HAS BEEN ACCEPTED BY

THESIS ADVISOR

(Assistant Professor Putchong Uthayopas, Ph.D.)

THESIS CO-ADVISOR

(Associate Professor Yuen Poovarawan, Ph.D.)

DEPARTMENT HEAD

(Associate Professor Anan Phonphoem, Ph.D.)

APPROVED BY THE GRADUATE SCHOOL ON _____

DEAN

(Associate Professor Gunjana Theeragool, D.Agr.)

THESIS

THE DEVELOPMENT OF A DISASTER RECOVERY SOFTWARE USING CLOUD TECHNOLOGY

NUTRAPONG JARIYATHANOI

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of
Master of Science (Information Technology)
Graduate School, Kasetsart University
2014

Nutrapong Jariyathanoi 2014: The Development of a Disaster Recovery Software using Cloud Technology. Master of Science (Information Technology), Major Field: Information Technology, Department of Computer Engineering. Thesis Advisor: Assistant Professor Putchong Uthayopas, Ph.D. 90 pages.

Disaster recovery is a process involving the implementation of a backup system to support and fix problems stemming from natural disaster or some changes by humans hand that causes the system to be unusable. Currently, many organizations do not set up a proper disaster recovery system due to the very high cost of using commercial solution. In this work, the open source cloud computing system is proposed as a cost effective solution for the building of a disaster recovery (DR) system.

Currently, there are several enterprise companies that provide in the fields of construction and to advise on the issue for creating a system backup through cloud computing such as Amazon EC2 or HP Cloud. Many researcher proposed public research for the preparation of a backup system with the application by cloud computing, which uses the operating system by open source software. The result from any research and experiments suggest that open source software can be applied in the preparation of backup center. This work uses the same approach to create an economical solution to be used in Thailand.

In this work, the use of OpenStack cloud as a disaster recovery solution is described. The system architecture, development and evolution results is presented. The experiences learned can be applied to provide a cost effective solution disaster recovery for many organizations.

_____ _____ __ / __ / ____.

Student's signature Thesis Advisor's signature

ACKNOWLEDGEMENTS

I would like to grateful thank and deeply indebted to Asst. Prof. Dr. Putchong Uthayopas my thesis advisor for advice, encouragement and valuable suggestion for completely in everything of this thesis. I would sincerely like to thank Assoc. Prof. Yuen Poovarawan my thesis co-advisor, and also Dr. Sivadon Chaisiri (penname: javaboom) from Shinawatra University for their valuable comments and suggestion. I gratefully thank TRF administrator, Prof. Dr. Sawasd Tantaratana who introduce my abilities in a cover letter to enter the Master Degree registration and also Asst. Prof. Vudhipong Techadamrongsin who was suggestion me in Information Technology Master Degree topic.

I would like to sincerely thank Mr. Pongsakorn U-chupala for his advice and consultation on the preparation of Openstck test-bed, Mr. Chakkrit Totanee for his help in website developed step. And I would like to greatly indebted to Mr. Phisut Kedpanya, Asst. Prof. Dr. Arnon Rungsawang and Dr. Paruj Ratanaworabhan for his advice. I am heartfelt thanks to my TRF staff and KU-MSIT15 friends for their assistance.

This research was supported all devices and test-bed by The Thailand Research Fund (TRF) cooperated with High Performance Computing and Networking Center (HPCNC), Department of Computer Engineering, Faculty of Engineering, Kasetsart University.

I am especially appreciated my parents, my MFLU-CS3 for their continuing encouragements. Finally, I am deeply appreciated to Miss Sujaree Son-ngay who always devotes time and supported my educated and my work, And also Miss Paemala Udtanut who always gives me heartfelt love during my graduate study.

Nutrapong Jariyathanoi

September 2014

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF ABBREVIATIONS

AD	=	Active Directory
AHCI	=	Advanced Host Controller Interface
API	=	Application Programming Interfaces
AWS	=	Amazon Web Services
BCP	=	Business Continuity Planning
BIA	=	Business Impact Analysis
CAT	=	Category
CPU	=	Central Processing Unit
CRC	=	Cyclic Redundancy Check
DDR	=	Double Data Rate
DHCP	=	Dynamic Host Configuration Protocol
DNS	=	Domain Name System
DR	=	Disaster Recovery
DVD	=	Digital Versatile Disc
EC2	=	Elastic Compute Cloud
FIPS	=	Federal Information Processing Standards
FTP	=	File Transfer Protocol
GB, GBytes	=	Gigabyte
Gbps	=	Gigabits per second
HDD	=	Hard Disk Drive
HP	=	Hewlett-Packard
IaaS	=	Infrastructure as a Service
IDE	=	Integrated Development Environment
IEEE	=	Institute of Electrical and Electronics Engineers
IIS	=	Internet Information Services
IP	=	Internet Protocol
ISO	=	International Organization for Standardization
ISP	=	Internet Service Provider
IT	=	Information Technology
KVM	=	Kernel-based Virtual Machine

LIST OF ABBREVIATIONS (Continued)

L2	=	CPU cache levels 2
LTS	=	Long Term Support
MAC	=	Media Access Control
Mbps	=	Megabit per second
MHz	=	megahertz
NASA	=	National Aeronautics and Space Administration
NIC	=	Network Interface Controller
NIST	=	National Institute of Standards and Technology
OVS	=	Open vSwitch
PaaS	=	Platform as a Service
PKI	=	Public-Key Infrastructure
RAM	=	Random-Access Memory
RPM, rpm	=	Revolutions per minute
RPO	=	Recovery Point Objective
RTO	=	Recovery Time Objective
RW	=	Rewritable
SaaS	=	Software as a Service
SATA	=	Serial Advanced Technology Attachment
SDLC	=	Systems Development Life Cycle
SDRAM	=	Synchronous Dynamic Random-Access Memory
SMBs	=	Small and medium enterprises
SSH	=	Secure Shell
SSI	=	Server Side Includes
SSL	=	Secure Sockets Layer
UDDI	=	Universal Description, Discovery and Integration
UTP	=	Unshielded Twisted Pair
VCPU, vCPU	=	Virtual Central Processing Unit
VM, VMs	=	Virtual Machine(s)

THE DEVELOPMENT OF A DISASTER RECOVERY SOFTWARE USING CLOUD TECHNOLOGY

INTRODUCTION

Disasters from natural are cannot be predicted to occur where and when. Every time when disaster happen, it cause loss and damage to everything in the surrounding area. And then, it not except to the computer system that used as tool for conducting in business activities. And though every day, the computer systems have been designed and developed to solve the problems of damage and corruption data (Jian-hua and Nan, 2011) including the discontinuance of the service. By the way, the problems of clashing or stopping service from disaster are still critical and tremendous impact to business. Meanwhile, setting up a disaster recovery (DR) systems in many companies or organizations are not also popular, because of the high-budget of the recovery system that usually use commercial software, or system software that requires purchasing many license for use in order to build those systems. So many agencies, organizations or companies still had not decided to build a recovery system. Due to cost and money to invest the DR system it's not seems that worth to be paid for.

This work, aims to show that an Open Source Cloud software can be used to set up DR systems. The IT infrastructure is viewed as a virtual infrastructure that consists of many VMs running on a cloud. The Primary-system is used on a daily basis to provide services to users in the organization. As a need to handle the disaster arises, the infrastructure will be migrated to a Backup-system that use the same PKI configuration. The migration is done using secure SSH protocol. In this case, OpenStack (OpenStack Foundation, 2014) cloud computing system running on Ubuntu Cloud (Canonical Ltd., 2014) and Linux version 12.04 LTS is used as a primary and backup system cloud.

As a result, the budget of setting up a Cloud DR systems with Open Source software are decreased, while delivering the same performance.

OBJECTIVES

To building a disaster recovery solution using open source cloud system by developing a set of Application Programming Interface (API) for managing backups system with efficiently and reduce cost from purchasing software license under the organization resources or limit of resources condition. This will let to a much lower total cost of ownership and get the benefits from cloud for organization.

LITERATURE REVIEW

1. Backup and Recovery

Swanson *et al*. (2012) Backup and recovery methods are a means to restore system operations quickly and effectively following a service disruption. The methods should address disruption impacts and allowable downtimes identified in the BIA and should be integrated into the system architecture during the Development/Acquisition phase of the SDLC. A wide variety of recovery approaches may be considered, with the appropriate choice being highly dependent upon the incident, type of system, BIA/FIPS 199 impact level, and the system's operational requirements.

Several alternative approaches should be considered when developing and comparing strategies. Including cost, maximum downtimes, security, recovery priorities, and integration with larger, organization-level contingency plans. Some good guideline is given by Microsoft (Microsoft Co. 2014) as follow.

1 Create a backup and restore plan.
2 Keep a written record of all changes to the system.
3 Implement the following measures to help prevent or minimize the effect of a disaster.
4 Implement fault tolerance into the organization at the hardware or software level.
5 Archive the backup media on a regular basis in a secure location.
6 Verify the integrity of backups and occur without error.
7 Keep identical spare hardware available.
8 Document and test recovery procedures.
9 Train IT staff on disaster recovery procedures.
10 Practice restoring from a backup in a test environment.

1.1 Backup and Recovery Site (Red Hat Inc. 2014)

One of the most important aspects of disaster recovery is to have a location from which the recovery can take place. This location is known as a backup site. In the event of a disaster, a backup site is where your data center will be recreated, and where you will operate from for the length of the disaster.

The following criteria when create a backup site should be considered by Geographic area, Accessibility, Security, Environment, and Cost.

1.1.1 Types of backup sites

a) Cold Sites are typically facilities with adequate space and infrastructure (electric power, telecommunications connections, and environmental controls) to support information system recovery activities.

b) Warm Sites are partially equipped office spaces that contain some or all of the system hardware, software, telecommunications, and power sources.

c) Hot Sites are facilities appropriately sized to support system requirements and configured with the necessary system hardware, supporting infrastructure, and support personnel.

1.2 System recovery

System recovery known as the process and policy implementation from any damage. Including the establishment of backup system to support and fix problems stemming from natural disaster damage or some change by humans hand which affects to the system cannot be used and non-respond (Jian-hua and Nan, 2011).

System recovery from disaster damage are the importance task in the action plan for Business Continuity Planning (BCP) for all organizations (ISO, 2012). By the recovery capability of such, a disaster recovery system can be defined by 2

metrics: Recovery Point objective (RPO) and Recovery Time Objective (RTO). RPO denotes with the possibility of data loss, i.e. how latest data can be recovered in case of disaster. RTO means inter-site takeover overhead, i.e. how soon the business can start again in the secondary site in case of disaster. If the organization prefer small RPO and RTO values, cost and budget used to build the recovery system always high. And the preparation for recovery system from the damage, should also takes into consideration various factors. Including the location, number of employees, equipment and tools, network connections, etc.

Currently, there are many companies that provide in the fields of DR construction and to advise on the issue for creating a system backup through cloud computing. These include, such as Amazon EC2 (Amazon Web Services, Inc., 2014), HP Cloud (Hewlett-Packard Development Company, L.P., 2014), etc. There are many research in the backup systems topic, many of which have embraced the backup system using cloud, i.e. Backup for Cloud and Disaster Recovery for Consumers and SMBs (Javaraiah, 2011). Many technique can be used to improve the performance of the backup system by applying cloud computing. It also can reduce the cost of building a backup system as well. And in OpenStack Spring 2012 Conference, Jorke Odolphi proposed his public research for the preparation of a backup system with the application by cloud computing, which uses the operating system by OpenStack and openSUSE in National ICT Australia Ltd. research lab. The result from research and experiments suggest that open source software can be applied in the preparation of backup center (Odolphi, 2012.).

2. Cloud System

Mell and Grance (2011) National Institute of Standards and Technology (NIST) defines the term of cloud computing as follows: “Cloud computing is a model for enabling convenient, on demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service

provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and tree deployment standard models."

2.1 Essential Characteristics:

2.1.1 On-demand: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

2.1.2 Broad Network Access: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).

2.1.3 Resource Pooling: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.

2.1.4 Rapid Elasticity: Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

2.1.5 Measured Service: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

2.2 Service Models:

2.2.1 Software as a Service (SaaS). The capability of provided to the consumer which used the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email, Examples: Outlook.com, Gmail.com, Yahoo.com), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

2.2.2 Platform as a Service (PaaS). The capability that provided to the consumer by depending onto the cloud infrastructure service with consumer-created or acquired applications created by using programming languages, libraries, services, and tools supported by the provider (i.e. AWS Elastic Beanstalk (Amazon Web Services, Inc., 2014), Google App Engine (Google, Inc., 2014), Windows Azure Development and Test (Microsoft Co., 2014)). The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

2.2.3 Infrastructure as a Service (IaaS). The capability provided to the consumer with the provision processing, storage, networks, and other fundamental computing resources where the consumer was able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but can control the operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., virtual machines, Examples: VMware vCloud (VMware, Inc., 2014), Windows Azure Infrastructure (Microsoft Co., 2014), Google Compute Engine (Google, Inc., 2014), Oracle Infrastructure as a Service (Oracle Corporation, 2014), Rackspace Cloud (Rackspace Ltd,., 2014)).

2.3 Deployment Models (Bakshi, 2011)

2.3.1 Private clouds. Private clouds are typically designed and managed by an IT department within an organization. A private cloud is usually built specifically to provide services internally to an organization. Private clouds may be built in a co-located facility or in an existing data center. This model gives a high level of control over the cloud services and the cloud infrastructure. Examples: VMware vCloud (VMware, Inc., 2014), OpenStack (OpenStack Foundation., 2014), etc.

2.3.2 Public clouds. Public clouds are “stand-alone” or proprietary, mostly offpremise, run by third party companies such as Google, Amazon, and Microsoft. Public clouds are hosted off consumer premises and usually transparently mix applications from different customers on shared infrastructure.

2.3.3 Hybrid clouds. The hybrid cloud infrastructure is a composition of two or more clouds deployment models (private, community, or public). These cloud entities are bound together by standardized or proprietary technology that enables data and application portability and interoperability.

3. OpenStack

OpenStack Foundation (2014) OpenStack is an open-source software for building private and public clouds, which was initially created by the source code contribution from the National Aeronautics and Space Administration of the USA (NASA) and Rackspace, a company providing web hosting and cloud computing services. At the moment, the number of companies and organizations involved in the OpenStack project has grown more than 200 companies, which emphasizes the importance of studying OpenStack and researching how various security issues are handled in this software (OpenStack Foundation., 2014).

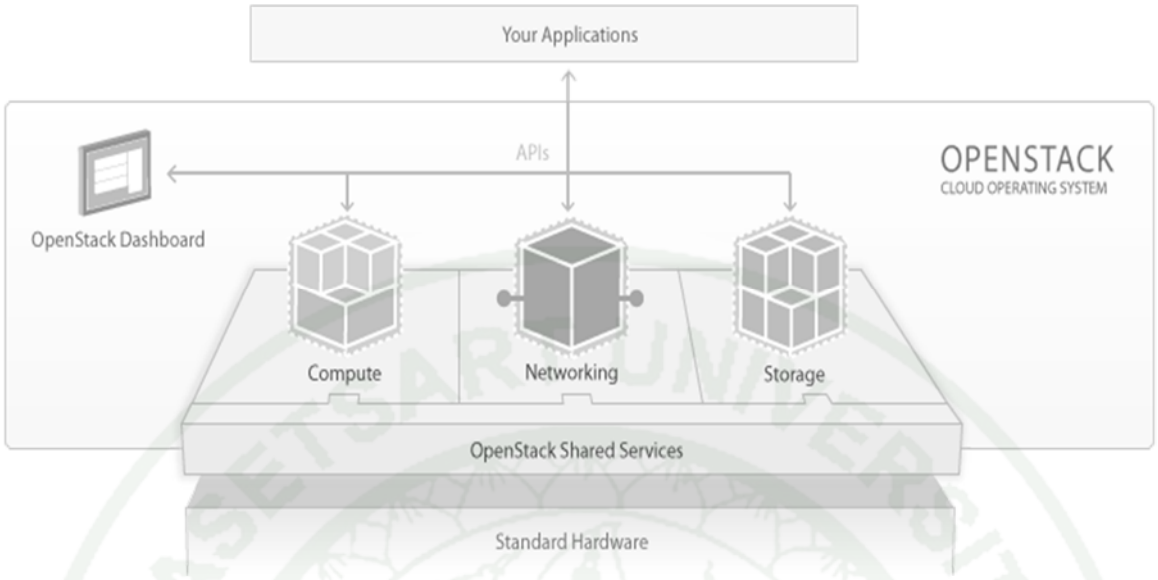

Figure 1 OpenStack architect.

