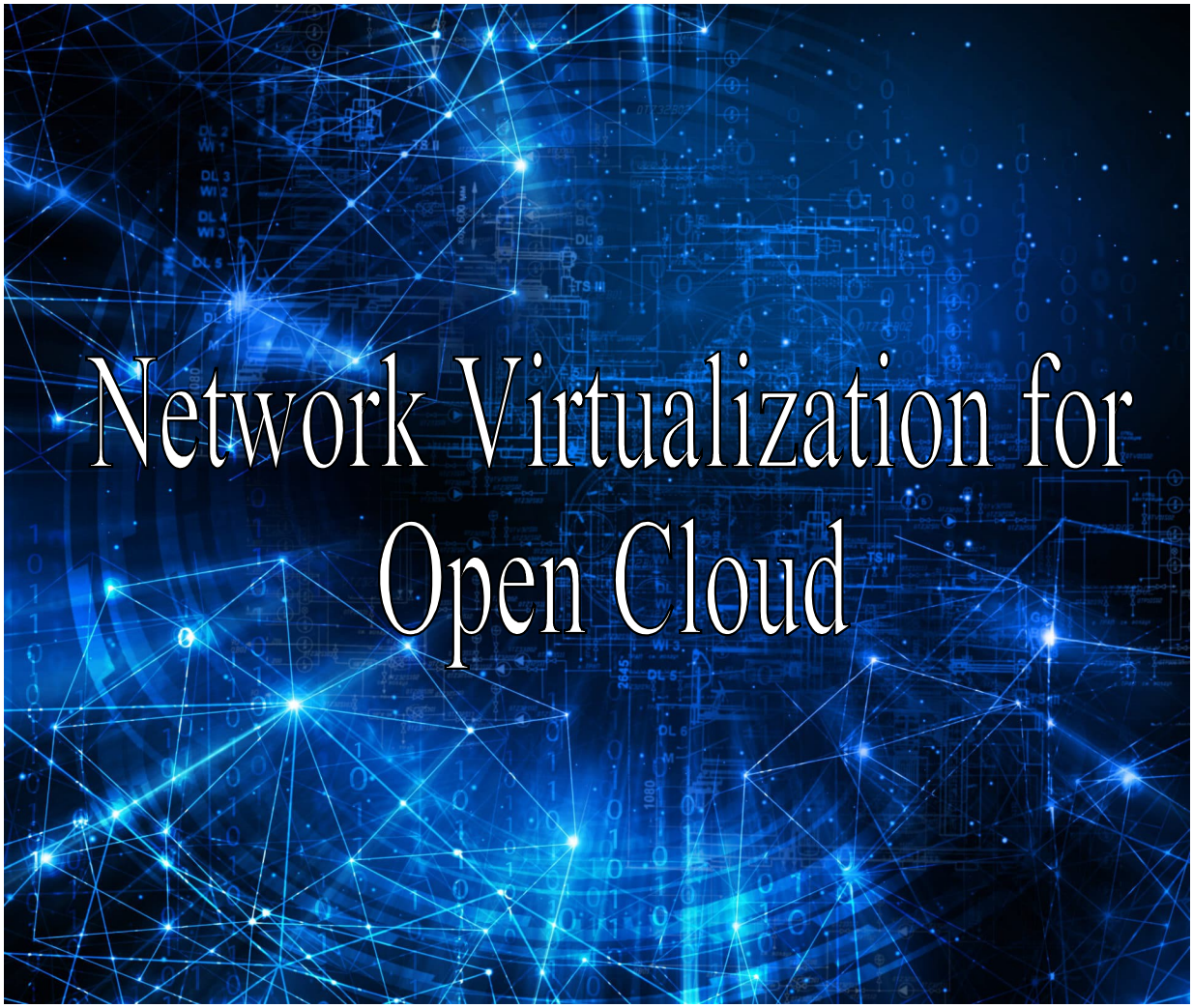
Πανεπιστήμιο Δυτικής Αττικής
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών



Network Virtualization for Open Cloud

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
της
ΣΟΥΒΑΛΙΩΤΗ ΜΑΡΙΑΣ

Επιβλέπων Καθηγητής: Ξυδάς Ιωάννης
Επίκουρος Καθηγητής ΠΑ.Δ.Α.