3.1 Architecture :

3.1.1 As a major part of OpenStack, Nova is the computing fabric controller for the OpenStack cloud. It has six components including Nova-API, Message Queue, Nova-Compute, Nova-Network, Nova-Volume and Nova-Scheduler. Nova supports all the life cycles of instances in the cloud, so we can view it as a management platform that manage compute resources, networking, authorization and scalability of needs. All components in the architecture follow a shared-nothing and messaging-based policy, sharednothing means that each component or each group of components can be installed on any server.

3.1.2 Swift, the storage infrastructure in OpenStack is a scalable object storage system. Including a Proxy Server, an Object Server, an Account Server, a Container Server and the Ring. It is a long-term storage system for more permanent type of static data that can be retrieved, leveraged and updated.

3.1.3 The OpenStack image is a solution to build the lookup and retrieval system from virtual machine images, under the name of Glance. Glance-Control and

Glance-Registry are the two parts of Glance service which provides an image service function.

3.1.4 Horizon, which provides a user interface based on browser, makes user easier to manage the cloud services.

3.1.5 Keystone is a cloud identity which provides a common authentication and authorization layer on OpenStack service catalog.

3.1.6 Cinder, provides persistent block-level storage devices for use in OpenStack compute instances. The block storage system manages by creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing users to manage their own needs in cloud storage.

3.1.7 Neutron (formerly known as Quantum) is a system for managing networks and IP addresses. Same as other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing datacenter assets. OpenStack Networking was ensures the network will not have the bottleneck event or limiting of factor in a cloud deployment. And Neutron allows users can manage the network IP with self-service by accept all their own network configurations.

MATERIALS AND METHODS

Materials

1. Hardware

1.1 Four computers

Table 1 Test-bed hardware specification.

Topic	Details
CPU	Intel Core 2 Duo E6550, 2333 MHz (7 x 333), L2 Caches 4096 KBytes
Motherboard	Dell OptiPlex 755
Motherboard Chipset	Intel Bearlake Q35, Socket LGA775
RAM	1 Gbytes DDR2-667 DDR2 SDRAM PC2-5300 (333 MHz)
IDE Controller	Intel(R) ICH9 SATA AHCI Controller
Disk Drive	Western Digital 160 GB, 7200 RPM, SATA-II
Optical Drive	Philips & BenQ Digital Storage DVD+-RW
Network Adapter	Intel(R) 82566 DM-2 Gigabit Network Connection

1.2 Six UTP-CAT 5e type, with data transfer rate at 10/100/1000 Mbps, 200 cm in long

1.3 Two networks switch

Table 2 Summary of networks switch specification.

Hardware/Specification	*SMC FS8* Networks EZ Switch 10/100Mbps	*ASUS GX-D1081* Gigabit Switch 10/100/1000Mbps Power-Saving
Standard	- 10Base-T (IEEE 802.3) - 100Base-TX (IEEE 802.3u) - Flow Control (IEEE 802.3x)	- IEEE 802.3 10Base-T Ethernet - IEEE 802.3u 100Base-TX Fast Ethernet - IEEE 802.3ab 1000BASE-T Gigabit Ethernet - IEEE 802.3az Energy Efficient Ethernet - IEEE 802.3x Flow Control
Port Attributes	8 x 10/100 Mbps Nway RJ-45 ports	8 x 10/100/1000/Gigabits Mbps RJ-45 Ethernet ports
Switching database	2K MAC address entries	Up to 8K MAC address entries
Forwarding Rate	(10Mbps Port): 14,800 pps (100Mbps Port): 148,800 pps	(10Mbps Port): 14,800 pps (100Mbps Port): 148,800 pps (1000Mbps Port): 1,488,000 pps
Transmission method	Store-and-forward architecture filters fragment & CRC error packets	Store-and-forward
Data rate	Supports NWay protocol for speed (10/100 Mbps) and duplex mode (Half/Full) auto-detection	Gigabit Ethernet: 2000 Mbps (Full-duplex)
Buffer memory	768Kbits per device	1000 Kbits
Jumbo frame	-	up to 9.2 KB
EMISSIONS	FCC Class B, CE, VCCI Class B	FCC, CE, IC, C-Tick

2. Software

    2.1 Linux Ubuntu Cloud version 12.04 LTS

    2.2 OpenStack

Methods

1. Experimental Setup

1.1 Hardware architected

For the test-bed in this work, four computers using Intel core 2 duo with CPU 2.33 GHz, Memory 1 Gb, HDD 160 Gb, Gigabit Ethernet and connected via network devices. These 2 systems are separated in order to simulate the primary and backup system. The link between these two systems are experimented at 1 Gbps and 100 Mbps data transfer accordingly. The hardware components installed on each system are summarized in Table 3 and systems architected show in Figure 1.

Table 3 Test-bed hardware details on each system.

Physical Setup	**Hardware Specification**	**Primary-system**		**Backup-system**	
		Controller	Compute	Controller	Compute
CPU	Core 2 Duo E6550, 2333 MHz, L2 Caches 4096 Kbytes	1	1	1	1
RAM	1 Gbytes DDR2-667 DDR2 SDRAM PC2-5300	1	1	1	1
Disk Drive	160 GB, 7200 RPM, SATA-II	2	1	2	1
Network Adapter	Gigabit Network Connection	2	1	2	1
Switch	Networks Switch 10/100 Mbps				
	Gigabit Networks Switch 10/100/1000 Mbps				
Link	UTP CAT 5e	2	1	2	1

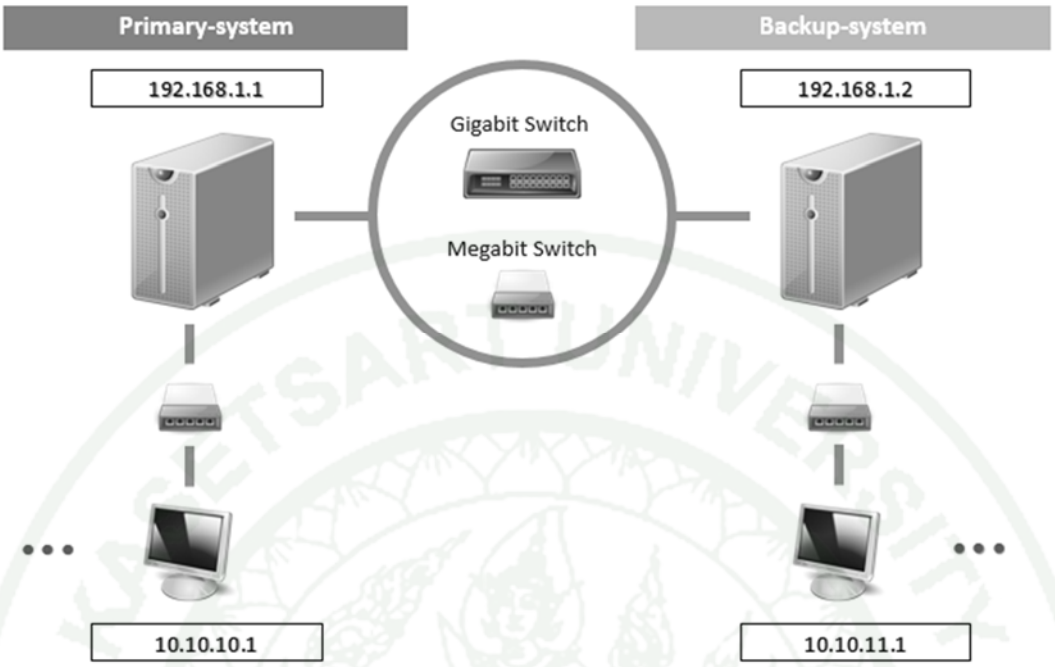

Figure 2 Configuration of the test-bed system.

1.2 Software architected

In this work, the IT infrastructure is viewed as a virtual infrastructure that consists of many VMs running on a cloud. This Primary-system is used on a daily basis to provide services to users in the organization. As a need to handle the disaster arises, the infrastructure will be migrated to Backup-system that use of same the PKI configuration. The migration is done using secure SSH protocol. And then, OpenStack cloud computing system are running on Ubuntu Cloud and Linux version 12.04 LTS is used as a primary and backup system cloud. The cloud components installed on each system are summarized in Table 4.

Table 4 Summary of cloud components.

Controller node	Compute node
Databases (with MySQL) Queues (with RabbitMQ) Keystone Glance Nova (without nova-compute) Cinder Quantum Server (with Open-vSwitch plugin) Dashboard (with Horizon)	Hypervisor (KVM) nova-compute Quantum OVS Agent
Key Authentication	

In the experiment, 5 different types of operating systems have been used to simulate the virtual IT infrastructure. These include the Cirros v0.3.1, Ubuntu Precise v12.04 Mini, CentOS v6. 4 Netinstall, Fedora v19, Ubuntu Precise v12.04. The operating systems summarized show in Table 5.

Table 5 Summary of operating systems types.

Operating System	Data size (MB)
Ubuntu Precise 12.04	1200.00
Fedora 19	689.10
CentOS 6.4 Netinstall	189.10
Ubuntu Precise 12.04 Mini	24.10
Cirros 0.3.1	18.70

All these operating system is configured to use by 2 virtual processors units and 2 GB memories per VM. Table 6 shows the configured of each VM.

Table 6 Test-bed virtualize setup.

Virtual Setup	Primary-system	Backup-system
Virtual Process	2 VCPUs	VCPUs
Virtual Memory	2 GB	2 GB
Virtual Disk	0	0
Virtual Network	2	2
Virtual Router	Public (192.168.1.0/24)	
	Private (10.10.10.0/24)	
Link	2	2

2. Experimental Security Configure

In terms of security management, we created new user to both system with the *sudo* privileges as same as root permissions under the user privilege specification, and used ssh keys with Keychain. For data transferred and replicated section, the researcher used a simple Rsync utilities for synchronized VMs over SSH encryption and secured charnel.

In this work, a set of scripting and tools has been developed to facilitate the migration of VMs. The backup process composed following by these steps:

2.1 Security Confirmation

First step, for every systems must be authorize user. This section will be verified and filtered user privilege level before request, access, or doing everything on the systems. After authorization success, user will connect to Primary-system with each account's permission level via SSH charnel.

2.2 Snapshot and Archive

Second step, user will be suspend and record selected VMs on the main system into image file format. For this process, the researcher recommended to archive the VMs image into a single compress file format for reduce data size and transfer time in migration step.

2.3 Migrate

Third step, for transference process of the VMs collection from Primary-system to the Backup-system through over the network. In this research used Rsync utilities to transfer data between both systems via SSH charnel.

2.4 Unpack and Revive

For final step, user must be unpacking the VMs image compress file on the target system, selected and start up the VMs collection on the backup-system and check data accuracy at last.

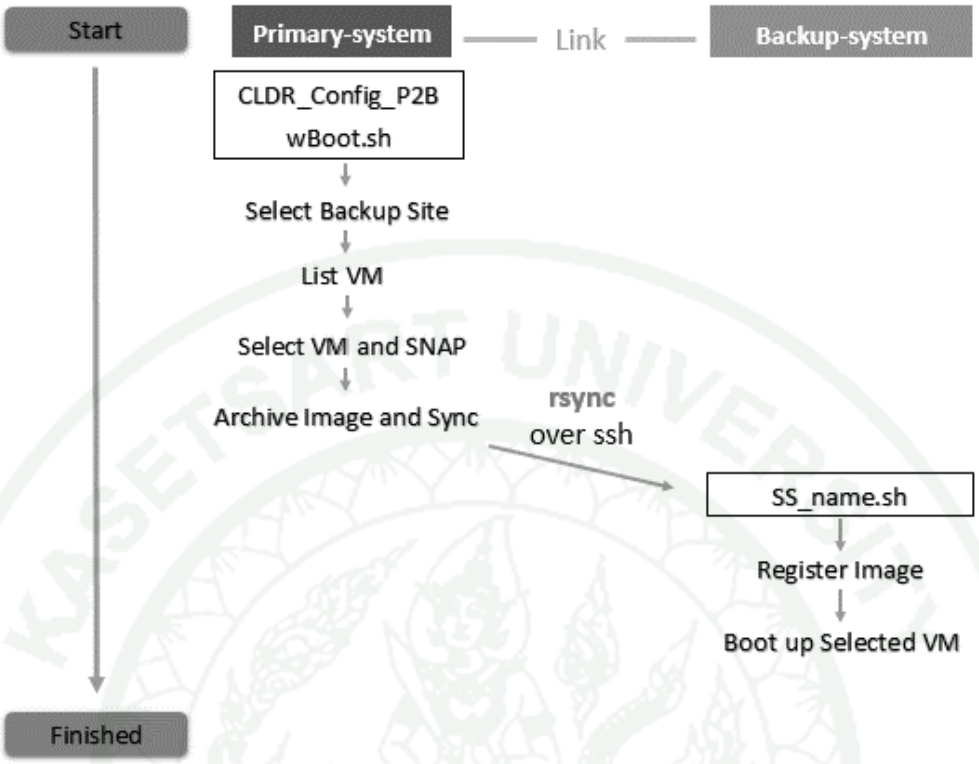

Figure 3 Procedure of the DR-system.

Implication

In this work, the researcher developed web application for help and use to generate shell script. This web application gives CLDR users can generate and export Openstack shell script for Snapshot, Migrate and Revive the VMs on primary and backup system easier.

Figure 4 showing the CLDR's web application login screen. In this step, user always identified them self to get the permission control. And when user unauthorized themselves, used wrong login name or bad password, the web application screen will be show same as Figure 5.

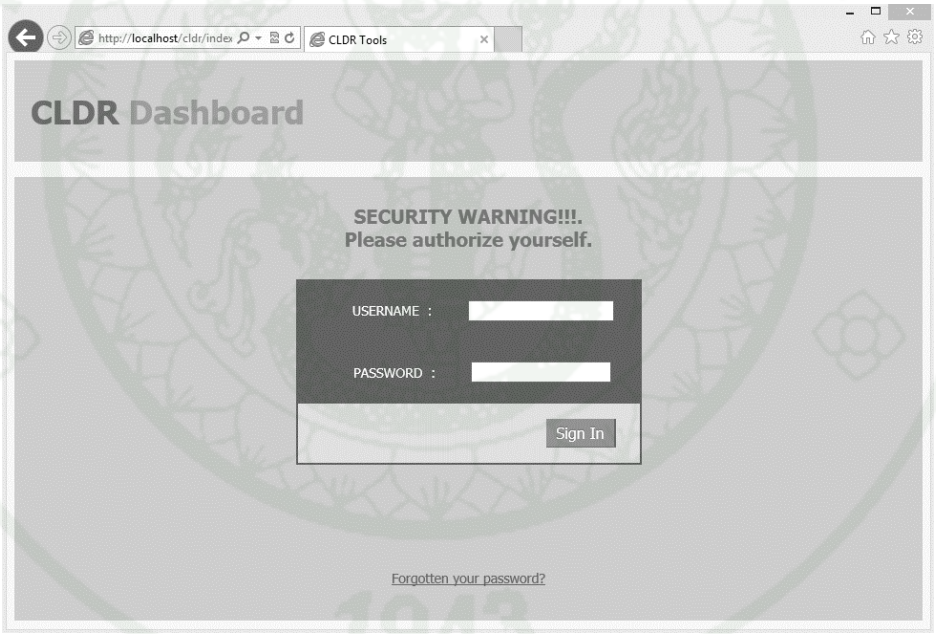

Figure 4 CLDR Login screen.