Αθήνα, Σεπτέμβριος, 2018

Περιεχόμενα

Κατάλογος εικόνων

Ευρετήριο πινάκων

Περίληψη

Στην παρούσα πτυχιακή θα ασχοληθούμε με το Network Virtualization, Open Cloud, software Data Center (sDC) με ανοιχτό και κλειστό λογισμικό, θα συγκρίνουμε τα λογισμικά και τον τρόπο διαμοιρασμού πόρων υλικού μεταξύ των εικονικών μηχανών του Data Center του Ε.Κ.Ε.Φ.Ε. Δημόκριτος. Θα δοκιμάσουμε σενάρια υλοποίησης της πλατφόρμας του open source λογισμικού oVirt (open Virtualization) πάνω σε τύπους και αριθμό hardware, αρχικά σε έναν server για επίδειξη του oVirt, έπειτα σε δύο servers με κεντρικό σύστημα storage και σε τρεις servers με κατανεμημένο σύστημα storage. Θα εξετάσουμε τις πρόσφατα αναπτυγμένες δυνατότητες για ιδεατά δίκτυα και διατάξεις (virtual networks, overlay networks), την κλωνοποίηση ιδεατών μηχανών, επίσης θα εξετάσουμε τους ρόλους διαφορετικών τύπων διαχειριστών (super admins, simple admins). Θα κάνουμε σύγκριση με την ανοικτή πλατφόρμας KVM - Virtual Manager που λειτουργεί με πραγματικό φορτίο δέκα ιδεατών μηχανών. Καθώς και με τις λειτουργίες της κλειστής πλατφόρμας VSphere με φορτίο δέκα ιδεατών μηχανών.

Σκοπός μας είναι να μάθουμε τους διαφορετικούς μηχανισμούς εικονικοποίησης υπολογιστικών αποθηκευτικών και δικτυακών πόρων, καθώς η κλίμακα εφαρμογής τους αυξάνεται. Επιθυμούμε να μπορέσουμε να οικειοποιηθούμε με την τεχνολογία της εικονικοποίησης και τους διάφορους τρόπους εφαρμογής της ώστε να γίνει δυνατή η εφαρμογή μίας τεχνικής πάνω στα ήδη υπάρχοντα συστήματα με απώτερο σκοπό την ενοποίησή τους σε συνεργασία με μία ανοιχτή τεχνολογία Cloud. Το ζητούμενο είναι να μπορεί να δημιουργηθεί ένα σταθερό και ευέλικτο Software Defined Data Center (SDDC) το οποίο θα παρέχει τις υπηρεσίες που χρειάζονται οι χρήστες του και θα μπορεί να είναι ασφαλές και πλεονάζον.

Abstract

In this dissertation, we deal with our Network Virtualization, the concept of Open Cloud, a case of software Data Center (sDC) based on open software as well as propriatory software. We compare the various ways of allocating resources to the virtual machines in NCSR Demokritos's Data Center. We will test the deployment of the open source platform oVirt (open Virtualization) on different types and number of hardware, initially by one Live CD demo PC, next on one server and finally on three servers with distributed file system storage. We examine the new features of oVirt like virtual networks, cloning of virtual machines and administrator roles (super admins, simple admins) for managing the platform. We study the extension of the present running platform consisting of KVM – Virtual Manager with oVirt hosting ten virtual machines and comparatively study the corresponding operation of the closed platform vSphere, both in our Data Center.

Our goal is to learn about the different types of Virtualization (compute, storage, network) and the increasing scale of their application, from one Data Center to the Open Cloud. We wish to familiriarize ourselves with the technology of Virtualization and the different ways of its application so a technique may be applied on the already existing systems, as the ultimate goal is to unite them under an Open Source Cloud technology. The desideratum being the creation of a stable and flexible Software Defined Data Center (SDDC) that will be able to provide the services the end users need as well as be secure and redundant.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω:

- Τον κ. Ιωάννη Ξυδά, επίκουρο καθηγητή του Πανεπιστημίου Δυτικής Αττικής, υπεύθυνο καθηγητή της εργασίας μου, για την βοήθεια και υποστήριξή του καθώς και για την καθοδήγησή του καθ' όλη τη διάρκεια της πτυχιακής εργασίας μου.
- Τον κ. Ιωάννη Κοροβέση, υπεύθυνο στο Κέντρο Αριάδνη του Ε.Κ.Ε.Φ.Ε. Δημόκριτος, που χάρη στην βοήθειά του, τις γνώσεις του και την παροχή των κατάλληλων μηχανημάτων αυτή η πτυχιακή εργασία κατέστη πραγματοποιήσιμη.
- Τον κ. Νίκο Μαρούγκα, για την βοήθεια και υποστήριξή του.
- Την οικογένειά μου, που είναι οι πιο σημαντικοί άνθρωποι στην ζωή μου.