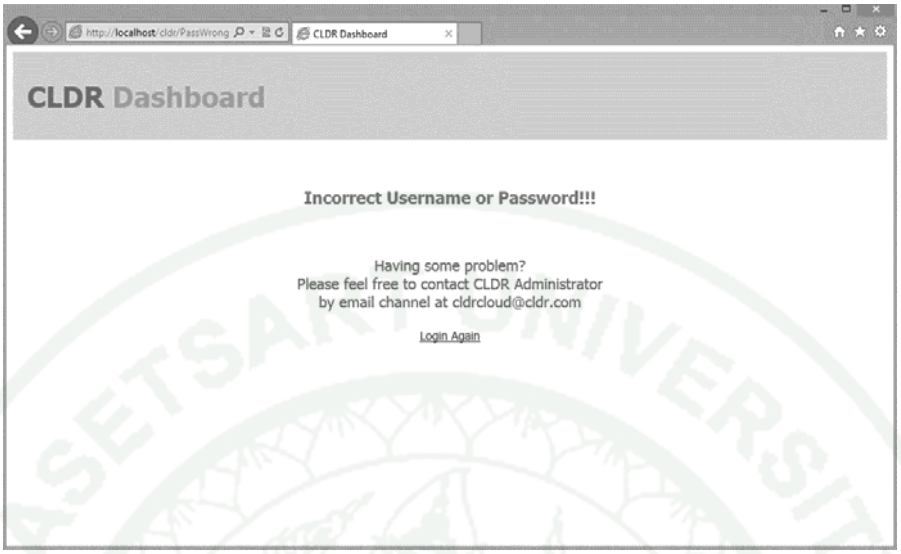

Figure 5 CLDR Login false screen.

Then, if user have the problem with CLDR's web application and click on link: [Forgotten your password?]. The CLDR's web application screen will be appear as Figure 6.

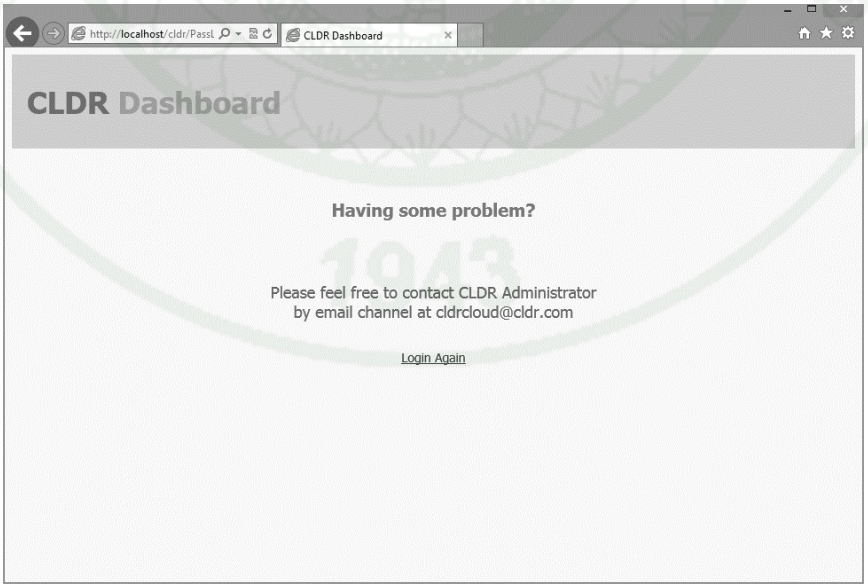

Figure 6 CLDR Contact screen.

CLDR's web application is separate user into two levels,

1. General user can view the summary details of the CLDR system only (Figure 7).

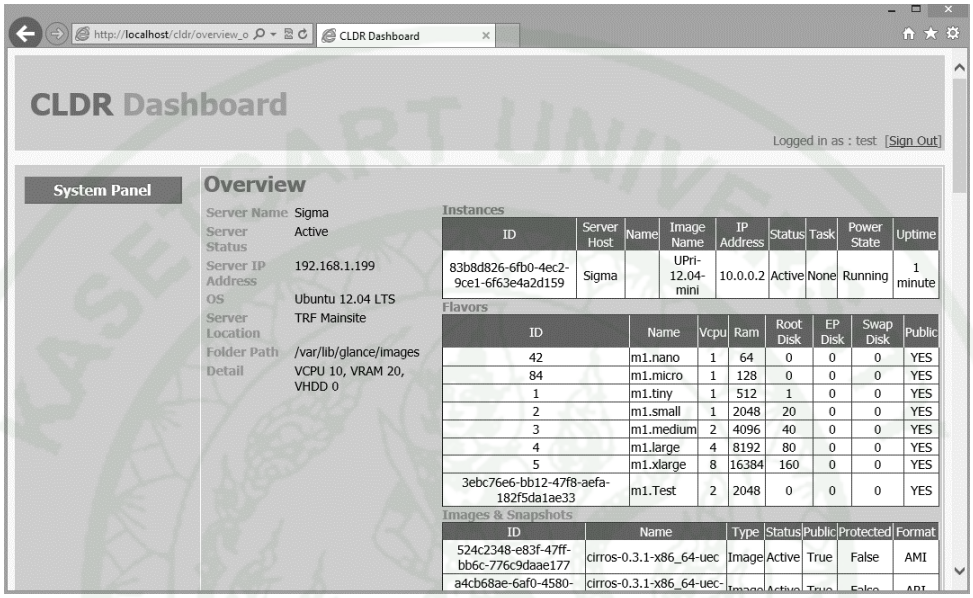

Figure 7 CLDR User view – System panel.

And 2. Administrator, this level can view overall of CLDR system, change, modify and export shell script from the CLDR's web application by click at the topics on the left hand tabs (Figure 8).

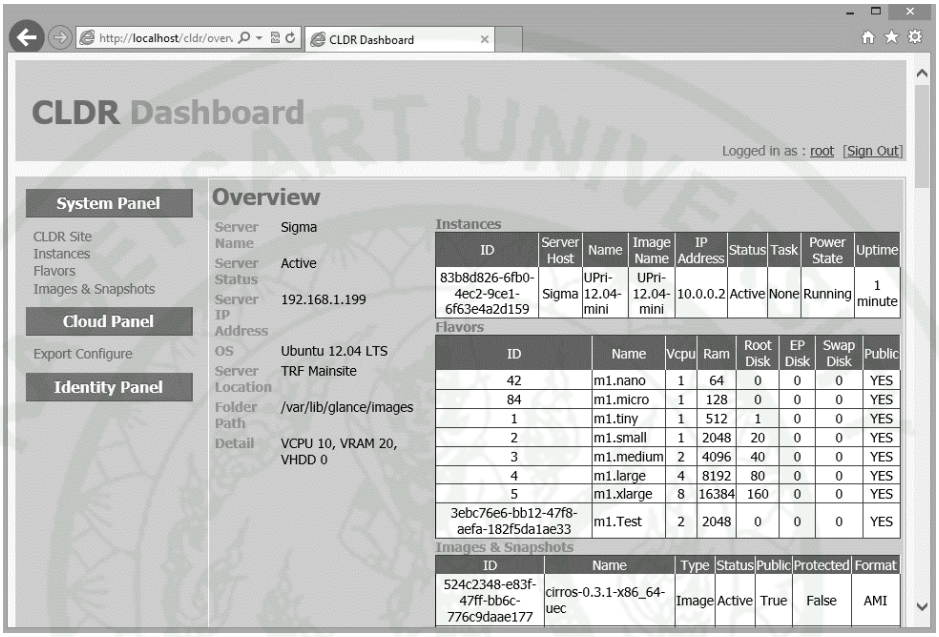

Figure 8 CLDR Administrator view : System panel.

When administrator click at the [CLDR Site] topic, the web application screen will be appear as Figure 9. In this section, the administrator can **Add** the other CLDR site to the CLDR's web application (Figure 10), **Edit** or **Delete** the currently CLDR site (Figure 11).

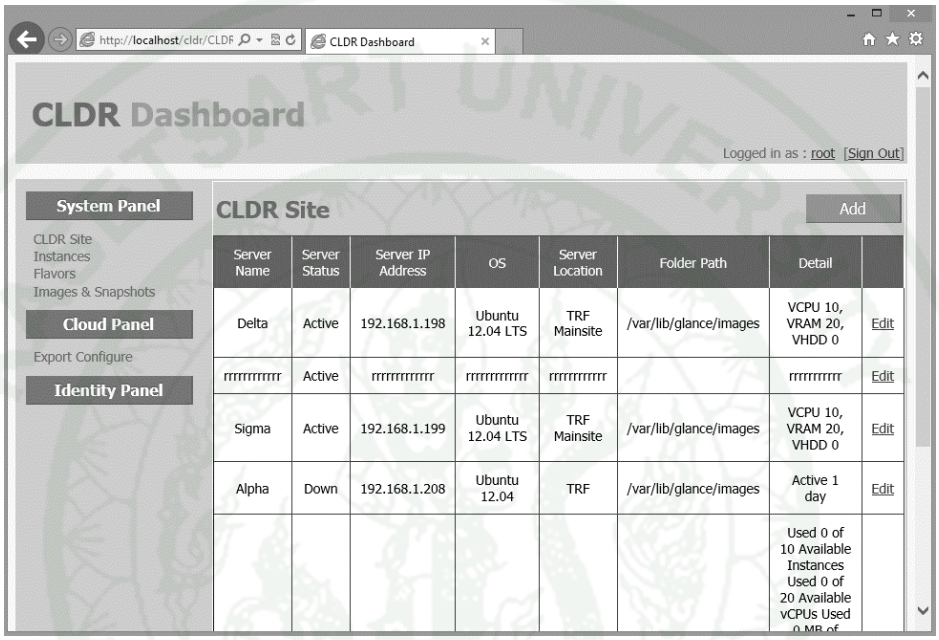

Figure 9 CLDR Administrator view : CLDR site.

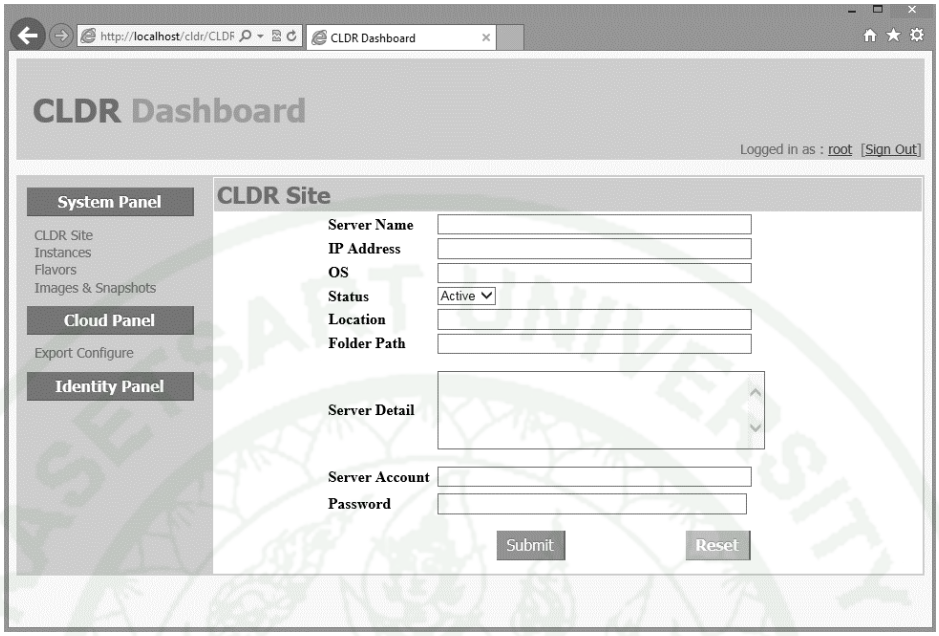

Figure 10 CLDR Administrator view : CLDR site – Add page.

Figure 11 CLDR Administrator view : CLDR site – Edit page.

And when administrator click at the [Instances] topic, the web application screen will be appear as Figure 12. In this section, the administrator can Add the other CLDR's instances (Figure 13), Edit or Delete the instances on currently CLDR's site (Figure 14).

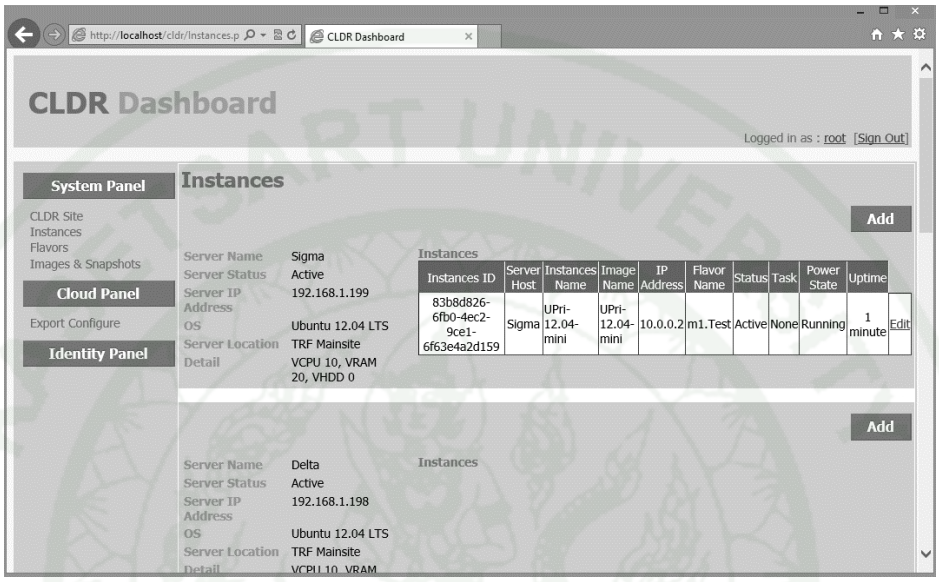

Figure 12 CLDR Administrator view : Instances.

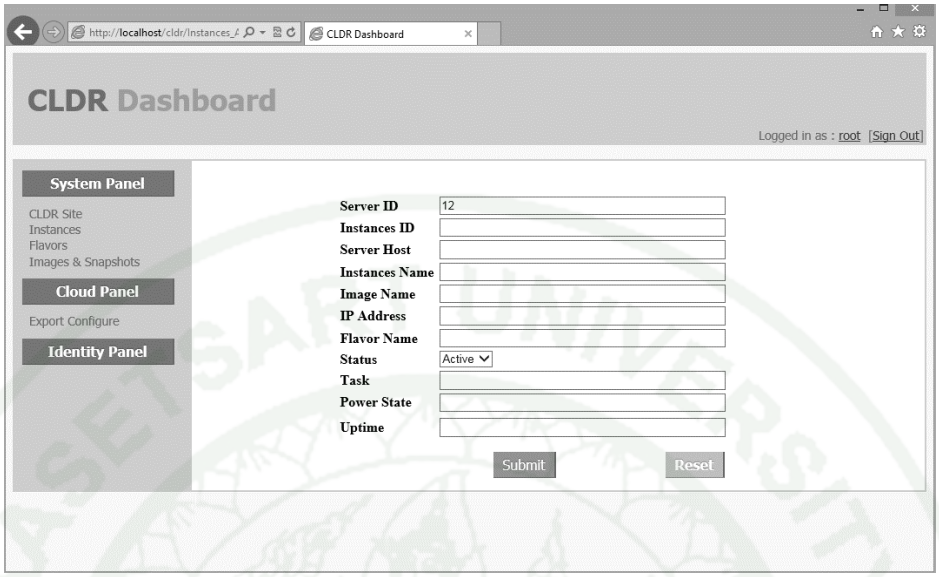

Figure 13 CLDR Administrator view : Instances – Add page.

Figure 14 CLDR Administrator view : Instances – Edit page.

If administrator click at the [Flavors] topic, the web application screen will be appear as Figure 15. In this section, the administrator can Add the other CLDR's flavors (Figure 16), Edit or Delete the flavors on currently CLDR's site (Figure 17).

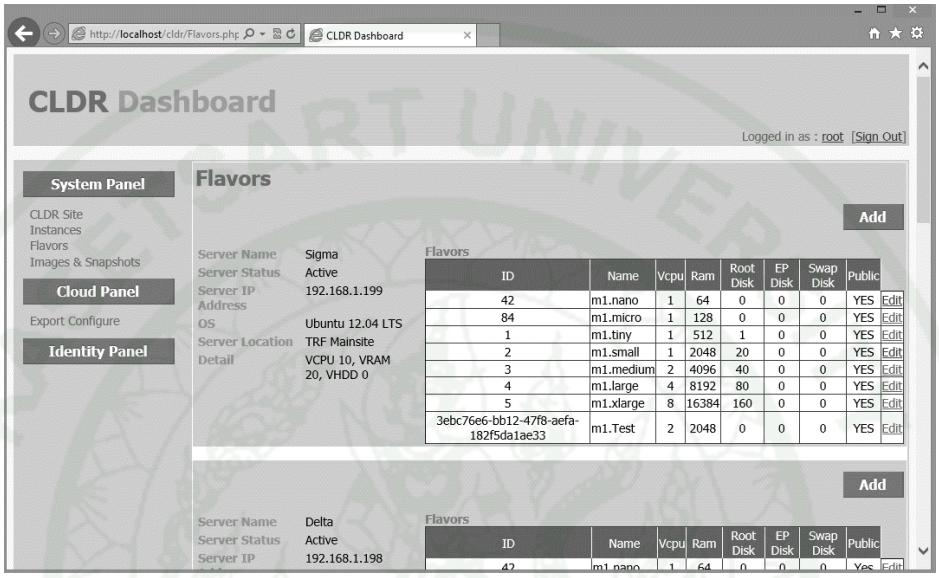

Figure 15 CLDR Administrator view : Flavors.