ΚΕΦΑΛΑΙΟ 1 – Εισαγωγή

1.1. Περίληψη Κεφαλαίου

Σε αυτό το κεφάλαιο θα αναλύσουμε τις τεχνολογίες με τις οποίες θα ασχοληθούμε σε αυτό το project. Οι τεχνολογίες αυτές περιλαμβάνουν:

- την Εικονικοποίηση ή αλλιώς Virtualization και τους διάφορους τύπους στους οποίους απαντάται,
- το εργαλείο Ansible που έχει αναπτυχθεί με σκοπό την εισαγωγή αυτοματισμού στη διαχείριση διαδικτυακών συστημάτων “κρύβοντας” την χειροκίνητη πολυπλοκότητα σε κώδικα (ένας νέος κλάδος αναδεικνύεται και ονομάζεται DevOps, με αυτόν τον τρόπο μεγάλες διατάξεις με δυναμικά χαρακτηριστικά – Data Centers, Cloud platforms, εφαρμογές ελαστικού τύπου – έχουν αποτελεσματική διαχείριση),
- το εργαλείο Gdeploy που κάνει χρήση του Playbook του Ansible,
- το GlusterFS το οποίο αποτελεί μία μορφή χώρου Cloud και με το οποίο μπορούμε να δημιουργούμε πανομοιότυπους δίσκους σε διαφορετικούς κόμβους ώστε να προσφέρουμε σταθερή εμπειρία στους χρήστες μας,
- το Cloud και τους διάφορους τύπους στους οποίους απαντάται.

1.2. Virtualization

Στην επιστήμη της Πληροφορικής και του Προγραμματισμού, η Εικονικοποίηση (Virtualization) αναφέρεται στην πράξη της δημιουργίας και διάθεσης εικονικών μηχανών, είτε για λόγους έρευνας είτε για παροχή υπηρεσιών. Συγκεκριμένα, μέσω της Εικονικοποίησης, μπορούν να διατεθούν πλατφόρμες υλικού εικονικού υπολογιστή (virtual computer hardware platforms), συσκευές αποθήκευσης (storage devices) καθώς και πόροι δικτύου υπολογιστών (computer network resources).

Η Εικονικοποίηση ξεκίνησε την δεκαετία του 1960, ως μια μέθοδος λογικής διαίρεσης των υπολογιστικών πόρων των συστημάτων που παρέχονταν από τους κεντρικούς υπολογιστές (mainframe) μεταξύ διαφορετικών εφαρμογών. Από τότε η Εικονικοποίηση έχει εξελιχθεί και διευρυνθεί, εμπεριέχοντας πλέον πολλαπλούς τρόπους Εικονικοποίησης. Για παράδειγμα, μπορούμε πλέον να παρέχουμε στους πελάτες μας υπηρεσίες όπως παροχή σταθερών συστημάτων που δεν αποτυχαίνουν στη λειτουργία τους ή τη δυνατότητα να προσομοιώνεται ένα εντελώς διαφορετικό περιβάλλον σε κάποια ριζικά διαφορετική μηχανή (π.χ. emulators που μας επιτρέπουν να παίζουμε παιχνίδια PlayStation™ σε έναν υπολογιστή Microsoft Windows™). Επίσης, η διάθεση του συστήματος του ΕΟΠΠΥ στους συμβεβλημένους με αυτόν ιατρούς είναι βασισμένο πάνω σε Virtualized περιβάλλοντα και υπηρεσίες.