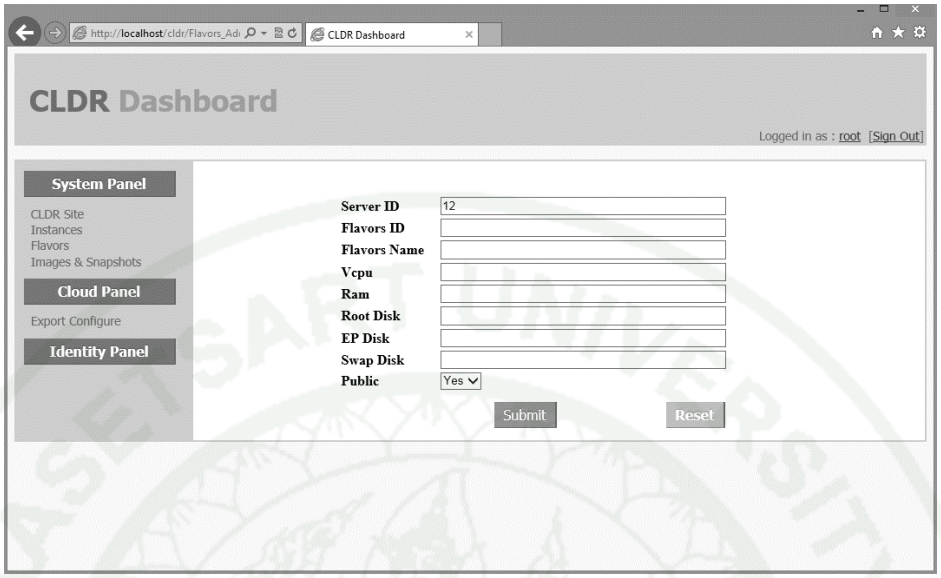

Figure 16 CLDR Administrator view : Flavors – Add page.

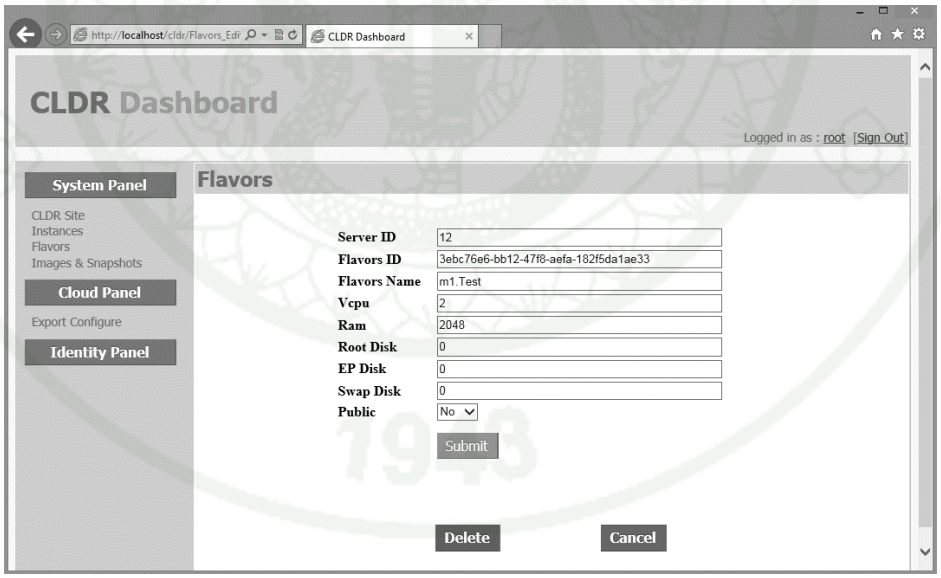

Figure 17 CLDR Administrator view : Flavors – Edit page.

And if administrator click at the [Images & Snapshots] topic, the web application screen will be appear as Figure 18. In this section, the administrator can Add the other CLDR's images or snapshots (Figure 19), Edit or Delete the images or snapshots on currently CLDR's site (Figure 20).

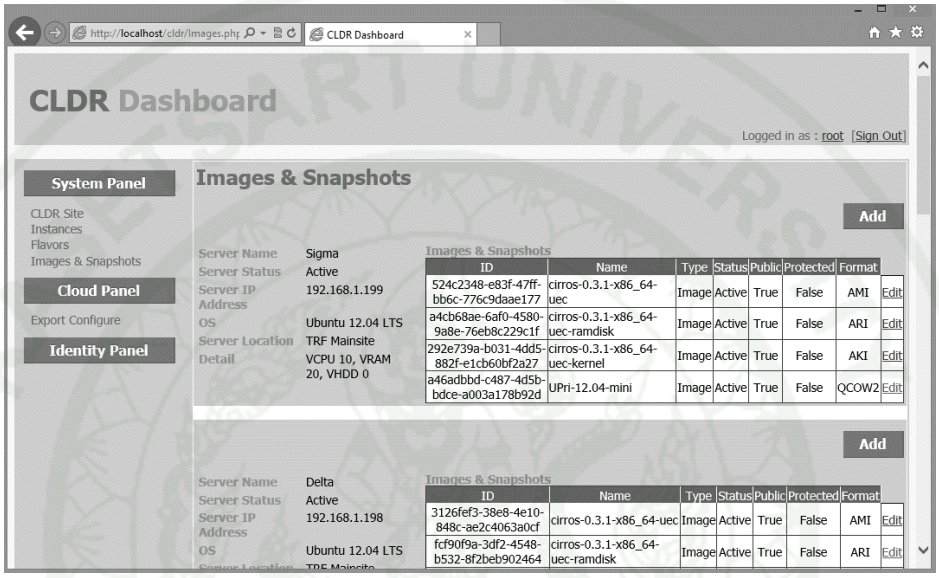

Figure 18 CLDR Administrator view : Images and snapshots.

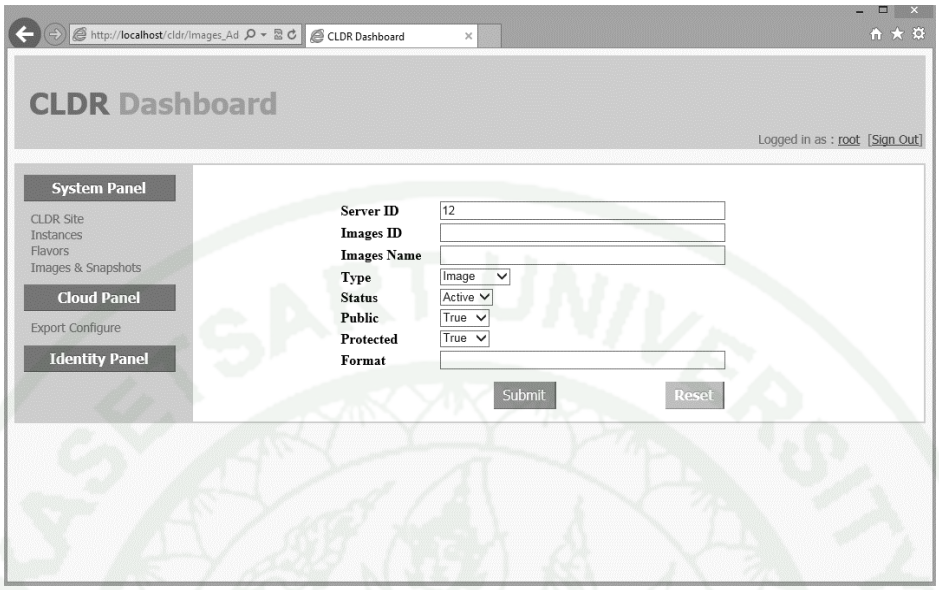

Figure 19 CLDR Administrator view : Images and snapshots – Add page.

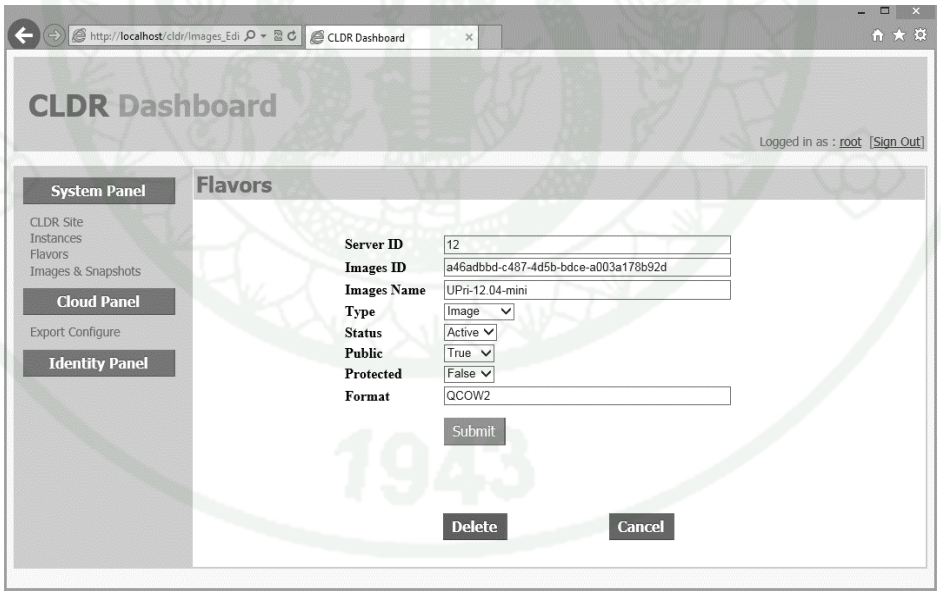

Figure 20 CLDR Administrator view : Image and snapshots – Edit page.

When administrator click at the [Cloud Panel] topic, the web application screen will be appear as Figure 21. In this section, the administrator can Edit the currently shell script on the CLDR's web application (Figure 22).

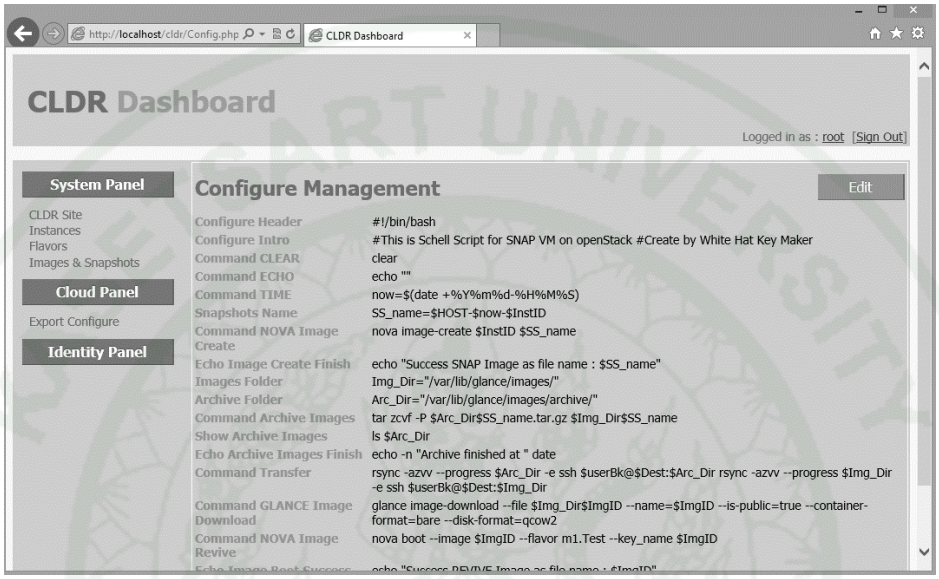

Figure 21 CLDR Administrator view : Configure management.

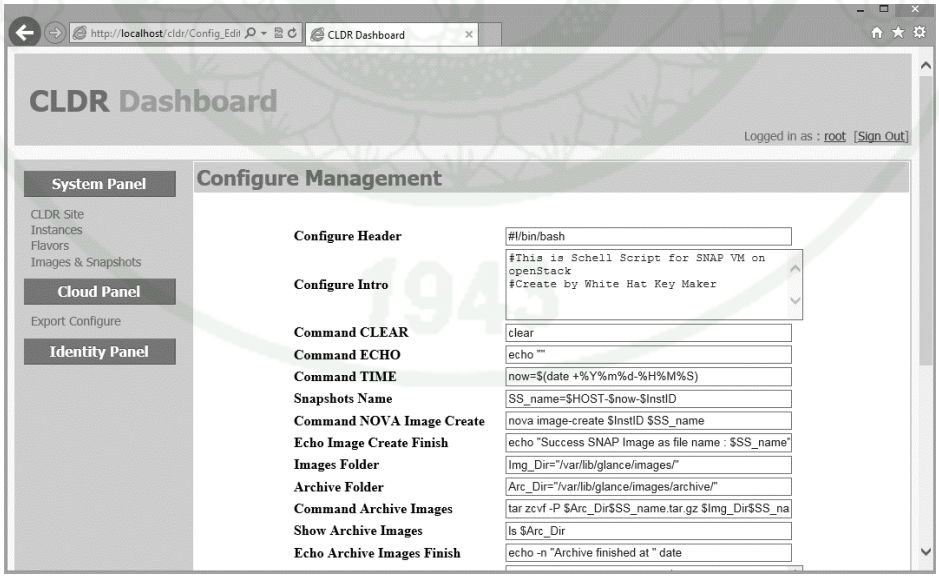

Figure 22 CLDR Administrator view : Configure management – Edit page.

To export shell script from the CLDR's web application, the administrator must be click on the [Export Configure] topic first (Figure 23). And then, choose the Primary CLDR site with select the active instance.

For backup the active instance on the Primary system to the Backup system without revive option. The administrator just only choose the Backup system without check mark on the [Revive after Transfer] and click [Export] on the top-right page. Then, put the shell script on the home path of Primary system (Figure 24). The shell script will be snapshot the instance, archive, and transfer it to the CLDR Backup system folder. But, if the administrator check mark on the [Revive after Transfer] option. The Export Configure page will be appear the flavors table on blank screen. In this step, the administrator must be choose a flavor from the Backup system for use in the revive step. When the shell script generated, administrator must be put the script on the home path both of Primary system and Backup system (Figure 25).

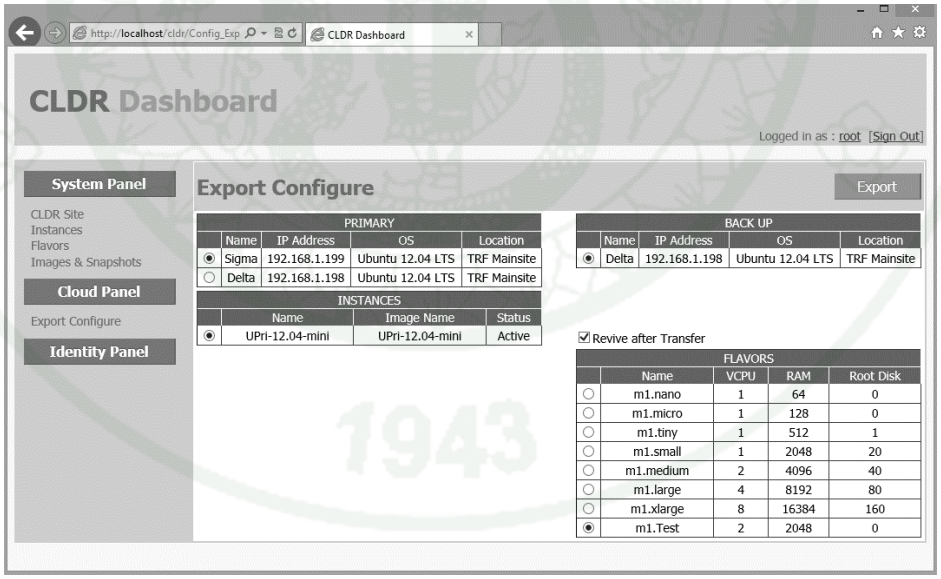

Figure 23 CLDR Administrator view : Configure management – Export page.

Figure 24 CLDR shell script on the Primary system's home path.