Οι κύριοι τύποι Virtualization είναι οι εξής:

1. Network Virtualization – Εικονικοποίηση Δικτύων
2. Hardware Virtualization – Εικονικοποίηση Υλικού
3. Desktop Virtualization
4. Nested Virtualization

1.2.1. Hypervisor – Virtual Machine Manager

Ένας Virtual Machine Manager επιτρέπει στους χρήστες:

- Να δημιουργούν και να επεξεργάζονται VMs (Εικονικές Μηχανές – Virtual Machines)
- Να εκκινούν και σταματούν τα VMs
- Να παρατηρούν και να ελέγχουν τα VMs
- Να παρακολουθούν την εκτέλεση και την χρήση των στατιστικών στοιχείων των VMs
- Να χρησιμοποιούν KVM, Xen ή QEMU εικονικές μηχανές που τρέχουν είτε τοπικά είτε απομακρυσμένα
- Να χρησιμοποιούν LXC containers

Το 1974, στο άρθρο τους «Τυπικές Απαιτήσεις για Εικονικοποιήσιμες Αρχιτεκτονικές Τρίτης Γενιάς» («Formal Requirements for Virtualizable Third Generation Architectures»), οι Gerald J. Popek και Robert P. Goldberg ταξινόμησαν τους Hypervisors σε δύο τύπους:

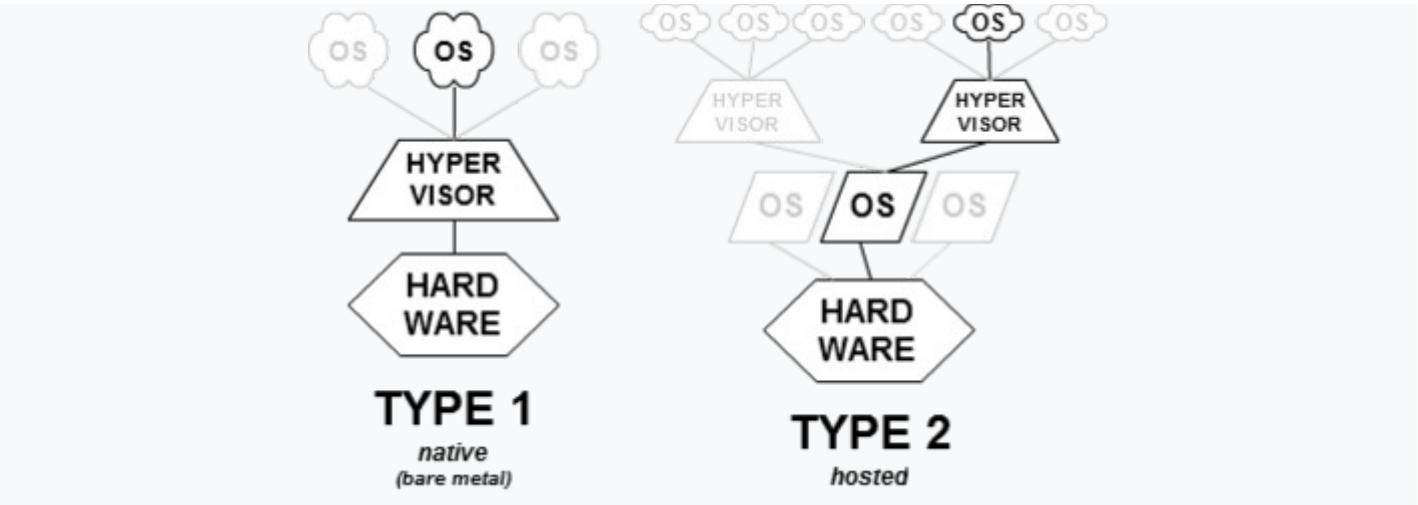
1. **Τύπος-1, φυσικός ή bare-metal Hypervisors**

   Αυτοί οι Hypervisors τρέχουν απευθείας πάνω στο υλικό του κεντρικού υπολογιστή και μπορούν να ελέγχουν το υλικό και να διαχειρίζονται το guest λειτουργικό σύστημα. Για αυτόν τον λόγο, αρκετές φορές, καλούνται bare-metal Hypervisors. Οι πρώτοι, τέτοιου τύπου, Hypervisors αναπτύχθηκαν από την IBM τη δεκαετία του 1960 και περιλάμβαναν το δοκιμαστικό λογισμικό SIMMON και το λειτουργικό σύστημα CP/CMS (που προηγήθηκε του z/VM της IBM). Σύγχρονα ισοδύναμα είναι τα AntsleOs, Xen, Oracle VM Server για SPARC, Oracle VM Server για x86, Microsoft Hyper-V, το λογισμικό συστήματος Xbox One και ο VMware ESX/ESXi Hypervisor.