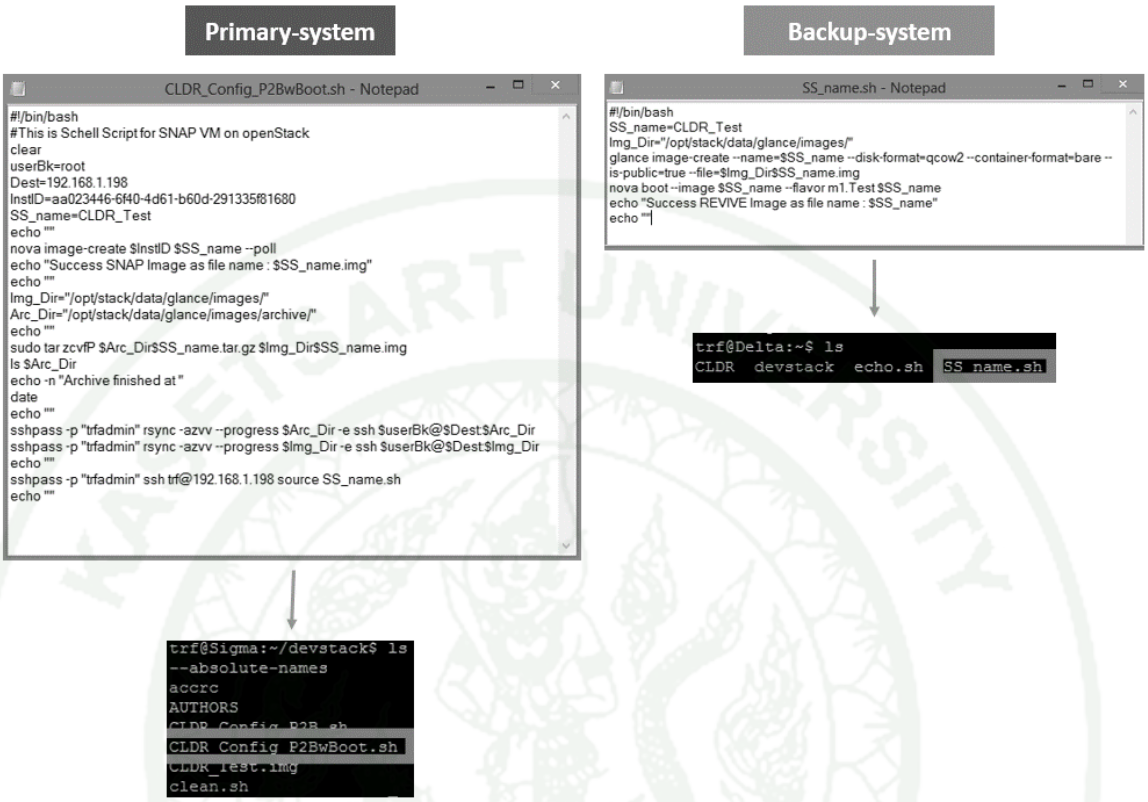

Figure 25 CLDR shell script on the Primary system and Backup system's home path.

At last, if the administrator click at the [Identity Panel] topic, the web application screen will be appear as Figure 26. In this section, the administrator can Add the new CLDR's web application user (Figure 27), Edit information, permission or Delete user on currently CLDR's web application (Figure 28).

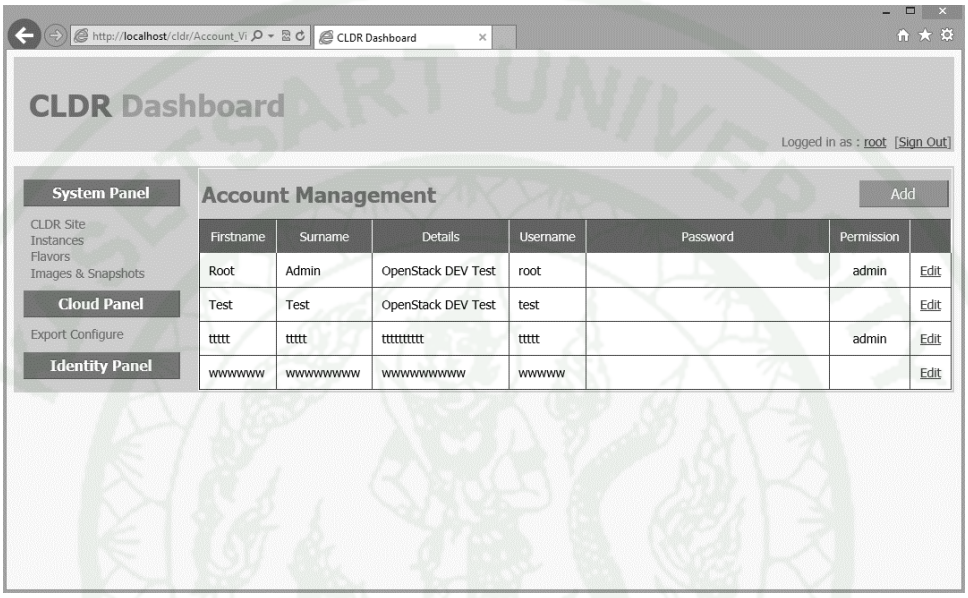

Figure 26 CLDR Administrator view : Account management.

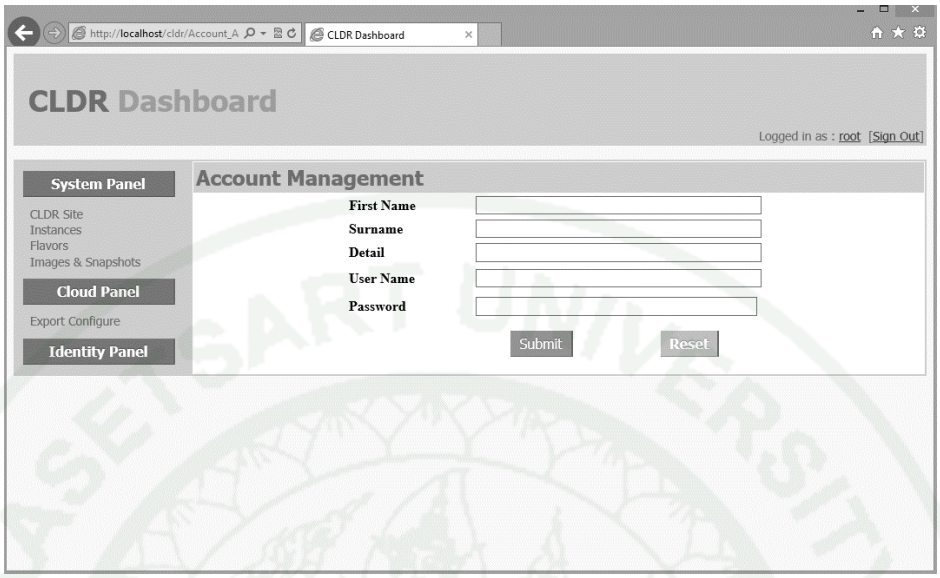

Figure 27 CLDR Administrator view : Account management – Add page.

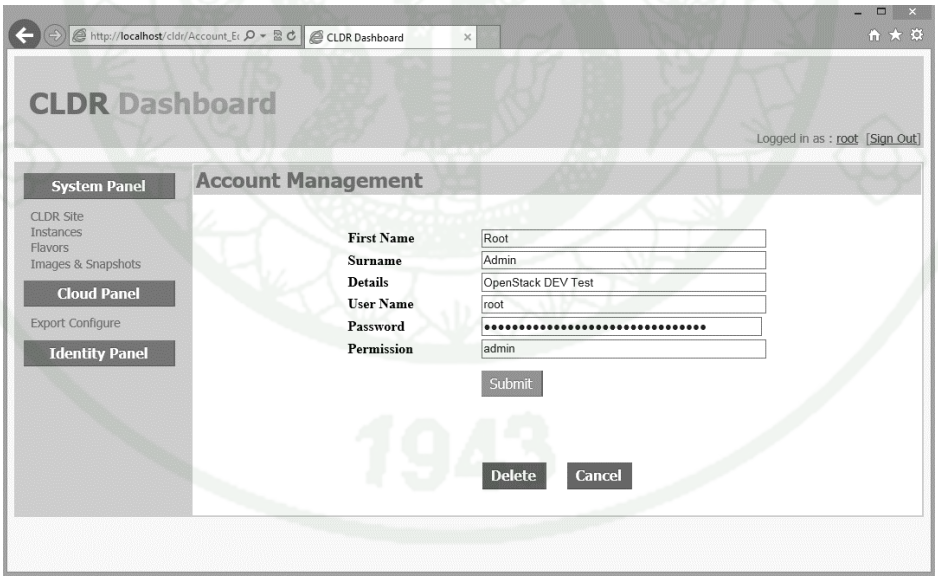

Figure 28 CLDR Administrator view : Account management – Edit page.

RESULTS AND DISCUSSION

Results

In the experiment, for measuring steps in the test-bed system. The experimental measurement the DR systems by using the RTO element, according by the preparation of the backup system to recover from the damage. Which is the main factor that was defined in a business continuity plan by ISO 27001 & ISO 22301.

In order to clear, the experiment were divided the results into 3 different processes (Snap, Transfer and Revive). Which in the Snap and Transfer process, the test has less repeated 5 times per step and used a stopwatch to capture the time. In the process of transferring data, the researcher used the NetPerf v2.4.5 utilities to monitor the performance of the network. The results of the experiments over the 1Gbps network link, as shown in Table 7, 8, 9, and the results from 100 Mbps network link appears in Table 10, 11, 12.

Table 7 Snapshot processing time using 1Gbps link.

Image Name	**Snapshot Time (Sec.)**					
	1	**2**	**3**	**4**	**5**	**Avg.**
Ubuntu Precise 12.04	231.98	239.48	232.36	232.10	240.93	**235.370**
Fedora 19	94.64	94.53	94.45	94.38	94.83	**94.556**
CentOS 6.4 Netinstall	37.58	32.65	37.08	32.10	36.98	**35.278**
Ubuntu Precise 12.04 Mini	12.21	11.90	12.03	11.88	12.28	**12.060**
Cirros 0.3.1	19.85	19.91	19.93	20.10	19.96	**19.950**

Table 8 Startup processing time using 1Gbps link.

Image Name	Startup Time (Sec.)					
	1	**2**	**3**	**4**	**5**	**Avg.**
Ubuntu Precise 12.04	172.60	172.63	172.95	172.80	145.16	**167.228**
Fedora 19	97.68	97.40	97.20	97.40	97.36	**97.408**
CentOS 6.4 Netinstall	37.02	33.13	33.30	35.81	35.73	**34.998**
Ubuntu Precise 12.04 Mini	9.83	9.66	9.76	10.03	9.85	**9.826**
Cirros 0.3.1	14.91	14.86	15.00	15.13	15.53	**15.086**

Table 9 Breakdown processing time using 1Gbps link.

Image Name	Snapshot Avg. Time (Sec.)	Transfer Time (Sec.)	Startup Avg. Time (Sec.)	Total Time (Sec.)
Ubuntu Precise 12.04	235.370	16.0	167.228	**418.598**
Fedora 19	94.556	9.5	97.408	**201.464**
CentOS 6.4 Netinstall	35.278	2.5	34.998	**72.776**
Ubuntu Precise 12.04 Mini	12.060	4.0	9.826	**25.886**
Cirros 0.3.1	19.950	3.0	15.086	**38.036**

Table 10 Snapshot processing time using 100 Mbps link.

Image Name	Snapshot Time (Sec.)					
	1	**2**	**3**	**4**	**5**	**Avg.**
Ubuntu Precise 12.04	239.67	231.64	232.43	232.46	238.75	**234.990**
Fedora 19	94.85	94.54	94.23	94.72	94.57	**94.582**
CentOS 6.4 Netinstall	37.83	37.92	32.70	35.49	38.12	**36.412**
Ubuntu Precise 12.04 Mini	12.32	11.57	11.80	12.38	11.86	**11.986**
Cirros 0.3.1	20.11	20.06	19.89	20.03	19.90	**19.998**

Table 11 Startup processing time using 100 Mbps link.

Image Name	Startup Time (Sec.)					
	1	**2**	**3**	**4**	**5**	**Avg.**
Ubuntu Precise 12.04	172.98	172.40	165.62	171.70	167.71	**170.082**
Fedora 19	97.62	97.65	97.83	97.58	97.23	**97.582**
CentOS 6.4 Netinstall	35.24	37.60	35.36	33.61	34.22	**35.206**
Ubuntu Precise 12.04 Mini	9.90	9.78	9.86	9.89	9.72	**9.830**
Cirros 0.3.1	15.05	14.88	15.11	15.10	14.93	**15.014**

Table 12 Breakdown processing time using 100 Mbps link.

Image Name	Snapshot Avg. Time (Sec.)	Transfer Time (Sec.)	Startup Avg. Time (Sec.)	Total Time (Sec.)
Ubuntu Precise 12.04	234.990	117.8	170.082	**522.872**
Fedora 19	94.582	67.5	97.582	**259.664**
CentOS 6.4 Netinstall	36.412	18.6	35.206	**90.218**
Ubuntu Precise 12.04 Mini	11.986	2.4	9.83	**24.216**
Cirros 0.3.1	19.998	1.9	15.014	**36.012**

To compare the results of the experiments, Figure 3 displayed both of 1Gbps network and 100 Mbps network link results.

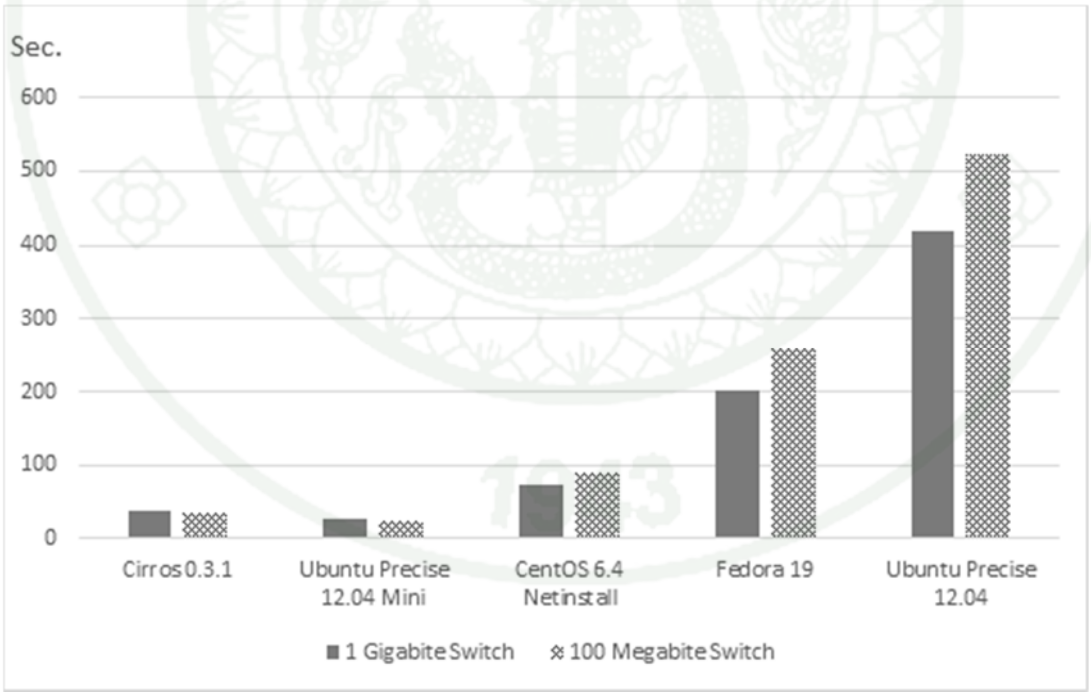

Figure 29 Preparing breakdown processing time of the test-bed.

Discussion

1. The Controller node machine should be setup by hard disks with the superior qualitative and performance than the Compute node. Because, the Controller node's behavior will detect and update the database regularly, although none of any Instant running. Thus, the Controller node machine's hard disks always work hard all time.

For the Compute node machine, if want to be speed up and increased the job process support, the researcher recommended for added more RAM inside. Because, most of all Instant running on the Compute node's memory.

2. When the DR systems does not have any user login via the OpenStack Dashboard for a period of time, or the system have some user logon and did not sign-out. Then, the service that serves to manage this process session will not working correctly, which made the another system process service not functioning properly too. For preliminary approach in this situation, the researcher recommended to restart all OpenStack cloud services together with database service at the same time. Thus, this action will allow the entire services can connect and working correctly as before. However, restarting or shutting down the systems machine could not resolve this problem.

3. Database was originated from the OpenStack process, even if it had export and backup for prepare in the failed case. It cannot used in the reverse process when installing the new system. The recommended way are backup and export the Instant out, and there were used for reverse when installing a new system.

CONCLUSION AND RECOMMENDATIONS

Conclusion

From the experimental and discussion of this study, the conclusion can be drawn as follow:

1. Both of Primary and Backup system, which be installed and applied by using the open source software, into both of the operating system and the cloud service can working with effectively in terms of the completeness of the data. And including the time duration of the backup and recovery process. According by the results that shown in the previous chapter.

In addition, the researcher has developed and added a security to both systems. So that, the DR systems can be separated and divided by the system authorize to improve user security even further .

2. Time duration of backup and transfer data process based on the computer hardware, network devices and networking system performance.

3. Type of utilities for used in data compress, unpack and transmitted process, has affects to the duration of the backup process and transfer data.

4. Type of executed threads that running on the VMs has time affect to progression in Snap and Revive steps.

5. For the budget's experience from this research can compare with other commercial software and cloud service provider appears in Table 13, 14.

Table 13 Open source cloud and Commercial cloud - Cost preparing.

Specifications/ Details	**Ubuntu Cloud with Openstack**	**Microsoft Windows Server 2012 R2 Standard Edition**	**VMWare vSphere 5 Essentials Kit**	**Virtual Private Cloud Service Provider**
Hardware Minimum Requirement	• Dual Quad Core Processor • 8 or 12 GB RAM • Optimized for cost per GB Disk space • 1 GB Network Interface Card (NIC)	• 1.4 GHz 64-bit processor • 512 MB RAM • 32 GB disk space • Gigabit (10/100/1000baseT) Ethernet adapter • DVD drive • Super VGA (1024 x 768) or higher-resolution monitor	• 2 GHz 64-bit processor • 3 MB RAM • 2 GB disk space • Gigabit (10/100/1000baseT) Ethernet adapter	• 2 vCPU • 2 GB vRAM • 80 GB vStorage
Operating System	Ubuntu Cloud 12.04	Microsoft Windows Server 2012 R2 Standard Edition	SUSE Linux Enterprise Server for Vmware	CentOS or Ubuntu
Prices	**FREE**	**USD $882**	**USD $6,673**	**USD $269/Month**

Table 14 Open source cloud and Commercial cloud - Specification preparing.

Specifications / Details	**Ubuntu Cloud with Openstack**	**Microsoft Windows Server 2012 R2 Standard Edition**	**VMWare vSphere 5 Essentials Kit**	**Virtual Private Cloud Service Provider**
Software Components	- Compute service - Image service - Networking service - Object Storage service - Block Storage service - Identity service - Telemetry service - Orchestration service - Dashboard service - Common Libraries service - MySQL - RabbitMQ	- AD Certificate Services - AD Domain Services - AD Lightweight Directory Services - Application Server - DHCP Server - DNS Server - File Services - Hyper-V - Network Policy & Access Services - Print & Doc Services - Remote Desktop Services - UDDI Services - Web Server (IIS) - Windows Deploy Services	- vSphere Hypervisor (ESXi) - vCenter Server Essentials - vCenter Operations Manager Foundation - vSphere Data Protection - vSphere High Availability (HA) - vSphere vMotion - vSphere vShield Endpoint - vSphere Replication - vSphere Storage Appliance	- Parallels Plesk - Quantity of Email Account - Throughput Limit - FTP Connection - Quantity of IP Address - SSH - SSL - Parl - PHP - SSI - MySQL (phpMyAdmin) - vmHA - DRS
Source	http://docs.openstack.org/trunk/install-guide/install/yum/content/index.html	http://www.microsoft.com/en-in/server-cloud/products/windows-server-2012-r2/default.aspx	http://www.vmware.com/products/vsphere/	http://www.private-cloud.asia/en/

Recommendations

From the results, one can see that the snapshot and reviving time is the major factors that contribute to the performance of the DR system. Thus, using this solution over a WAN used in Thailand seems to be practical. Nevertheless, one of the complicated matters is the IP assignment since it is the key to make infrastructure usable from the new site. Since, the primary site is assumed to be out of service at the time of disaster, there must be a mechanism to provide the same IP across the site. Therefore, we have suggestions on how to fix by 2 methods:

1. Use Dynamic DNS service.

Dynamic DNS also be explored as a solution to enhance the usability of the system after migration. This solution will store an IP address by domain name of computer that was registered. Even if IP address is not static, but the system can use domain name instead of IP address.

2. VLan service from provider or centralize DHCP from secure site.

Using VLan service from ISP (CS Loxinfo Co., Ltd.. 2014) allows broadcast domain boundaries in the backup system, which can solve the problem of losing path to send data by using some kind of across site.

At last, The backup service should be process by the periodically schedule to ensure the availability of the main system after disaster. That’s mean, the latest virtual infrastructure can be launch by using the latest backup if the transport of last state of VMs do not work at correctly.

SUMMARY AND FUTURE WORK

Summary

From the conclusion, if we are compare the DR system from Open source cloud versus Commercial cloud that release in the market under the organization resources or limit of resources condition. We found the DR system in this research can used for replace the Commercial cloud as well in term of providing, stability and integrity of data. However, it depends on the expertise and experience of system administrator.

And then, if we compare Open source cloud and Commercial cloud which more clearly in terms of cost analysis, the simple result appears in Table 15 and 16.

Table 15 Commercial cloud Cost Analysis.

Cost per Unit						
	Unit Label	**Unit Quantity**	**One-Time***	**Annual Recurring***	**3-Year Total**	**Comments**
Hardware	**Server**	2	฿ 200,000	฿ -	฿ 200,000	
		License Quantity	**One-Time (License)**	**Annual Recurring (Main-tenance)**	**3-Year Total**	**Comments**
Software	**OS (MS Windows 2010 R2)**	2	฿ 48,400	฿ -	฿ 48,400	
	Database (MS SQL 2012)	2	฿ 43,320	฿ -	฿ 43,320	
	Cloud Engine (VMware vCloud Suite Standard)	1	฿ 164,835	฿ 41,217	฿ 247,269	
Unit Quantity						
		Quantity	**One-Time**	**Cost per Unit**	**3-Year Total**	**Comments**
IT Labor	**System Engineer/System Administrator**	FTEs**	฿ -	฿ 360,000	฿ 1,080,000	
	Employee Trained Cost per Class	1	฿ 30,000	฿ 20,000	฿ 90,000	
	Consulting Services (Hour)	8/week	฿ -	฿ 120,000	฿ 360,000	
	Outsourcing (Hour)	48	฿ 30,000	฿ -	฿ 30,000	
		Quantity	**One Time**	**Annual Recurring**	**3-Year Total**	**Comments**
Reduce Power /Electricity Usage Costs	**Power/Electricity Usage per Server (1400w/24h/d)**	2	฿ -	฿ 92,840	฿ 278,520	
		Quantity	**One-Time Costs Avoided**	**Annual Costs Avoided**	**3-Year Total**	**Comments**
Other IT Cost Avoidance /Reductions	**Reduce Bandwidth Costs**	1	฿ -	฿ 120,000	฿ 360,000	
	Reduce Other Facilities Costs		฿ -	฿ 120,000	฿ 360,000	
	Other		฿ -	฿ 20,000	฿ 60,000	
Total			**฿ 516,555**	**฿ 894,057**	**฿ 3,157,509**	

Table 16 Open source cloud Cost Analysis.

Cost per Unit						
	Unit Label	**Unit Quantity**	**One-Time***	**Annual Recurring***	**3-Year Total**	**Comments**
Hardware	**PC**	4	฿ 20,000	฿ -	฿ 60,000	
		License Quantity	**One-Time (License)**	**Annual Recurring (Main-tenance)**	**3-Year Total**	**Comments**
Software	**OS (Ubuntu 12.04)**	4	฿ -	฿ -	฿ -	
	Database (MySQL)	2	฿ -	฿ -	฿ -	
	Cloud Engine (Openstack)	2	฿ -	฿ -	฿ -	
Unit Quantity						
		Quantity	**One-Time**	**Cost per Unit**	**3-Year Total**	**Comments**
IT Labor	**System Engineer/System Administrator**	FTEs**	฿ -	฿ 420,000	฿ 1,260,000	
	Employee Trained Cost per Class	1	฿ 30,000	฿ 25,000	฿ 105,000	
	Consulting Services (Hour)	8/week	฿ -	฿ 120,000	฿ 360,000	
	Outsourcing (Hour)	48	฿ 30,000	฿ -	฿ 30,000	
		Quantity	**One Time**	**Annual Recurring**	**3-Year Total**	**Comments**
Reduce Power /Electricity Usage Costs	**Power/Electricity Usage per Server (1400w/24h/d)**	4	฿ -	฿ 33,160	฿ 99,480	
		Quantity	**One-Time Costs Avoided**	**Annual Costs Avoided**	**3-Year Total**	**Comments**
Other IT Cost Avoidance /Reductions	**Reduce Bandwidth Costs**	1	฿ -	฿ 120,000	฿ 360,000	
	Reduce Other Facilities Costs		฿ -	฿ 120,000	฿ 360,000	
	Other		฿ -	฿ 20,000	฿ 60,000	
Total			**฿ 80,000**	**฿ 858,160**	**฿ 2,694,480**	

* One-Time costs are typically incurred at the beginning of the project. Annual Recurring costs are on-going costs necessary throughout the life of the project.

**FTE (Full Time Equivalent): 1 full time employee is equivalent to 1 employees each dedicating 100% of their work time to the task

Price reference source :

http://www.asis.co.th/, 11 November 2014.

http://www.vmware.com/products/vcloud-suite/pricing, 11 November 2014.

Exchange Rate 33 THB = 1 USD

source :

http://www.bot.or.th/Thai/Statistics/FinancialMarkets/ExchangeRate/_layouts/Application/ExchangeRate/ExchangeRa te.aspx, 11 November 2014.

Power Rate 3.785 THB/Unit

Source :

http://www.mea.or.th/profile/index.php?tid=3&mid=111&pid=109, 11 November 2014.

Future work

The next process for further on this research, the researcher emphasize on the backup VMs step. This process should enhanced the snapshot ability at the application level. This step should have the ability to store, retrieve and manage it easily. Similar to the Apple's Time Machine logic (Simenstad, 2014).

At the software development process, the reproduce software should be developed further to improve the automation ability like the thin client agents. This agents should be easy to control and handling the DR system, have the characteristics in term of the Centralized management control. At the end, it should have the ability to workable at the DR-as-a-service level which handle the DR site and easily to manage the VMs (Hewlett-Packard Development Company, L.P., 2014).

LITERATURE CITED

Jian-hua, Z. and Z. Nan. 2011. Cloud Computing-based Data Storage and Disaster Recovery. pp 629-632. *In* **2011 int. conf. on Future Computer Science and Education (ICFCSE)**. 20-21 August 2011, Xi'an, China.

Swanson, M., P. Bowen, A. W. Phillips, D. Gallup and D. Lynes. 2010. Contingency Planning Guide for Federal Information Systems. **NIST Special Publication 800-34 Rev.** 1: 20-21.

Microsoft Co., Ltd. 2014. **Best Practices for Disaster Recovery**. Available Source: http://msdn.microsoft.com/en-us/library/gg634596.aspx, January 03, 2014.

Red Hat, Inc. 2014. **Introduction To System Administration**. Available Source: https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/4/html/Introduction_To_System_Administration/s2-disaster-recovery-sites.html, January 03, 2014.

International Standards Organization (ISO) and the International Electro technical Commission (IEC). 2012. **ISO 22301:2012 - Business continuity management systems**. Available Source: http://www.iso.org/iso/catalogue_detail?csnumber=50038, April 12, 2013.

Amazon Web Services, Inc. 2014. **Disaster Recovery**. Available Source: https://aws.amazon.com/disaster-recovery, January 03, 2014.

Hewlett-Packard Development Company, L.P. 2014. **Disaster Recovery with HP Cloud Object Storage**. Available Source: http://www.hpcloud.com/solutions/disaster-recovery, January 03, 2014.

Javaraiah, V. 2011. Backup for cloud and disaster recovery for consumers and SMBs. pp 1-3. *In* **2011 5th IEEE int. conf. on Advanced Networks and Telecommunication Systems (ANTS)**. 18-21 December 2011, TBD, Bangalore, Karnataka, India.

Odolphi, J. 2012. Building a Disaster Recovery Solution Using OpenStack. pp. 109-114. *In* **Openstack Conference Spring 2012**. 16-20 April 2012, Hyatt Regency San Francisco, San Francisco, United States.

Bakshi, K. 2011. Considerations for cloud data centers: Framework, architecture and adoption. pp. 1-7. *In* **IEEE Aerospace Conference 2011**. 05-12 March 2011, Big Sky Resort, Big Sky, MT, USA.

Mell, P. and T. Grance. 2011. NIST Definition of Cloud Computing. **NIST Special Publication 800-145**: 1-3.

Amazon Web Services, Inc. 2014. **AWS Elastic Beanstalk**. Available Source: https://aws.amazon.com/elasticbeanstalk, January 03, 2014.

Google, Inc. 2014. **Google App Engine**. Available Source: https://cloud.google.com/products/app-engine, January 03, 2014.

Microsoft Co., Ltd. 2014. **Development and Test Using Virtual Machines**. Available Source: http://www.windowsazure.com/en-us/solutions/dev-test, January 03, 2014.

Microsoft Co., Ltd. 2014. **Infrastructure Services**. Available Source: http://www.windowsazure.com/en-us/solutions/infrastructure, January 03, 2014.

Google, Inc. 2014. **Compute Engine**. Available Source: https://cloud.google.com/products/compute-engine, January 03, 2014.

VMware, Inc. 2014. **vCloud Suite**. Available Source: http://www.vmware.com/products/vcloud-suite, January 03, 2014.

Oracle Corporation. 2014. **Oracle Infrastructure as a Service (IaaS)**. Available Source: http://www.oracle.com/us/products/engineered-systems/iaas/overview/index.html, January 03, 2014.

Rackspace Ltd,. 2014. **Public Cloud Hosting, Computing, Storage & Networking**. Available Source: http://www.rackspace.com/cloud, January 03, 2014.

OpenStack Foundation. 2014. **Getting Started with OpenStack**. Available Source: https://wiki.openstack.org/wiki/Getting_Started, January 03, 2014.

OpenStack Foundation. 2014. **Companies Supporting The OpenStack Foundation**. Available Source: http://www.openstack.org/ foundation/ companies, January 03, 2014.

Microsoft Co., Ltd. 2014. **Windows Server 2012 R2**. Available Source: http://www.microsoft.com/en-in/server-cloud/products/windows-server-2012-r2/default.aspx, March 28, 2014.

VMware, Inc. 2014. **vSphere**. Available Source: http://www.vmware.com/products/vsphere/, March 28, 2014.

Crescere (Thailand) Co., Ltd. 2014. **VIRTUAL PRIVATE CLOUD ASIA**. Available Source: http://www.private-cloud.asia/en/, March 28, 2014.

Fifield, T., D. Fleming, A. Gentle, L. Hochstein, J. Proulx, E. Toews and J. Topjian. 2014. **OpenStack Operations Guide**. First Edition. O'Reilly Media, Inc., USA.

Simenstad, J. 2014. **Understanding Apple's Time Machine**. Available Source: http://icanhelpyourmac.com/tip_archive/understanding_apples_time_machine/index.html, November 11, 2014.

Hewlett-Packard Development Company, L.P. 2014. **Open for business - Ins and outs of enterprise-class cloud continuity**. Available Source: http://h20195.www2.hp.com/v2/GetPDF.aspx/4aa4-1570enw.pdf, November 11, 2014.

APPENDIX

CONTROLLER AND COMPUTE NODE SETUP

Controller Node Setup

1. Install Ubuntu Cloud 12.04 on controller node.

2. Install OpenStack cloud service : *apt-get install ubuntu-cloud-keyring*.

3. Configure the network by edit **/etc/network/interfaces**

Primary-system

Internal Network
auto eth0
iface eth0 inet static
address 10.10.10.254
netmask 255.255.255.0

External Network
auto eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
gateway 192.168.1.254
dns-nameservers 8.8.8.8

Backup-system

```
# Internal Network
auto eth0
iface eth0 inet static
address 10.10.11.254
netmask 255.255.255.0

# External Network
auto eth1
iface eth1 inet static
address 192.168.1.2
netmask 255.255.255.0
gateway 192.168.1.254
dns-nameservers 8.8.8.8
```

4. Install the database packages : *apt-get install -y python-mysqldb mysql-server*

5. Configure MySQL accept connections access from the compute nodes : *sed -i 's/127.0.0.1/0.0.0.0/g' /etc/mysql/my.cnf*

6. Create database table

```
mysql -u root -p <<EOF
CREATE DATABASE nova;
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
IDENTIFIED BY 'password';
CREATE DATABASE cinder;
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \
IDENTIFIED BY 'password';
CREATE DATABASE glance;
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
```

IDENTIFIED BY 'password';
CREATE DATABASE keystone;
GRANT ALL PRIVILEGES ON keystone. TO 'keystone'@'localhost' *
IDENTIFIED BY 'password';
CREATE DATABASE quantum;
GRANT ALL PRIVILEGES ON quantum. TO 'quantum'@'localhost' *
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON quantum. TO 'quantum'@'localhost' *
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON quantum. TO 'quantum'@'localhost' *
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
EOF

7. Install RabbitMQ Messaging Service : *apt-get install -y rabbitmq-server* and change the default password by : *rabbitmqctl change_password guest password*

8. install OpenStack Identity Service : *apt-get install -y keystone python-keystone python-keystoneclient*

And edit **/etc/keystone/keystone.conf**

[DEFAULT]
admin_token = password
debug = True
verbose = True

[sql]
connection = mysql://keystone:password@localhost/keystone

9. Create an *~/openrc* File and insert those command line

export OS_TENANT_NAME=admin

export OS_USERNAME=admin

export OS_PASSWORD=password

export OS_AUTH_URL="http://localhost:5000/v2.0/"

export SERVICE_ENDPOINT="http://localhost:35357/v2.0"

export SERVICE_TOKEN=password

10. Install Keystone services by bash script with some initial data:

- Projects: admin and services
- Roles: admin, Member
- Users: admin, demo, nova, glance, quantum, and cinder
- Services: compute, volume, image, identity, ec2, and network

#!/bin/bash

 # Modify these variables as needed

ADMIN_PASSWORD=${ADMIN_PASSWORD:-password}

SERVICE_PASSWORD=${SERVICE_PASSWORD:-
$ADMIN_PASSWORD}

DEMO_PASSWORD=${DEMO_PASSWORD:-
$ADMIN_PASSWORD}

export OS_SERVICE_TOKEN="password"

export OS_SERVICE_ENDPOINT="http://localhost:35357/v2.0"

SERVICE_TENANT_NAME=${SERVICE_TENANT_NAME:-service}

#

MYSQL_USER=keystone

MYSQL_DATABASE=keystone

MYSQL_HOST=localhost

MYSQL_PASSWORD=password

#

KEYSTONE_REGION=RegionOne

KEYSTONE_HOST=10.10.10.254

 # Shortcut function to get a newly generated ID

```
function get_field() {
while read data; do
if [ "$1" -lt 0 ]; then
field="(\$(NF$1))"
else
field="\$$(($1 + 1))"
fi
echo "$data" | awk -F'[ \t]*\|[ \t]*' "{print $field}"
done
}
# Tenants
ADMIN_TENANT=$(keystone tenant-create --name=admin | grep " id
" | get_field
2)
DEMO_TENANT=$(keystone tenant-create --name=demo | grep " id "
| get_field
2)
SERVICE_TENANT=$(keystone tenant-create --
name=$SERVICE_TENANT_NAME | grep "
id " | get_field 2)
# Users
ADMIN_USER=$(keystone user-create --name=admin --
pass="$ADMIN_PASSWORD" --
email=admin@domain.com | grep " id " | get_field 2)
DEMO_USER=$(keystone user-create --name=demo --
pass="$DEMO_PASSWORD" --
email=demo@domain.com --tenant-id=$DEMO_TENANT | grep " id "
| get_field 2)
NOVA_USER=$(keystone user-create --name=nova --
pass="$SERVICE_PASSWORD" --
tenant-id $SERVICE_TENANT --email=nova@domain.com | grep " id
" | get_field
```

```
2)
GLANCE_USER=$(keystone user-create --name=glance --
pass="$SERVICE_PASSWORD"
--tenant-id $SERVICE_TENANT --email=glance@domain.com | grep "
id " |
get_field 2)
QUANTUM_USER=$(keystone user-create --name=quantum --pass=
"$SERVICE_PASSWORD" --tenant-id $SERVICE_TENANT --
email=quantum@domain.com |
grep " id " | get_field 2)
CINDER_USER=$(keystone user-create --name=cinder --
pass="$SERVICE_PASSWORD"
--tenant-id $SERVICE_TENANT --email=cinder@domain.com | grep "
id " |
get_field 2)
# Roles
ADMIN_ROLE=$(keystone role-create --name=admin | grep " id " |
get_field 2)
MEMBER_ROLE=$(keystone role-create --name=Member | grep " id "
| get_field
2)
# Add Roles to Users in Tenants
keystone user-role-add --user-id $ADMIN_USER --role-id
$ADMIN_ROLE --tenantid
$ADMIN_TENANT
keystone user-role-add --tenant-id $SERVICE_TENANT --user-id
$NOVA_USER --
role-id $ADMIN_ROLE
keystone user-role-add --tenant-id $SERVICE_TENANT --user-id
$GLANCE_USER --
role-id $ADMIN_ROLE
```

keystone user-role-add --tenant-id $SERVICE_TENANT --user-id $QUANTUM_USER
--role-id $ADMIN_ROLE
keystone user-role-add --tenant-id $SERVICE_TENANT --user-id $CINDER_USER --
role-id $ADMIN_ROLE
keystone user-role-add --tenant-id $DEMO_TENANT --user-id $DEMO_USER --roleid
$MEMBER_ROLE
 # Create services
COMPUTE_SERVICE=$(keystone service-create --name nova --type compute --
description 'OpenStack Compute Service' | grep " id " | get_field 2)
VOLUME_SERVICE=$(keystone service-create --name cinder --type volume --
description 'OpenStack Volume Service' | grep " id " | get_field 2)
IMAGE_SERVICE=$(keystone service-create --name glance --type image --
description 'OpenStack Image Service' | grep " id " | get_field 2)
IDENTITY_SERVICE=$(keystone service-create --name keystone --type identity
--description 'OpenStack Identity' | grep " id " | get_field 2)
EC2_SERVICE=$(keystone service-create --name ec2 --type ec2 --description
'OpenStack EC2 service' | grep " id " | get_field 2)
NETWORK_SERVICE=$(keystone service-create --name quantum --type network --
description 'OpenStack Networking service' | grep " id " | get_field 2)
 # Create endpoints
keystone endpoint-create --region $KEYSTONE_REGION --service-id $COMPUTE_SERVICE --publicurl 'http://"$KEYSTONE_HOST":8774/v2/

$(tenant_id)s' --adminurl 'http://"'$KEYSTONE_HOST"':8774/v2/$(tenant_id)s' --internalurl 'http://"'$KEYSTONE_HOST"':8774/v2/$(tenant_id)s'

keystone endpoint-create --region $KEYSTONE_REGION --service-id $VOLUME_SERVICE --publicurl 'http://"'$KEYSTONE_HOST"':8776/v1/ $(tenant_id)s' --adminurl 'http://"'$KEYSTONE_HOST"':8776/v1/$(tenant_id)s' --internalurl 'http://"'$KEYSTONE_HOST"':8776/v1/$(tenant_id)s'

keystone endpoint-create --region $KEYSTONE_REGION --service-id $IMAGE_SERVICE --publicurl 'http://"'$KEYSTONE_HOST"':9292/v2' --adminurl 'http://"'$KEYSTONE_HOST"':9292/v2' --internalurl 'http:// "'$KEYSTONE_HOST"':9292/v2'

keystone endpoint-create --region $KEYSTONE_REGION --service-id $IDENTITY_SERVICE --publicurl 'http://"'$KEYSTONE_HOST"':5000/v2.0' --adminurl 'http://"'$KEYSTONE_HOST"':35357/v2.0' --internalurl 'http:// "'$KEYSTONE_HOST"':5000/v2.0'

keystone endpoint-create --region $KEYSTONE_REGION --service-id $EC2_SERVICE --publicurl 'http://"'$KEYSTONE_HOST"':8773/services/Cloud' --adminurl 'http://"'$KEYSTONE_HOST"':8773/services/Admin' --internalurl 'http:// "'$KEYSTONE_HOST"':8773/services/Cloud'

keystone endpoint-create --region $KEYSTONE_REGION --service-id $NETWORK_SERVICE --publicurl 'http://"'$KEYSTONE_HOST"':9696/' --adminurl 'http://"'$KEYSTONE_HOST"':9696/' --internalurl 'http:// "'$KEYSTONE_HOST"':9696/'

11. Install the Glance packages : *apt-get install -y glance glance-api glance-registry python-glanceclient glance-common*

12. Configure Glance by edit **/etc/glance/glance-api.conf** and **/etc/glance/glance-registry.conf**

[DEFAULT]
sql_connection = mysql://glance:password@localhost/glance
[keystone_authtoken]
admin_tenant_name = service
admin_user = glance
admin_password = password

13. Download test image : *wget http://download.cirros-cloud.net/0.3.1/cirros-0.3.1-x86_64-disk.img* and *glance image-create --is-public true --disk-format qcow2 --container-format bare --name "Cirros 0.3.1" < cirros-0.3.1-x86_64-disk.img*

14. Install the Nova packages : *apt-get install -y nova-api nova-cert nova-common nova-conductor \ nova-scheduler python-nova python-novaclient nova-consoleauth novnc \nova-novncproxy*

15. Configure Nova by edit **/etc/nova/api-paste.ini**

admin_tenant_name = service
admin_user = nova
admin_password = password

And **/etc/nova/nova.conf**

[DEFAULT]
sql_connection=mysql://nova:password@localhost/nova
rabbit_password=password

auth_strategy=keystone

 # Networking

network_api_class=nova.network.quantumv2.api.API

quantum_url=http://10.10.10.254:9696

quantum_auth_strategy=keystone

quantum_admin_tenant_name=service

quantum_admin_username=quantum

quantum_admin_password=password

quantum_admin_auth_url=http://10.10.10.254:35357/v2.0

libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver

linuxnet_interface_driver=nova.network.linux_net.LinuxOVSInterface Driver

 # Security Groups

firewall_driver=nova.virt.firewall.NoopFirewallDriver

security_group_api=quantum

 # Metadata

quantum_metadata_proxy_shared_secret=password

service_quantum_metadata_proxy=true

metadata_listen = 10.10.10.254

metadata_listen_port = 8775

 # Cinder

volume_api_class=nova.volume.cinder.API

 # Glance

glance_api_servers=10.10.10.254:9292

image_service=nova.image.glance.GlanceImageService

 # novnc

novnc_enable=true

novncproxy_port=6080

novncproxy_host=192.168.1.1

vncserver_listen=0.0.0.0

16. Install the Cinder packages : *apt-get install -y cinder-api cinder-scheduler cinder-volume iscsitarget \open-iscsi iscsitarget-dkms python-cinderclient linux-headers-`uname -r`*

17. Configure & start the iSCSI services :

sed -i 's/false/true/g' /etc/default/iscsitarget
service iscsitarget start
service open-iscsi start

18. Configure Cinder by **edit /etc/cinder/cinder.conf** :

[DEFAULT]
sql_connection = mysql://cinder:password@localhost/cinder
rabbit_password = password

And **/etc/cinder/api-paste.ini** :

admin_tenant_name = service
admin_user = cinder
admin_password = password

19. Install the Quantum Server : *apt-get install -y quantum-server*

20. Configure Quantum service by **edit /etc/quantum/quantum.conf** :

[DEFAULT]
verbose = True
rabbit_password = password
rabbit_host = 10.10.10.254

[keystone_authtoken]
auth_host = 10.10.10.254

admin_tenant_name = service
admin_user = quantum
admin_password = password

Next, edit Quantum OVS service **/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini** :

[DATABASE]
sql_connection = mysql://quantum:password@10.10.10.254/quantum

[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
local_ip = 10.10.10.254

[SECURITYGROUP]
firewall_driver = quantum.agent.linux.iptables_firewall.
OVSHybridIptablesFirewallDriver

And then, edit **/etc/quantum/dhcp_agent.ini** :

[DEFAULT]
enable_isolated_metadata = True
enable_metadata_network = True

For next, edit **/etc/quantum/metadata_agent.ini** again :

[DEFAULT]
auth_url = http://10.10.10.254:35357/v2.0
auth_region = RegionOne
admin_tenant_name = service

```
admin_user = quantum
admin_password = password
nova_metadata_ip = 10.10.10.254
metadata_proxy_shared_secret = password
```

Finally, create bash script for create an internal network as the "demo" project

```
#!/bin/bash

TENANT_NAME="demo"
TENANT_NETWORK_NAME="demo-net"
TENANT_SUBNET_NAME="${TENANT_NETWORK_NAME}-
subnet"
TENANT_ROUTER_NAME="demo-router"
FIXED_RANGE="10.10.10.0/24"
NETWORK_GATEWAY="10.10.10.254"
TENANT_ID=$(keystone tenant-list | grep " $TENANT_NAME " | awk
'{print $2}')
TENANT_NET_ID=$(quantum net-create --tenant_id $TENANT_ID
$TENANT_NETWORK_NAME --provider:network_type gre --
provider:segmentation_id 1 | grep " id " | awk '{print $4}')
TENANT_SUBNET_ID=$(quantum subnet-create --tenant_id
$TENANT_ID --ip_version
4 --name $TENANT_SUBNET_NAME $TENANT_NET_ID
$FIXED_RANGE --gateway
$NETWORK_GATEWAY --dns_nameservers list=true 8.8.8.8 | grep "
id " | awk
'{print $4}')
ROUTER_ID=$(quantum router-create --tenant_id $TENANT_ID
$TENANT_ROUTER_NAME
| grep " id " | awk '{print $4}')
quantum router-interface-add $ROUTER_ID $TENANT_SUBNET_ID
```

21. Enable the OVS plugin : *ln -s /etc/quantum/plugins/openvswitch/ovs_quantum _plugin.ini /etc/quantum/plugin.ini*

22. Install the Horizon package and its dependencies : *apt-get install -y openstack-dashboard memcached python-memcache*

Compute Node Setup

1. Install Ubuntu Cloud 12.04 on compute node.

2. Install OpenStack cloud service : *apt-get install ubuntu-cloud-keyring.*

3. Configure the network by edit **/etc/network/interfaces**

Primary-system

 # Internal Network
auto eth0
iface eth0 inet static
address 10.10.10.1
netmask 255.255.255.0
gateway 10.10.10.254

Backup-system

 # Internal Network
auto eth0
iface eth0 inet static
address 10.10.11.1
netmask 255.255.255.0
gateway 10.10.11.254

4. Edit **/etc/sysctl.conf** :

net.ipv4.conf.all.rp_filter = 0

net.ipv4.conf.default.rp_filter = 0

5. Install the Nova Compute package : *apt-get install nova-compute-kvm*

6. Configure Nova by edit **/etc/nova/api-paste.ini** :

[filter:authtoken]
auth_host = 10.10.10.254
admin_tenant_name = service
admin_user = nova
admin_password = password

And edit /etc/nova/nova.conf :

[DEFAULT]
 # General
verbose=True
rabbit_host=10.10.10.254
rabbit_password=password
auth_strategy=keystone
ec2_host=10.10.10.254
ec2_url=http://10.10.10.254:8773/services/Cloud
 # Networking
libvirt_use_virtio_for_bridges=True
network_api_class=nova.network.quantumv2.api.API
quantum_url=http://10.10.10.254:9696
quantum_auth_strategy=keystone
quantum_admin_tenant_name=service
quantum_admin_username=quantum
quantum_admin_password=password
quantum_admin_auth_url=http://10.10.10.254:35357/v2.0
 # Security Groups

firewall_driver=nova.virt.firewall.NoopFirewallDriver
security_group_api=quantum
 # Compute #
compute_driver=libvirt.LibvirtDriver
connection_type=libvirt
 # Cinder
volume_api_class=nova.volume.cinder.API
 # Glance
glance_api_servers=10.10.10.254:9292
image_service=nova.image.glance.GlanceImageService
 # novnc
vnc_enabled=true
vncserver_proxyclient_address=10.10.10.254
novncproxy_base_url=http://192.168.1.1:6080/vnc_auto.html
vncserver_listen=0.0.0.0

7. Install the Open vSwitch packages : *apt-get install -y openvswitch-switch*

8. Create an internal bridge : *ovs-vsctl add-br br-int*

9. Install the Quantum packages : *apt-get install -y quantum-plugin-openvswitch-agent*

10. Edit **/etc/quantum/quantum.conf**

[DEFAULT]
rabbit_host = 10.10.10.254
rabbit_password = password
verbose = True

And **/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini :**

[DATABASE]

sql_connection = mysql://quantum:password@10.10.10.254/quantum

[OVS]

tenant_network_type = gre

tunnel_id_ranges = 1:1000

local_ip = 10.10.10.254

enable_tunneling = True

[SECURITYGROUP]

firewall_driver = quantum.agent.linux.iptables_firewall.

OVSHybridIptablesFirewallDriver

SECURITY SETUP

User Permission Setup

Create new user to CLDR System with root permission.

1. Use the adduser command, replacing the “**newuser**” with username "**TRF**" : *sudo adduser newuser*

2. Grant a User Root Privileges with command : *sudo /usr/sbin/visudo*

3. Edit **TRF** user with the *sudo* privileges same permissions as root under the the user privilege specification

User privilege specification
root *ALL=(ALL:ALL) ALL*
TRF *ALL=(ALL:ALL) ALL*

Keychain Setup

Manage SSH keys with Keychain by

1. Install Keychain with sshpass to manage SSH and GPG keys in a convenient and secure manner on both system : *sudo apt-get -y install keychain* & *sudo apt-get install sshpass*

2. Generate a private/public pair of keys to allow a 'SSH' connection without asking for a password : *ssh-keygen -t rsa -b 2048*

3. Clone an id_rsa.pub both of Primary-system and Backup-system using ssh-copy-id

From Primary-system

ssh-copy-id -i ~/.ssh/id_rsa.pub Backup-system

From Backup-system

ssh-copy-id -i ~/.ssh/id_rsa.pub Primary-system

SHELL SCRIPT CODE

CLDR_CONNECT.sh

```
#!/bin/bash

#This is Schell Script for connect Primary-System on OpenStack
#Create by White Hat Key Maker

clear
echo -n "Enter your Primary-system IP : "
read Src
echo ""
echo -n "Enter Primary-system username : "
read userPri
echo ""
ssh $userPri@$Src
```

CLDR_CORE.sh

```
#!/bin/bash

#This is Schell Script for SNAP VM on openStack
#Create by White Hat Key Maker
#
#########################################################
########       Put This Shell Script on Backup-System  #########
#########################################################
#
clear
#
#         Identify User and Server ID.
#
echo -n "Enter Primary-system IP : "
read Src
#Src=192.168.1.199
#echo "$Src"
echo -n "Enter Primary-system username : "
read userPri
#userPri=root
#echo "$userPri"
echo ""
#
#         Show Nova list.
#
ssh $userPri@$Src source CLDR_NOVA_LIST.sh
#eval $(ssh $userPri@$Src source CLDR_NOVA_LIST.sh)
#source <(ssh $userPri@$Src source CLDR_NOVA_LIST.sh)
#echo "$Nova_List"
#ssh $userPri@$Src<<EOF
```

```
#source CLDR_NOVA_LIST.sh
#echo "SELECT display_name AS 'Instance_Name'\t, uuid AS 'Instance_ID'\t, vm_state AS 'Status' FROM nova.instances WHERE deleted = 'NULL';" > test.sql
#ssh $userPri@$Src 'mysql -h 127.0.0.1 -u root -ptrfadmin -e "SELECT display_name AS 'Instance_Name'\t, uuid AS 'Instance_ID'\t, vm_state AS 'Status' FROM nova.instances WHERE deleted = 'NULL';";'
#ssh $userPri@$Src 'mysql -uroot -pmygreatsecret < test.sql'
#EOF
echo ""
#
#       Nova Image Create with Transfer back into This System.
#
ssh -t $userPri@$Src source CLDR_VM_SNAP.sh
#Nova_List=$(ssh $userPri@$Src source CLDR_NOVA_LIST.sh)
#echo $Nova_List
#
#       Transfer image from Backup-System to Primary-System.
#
#source CLDR_VM_TRANSFER.sh
#Arc_Dir="/var/lib/glance/images/archive/"
#rsync -azvv --progress -e ssh $userBk@$Dest:$Arc_Dir $Arc_Dir
echo ""
ssh $userPri@$Src source CLDR_GLANCE_LIST.sh
echo ""
source CLDR_VM_REVIVE.sh
echo ""
```

CLDR_GLANCE_LIST.sh

```
#!/bin/bash

#This is Schell Script for LIST GLANCE Image on OpenStack
#Create by White Hat Key Maker
#
#########################################################
########       Put This Shell Script on BOTH-System  #########
#########################################################
#
#clear
echo ""
#
#         SQL command for execute GLANCE list
#
mysql -u root -ptrfadmin -e "SELECT name AS 'Image Name', id AS 'Image ID', status AS 'Status' FROM glance.images WHERE deleted = 'NULL';"
#GLANCE_List=$(mysql -u root -ptrfadmin -e "SELECT name AS 'Image Name', id AS 'Image ID', status AS 'Status' FROM glance.images WHERE deleted = 'NULL';")
echo ""
```

CLDR_NOVA_LIST.sh

```
#!/bin/bash

#This is Schell Script for LIST NOVA Instance on OpenStack
#Create by White Hat Key Maker
#
#########################################################
########       Put This Shell Script on BOTH-System  ##########
#########################################################
#
clear
echo ""
#
#         SQL command for execute Instance list
#
mysql -uroot -ptrfadmin -e "SELECT display_name AS 'Instance Name', uuid AS 'Instance ID', vm_state AS 'Status' FROM nova.instances WHERE deleted = 'NULL';"
#Nova_List=$(mysql -uroot -ptrfadmin -e "SELECT display_name AS 'Instance Name', uuid AS 'Instance ID', vm_state AS 'Status' FROM nova.instances WHERE deleted = 'NULL';")
echo ""
```

CLDR_VM_REVIVE.sh

```
#!/bin/bash

#This is Schell Script for Boot Image up on OpenStack
#Create by White Hat Key Maker
#
#########################################################
########       Put This Shell Script on BOTH-System  #########
#########################################################
#
#clear
echo ""
#
#         Select Image ID for boot.
#
echo "Enter an Image ID for REVIVE to Current-system : "
read ImgID
#InstID=16054f88-5639-4f28-aa7e-5b1b46ef455a
#InstID=2c4d16cd-001d-4f95-a804-b5f80aa6d7e4
echo ""
#
#         Upload Image to GLANCE.
#
Img_Dir="/var/lib/glance/images/"
#Img_Tmp=$Img_Dir$ImgID
#mv $Img_Tmp $Img_Tmp.img
glance image-download --file $Img_Dir$ImgID --name=$ImgID --is-public=true --
container-format=bare --disk-format=qcow2
#
#         Nova REVIVE Image.
#
```

#source ~/devstack/openrc

nova boot --image $ImgID --flavor m1.Test --key_name $ImgID

echo "Success REVIVE Image as file name : $ImgID"

echo ""

CLDR_VM_SNAP.sh

```
#!/bin/bash

#This is Schell Script for create and transfer NOVA Image from Primary-System on OpenStack
#Create by White Hat Key Maker
#clear
#########################################################
########       Put This Shell Script on Primary-System  ########
#########################################################
#
#         Select Instances ID for create Image.
#
echo "Enter an Instances ID for migrate to Disaster Recovery site : "
read InstID
#InstID=16054f88-5639-4f28-aa7e-5b1b46ef455a
#InstID=2c4d16cd-001d-4f95-a804-b5f80aa6d7e4
now=$(date +%Y%m%d-%H%M%S)
HOST=$(hostname)
SS_name=$HOST-$now-$InstID
echo ""
#
#         Nova Image Create.
#
#source ~/devstack/openrc
nova image-create $InstID $SS_name
echo "Success SNAP Image as file name : $SS_name"
echo ""
#
#         Archive Image file.
#
```

```
Img_Dir="/var/lib/glance/images/"
Arc_Dir="/var/lib/glance/images/archive/"
#SS_name=81b72437-2cde-4966-9496-b66913463c32
tar zcvf -P $Arc_Dir$SS_name.tar.gz $Img_Dir$SS_name
ls $Arc_Dir
echo ""
echo -n "Archive finished at "
date
echo ""
#
#       Identify Backup User and Server ID.
#
echo -n "Enter Backup-system IP : "
read Dest
#Dest=192.168.1.199
#echo "$Dest"
echo -n "Enter Backup-system username : "
read userBk
#userPri=root
#echo "$userBk"
echo ""
#
#       Transfer image from Primary-System to Backup-System.
#
rsync -azvv --progress $Arc_Dir -e ssh $userBk@$Dest:$Arc_Dir
rsync -azvv --progress $Img_Dir -e ssh $userBk@$Dest:$Img_Dir
```

CLDR_VM_TRANSFER.sh

```
#!/bin/bash

#This is Schell Script for Get Image from Primary-System to Backup-System on OpenStack
#Create by White Hat Key Maker
#clear
#########################################################
########       Put This Shell Script on Backup-System  ########
#########################################################
#
#         Identify User and Server ID.
#
#echo -n "Enter Primary-system IP : "
#read Src
#Src=192.168.1.199
#echo "$Src"
#echo -n "Enter Primary-system username : "
#read userPri
#userPri=root
#echo "$userPri"
echo ""
#
#         Transfer image from Backup-System to Primary-System.
#
Arc_Dir="/var/lib/glance/images/archive/"
rsync -azvv --progress -e ssh $userBk@$Dest:$Arc_Dir $Arc_Dir
echo ""
```

CLDR_CONFIG_CODE

Configure Management

Configure Header	#!/bin/bash
Configure Intro	#This is Schell Script for SNAP VM on openStack #Create by White Hat Key Maker
Command CLEAR	clear
Command ECHO	echo ""
Snapshots Name	CLDR_Test
Command NOVA Image Create	nova image-create $InstID $SS_name.img --poll
Echo Image Create Finish	echo "Success SNAP Image as file name : $SS_name.img"
Images Folder	Img_Dir="/var/lib/glance/images/"
Archive Folder	Arc_Dir="/var/lib/glance/images/archive/"
Command Archive Images	sudo tar zcvfP Arc_DirSS_name.tar.gz Img_DirSS_name.img
Show Archive Images	ls $Arc_Dir
Echo Archive Images Finish	echo -n "Archive finished at " date
Command Transfer	rsync -azvv --progress $Arc_Dir -e ssh $userBk@$Dest:$Arc_Dir rsync -azvv --progress $Img_Dir -e ssh $userBk@$Dest:$Img_Dir
Command GLANCE Image Download	glance image-create --name=$SS_name --disk-format=qcow2 --container-format=bare --is-public=true --file=Img_DirSS_name.img
Command NOVA Image Revive	nova boot --image $SS_name --flavor
Echo Image Boot Success	echo "Success REVIVE Image as file name : $SS_name"
Command OPENRC	source ~/devstack/openrc admin admin
Command SSH	ssh $userPri@$Src source SS_name.sh

Appendix Figure 1 CLDR Default Configure Code.

DATA DICTIONARY

DATABASE NAME : CLDR

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	config_number	int(11)			No	0	
2	header	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
3	intro_2	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
4	intro_3	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
5	intro	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
6	clear	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
7	echo	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
8	con_time	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
9	ss_name	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
10	nova_Img_cr	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
11	Img_success	varchar(80)	utf8_unicode_ci		Yes	*NULL*	
12	Img_Dir	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
13	Arc_Dir	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
14	Arc_Action	varchar(100)	utf8_unicode_ci		Yes	*NULL*	
15	Arc_List	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
16	Arc_success	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
17	con_date	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
18	Rsync_Arc	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
19	Rsync_Img	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
20	glance_Img_dn	varchar(150)	utf8_unicode_ci		Yes	*NULL*	
21	nova_Img_bt	varchar(100)	utf8_unicode_ci		Yes	*NULL*	
22	Boot_success	varchar(80)	utf8_unicode_ci		Yes	*NULL*	
23	openrc	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
24	ssh	varchar(60)	utf8_unicode_ci		Yes	*NULL*	

Appendix Figure 2 Config.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	con_id	int(11)			No	None	AUTO_INCREMENT
2	pri_con	text	utf8_unicode_ci		Yes	NULL	
3	bk_con	text	utf8_unicode_ci		Yes	NULL	

Appendix Figure 3 Config_file.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	flavors_number	int(11)			No	None	AUTO_INCREMENT
2	flavors_id	varchar(40)	utf8_unicode_ci		No	None	
3	id_server	int(11)			No	None	
4	name	varchar(20)	utf8_unicode_ci		Yes	NULL	
5	vcpu	int(11)			Yes	NULL	
6	ram	int(11)			Yes	NULL	
7	root_disk	int(11)			Yes	NULL	
8	ep_disk	int(11)			Yes	NULL	
9	swap_disk	int(11)			Yes	NULL	
10	public	varchar(20)	utf8_unicode_ci		Yes	NULL	

Appendix Figure 4 Flavors.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	images_number	int(11)			No	None	AUTO_INCREMENT
2	images_id	varchar(40)	utf8_unicode_ci		No	None	
3	id_server	int(11)			No	None	
4	name	varchar(40)	utf8_unicode_ci		Yes	NULL	
5	type	varchar(10)	utf8_unicode_ci		Yes	NULL	
6	status	varchar(20)	utf8_unicode_ci		Yes	NULL	
7	public	varchar(10)	utf8_unicode_ci		Yes	NULL	
8	protected	varchar(10)	utf8_unicode_ci		Yes	NULL	
9	image_format	varchar(20)	utf8_unicode_ci		Yes	NULL	

Appendix Figure 5 Images.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	instances_number	int(11)			No	*None*	AUTO_INCREMENT
2	instances_id	varchar(40)	utf8_unicode_ci		No	*None*	
3	id_server	int(11)			No	*None*	
4	server_host	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
5	name	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
6	image_name	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
7	ip_address	varchar(40)	utf8_unicode_ci		Yes	*NULL*	
8	flavors_name	varchar(40)	utf8_unicode_ci		Yes	*NULL*	
9	status	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
10	task	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
11	power_state	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
12	uptime	varchar(50)	utf8_unicode_ci		Yes	*NULL*	

Appendix Figure 6 Instances.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id_server	int(11)			No	*None*	AUTO_INCREMENT
2	ip_address	varchar(40)	utf8_unicode_ci		No	*None*	
3	name	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
4	os	varchar(20)	utf8_unicode_ci		Yes	*NULL*	
5	status	varchar(10)	utf8_unicode_ci		Yes	*NULL*	
6	location	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
7	detail	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
8	folder_path	varchar(40)	utf8_unicode_ci		Yes	*NULL*	
9	username	varchar(40)	utf8_unicode_ci		No	*None*	
10	password	varchar(40)	utf8_unicode_ci		No	*None*	

Appendix Figure 7 Server.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id_user**	int(11)			No	*None*	AUTO_INCREMENT
2	**firstname**	varchar(50)	utf8_unicode_ci		No	*None*	
3	**surname**	varchar(50)	utf8_unicode_ci		Yes	*NULL*	
4	**address**	varchar(255)	utf8_unicode_ci		Yes	*NULL*	
5	**username**	varchar(40)	utf8_unicode_ci		No	*None*	
6	**password**	varchar(40)	utf8_unicode_ci		No	*None*	
7	**permission**	varchar(50)	utf8_unicode_ci		No	*None*	

Appendix Figure 8 User.

CURRICULUM VITAE

NAME : Mr. Nutrapong Jariyathanoi

BIRTH DATE : March 26, 1984

BIRTH PLACE : Chiang Mai, Thailand

EDUCATION :

YEAR	**INSTITUTE**	**EGREE/DIPLOMA**
2006	Mae Fah Luang Univ.	B.Sc.(Computer Science)

POSITION/TITLE : System Engineer

WORK PLACE : The Thailand Research Fund

CERTIFICATION : MCPS : Microsoft Certified Professional
: MCSA : Microsoft Certified Solutions Associate Windows 8
: Cloud Security Alliance, Thailand Chapter 2014.
: International Cloud Workshop Foundation