2. **Τύπος-2, φιλοξενούμενοι (hosted) Hypervisors**

   Αυτοί οι Hypervisors τρέχουν σε ένα συμβατικό λειτουργικό σύστημα (OS), όπως κάνουν και τα υπόλοιπα προγράμματα ηλεκτρονικών υπολογιστών. Ένα guest λειτουργικό σύστημα λειτουργεί ως διαδικασία στον κεντρικό υπολογιστή. Οι Τύπου-2 Hypervisors διαχειρίζονται από το host λειτουργικό σύστημα του κεντρικού υπολογιστή τα guest λειτουργικά συστήματα. Το VMware Workstation, το VMware Player, το VirtualBox, το Parallels Desktop για Mac και το QEMU στα Linux είναι παραδείγματα τύπου 2 Hypervisors.

   Η διάκριση μεταξύ αυτών των δύο τύπων δεν είναι απαραιτήτως σαφής. Η εικονική μηχανή βασισμένη στον πυρήνα των Linux, KVM (Kernel-based Virtual Machine), και το bhyve του FreeBSD είναι μονάδες πυρήνα (kernel modules) που μετατρέπουν αποτελεσματικά το λειτουργικό σύστημα κεντρικού υπολογιστή σε έναν Τύπου-1 Hypervisor. Ταυτόχρονα, δεδομένου ότι οι διανομές Linux και το FreeBSD εξακολουθούν να είναι λειτουργικά συστήματα γενικής χρήσης, ενώ άλλες εφαρμογές ανταγωνίζονται για VM πόρους, τα KVM και bhyve μπορούν επίσης να κατηγοριοποιηθούν ως Τύπου-2 Hypervisors.



Εικόνα 1: Τύπος-1 και Τύπος-2 Hypervisors

1.2.1.1. Ενσωματωμένοι Hypervisors

Οι ενσωματωμένοι hypervisors, οι οποίοι στοχεύουν σε ενσωματωμένα συστήματα και σε συγκεκριμένα περιβάλλοντα λειτουργικού συστήματος σε πραγματικό χρόνο (Real Time Operating System – RTOS), σχεδιάζονται με διαφορετικές απαιτήσεις σε σύγκριση με τα desktop συστήματα και τα συστήματα production, συμπεριλαμβανομένων των δυνατοτήτων ανθεκτικότητας, ασφάλειας και πραγματικού χρόνου.

Η περιορισμένη από τους πόρους φύση πολλών ενσωματωμένων συστημάτων, ειδικά κινητών συστημάτων με μπαταρία, επιβάλλει μια επιπλέον απαίτηση για μικρό μέγεθος μνήμης και χαμηλό γενικό κόστος. Τέλος, σε αντίθεση με την πανταχού παρούσα αρχιτεκτονική x86 στον κόσμο των υπολογιστών, ο ενσωματωμένος κόσμος χρησιμοποιεί μια ευρύτερη ποικιλία αρχιτεκτονικών και λιγότερο τυποποιημένο περιβάλλον. Η υποστήριξη για εικονικοποίηση απαιτεί προστασία μνήμης (με τη μορφή μίας μονάδας διαχείρισης μνήμης ή τουλάχιστον μίας μονάδας προστασίας μνήμης) και μια διάκριση μεταξύ user mode και privileged mode, πράγμα που αποκλείει τους περισσότερους microconrtolers. Αυτό εξακολουθεί να αφήνει τις αρχιτεκτονικές x86, MIPS, ARM και PowerPC ως ευρέως αναπτυγμένες σε ενσωματωμένα συστήματα μεσαίου έως υψηλού επιπέδου.

Καθώς οι κατασκευαστές ενσωματωμένων συστημάτων έχουν συνήθως τον πηγαίο κώδικα στα λειτουργικά τους συστήματα, έχουν λιγότερη ανάγκη για πλήρες Virtualization σε αυτό το πλαίσιο. Αντ' αυτού, τα πλεονεκτήματα απόδοσης του paravirtualization κάνουν αυτόν τον τρόπο Virtualization συνήθως την τεχνολογία επιλογής. Παρόλα αυτά, οι ARM και MIPS πρόσθεσαν πρόσφατα τη δυνατότητα για πλήρη υποστήριξη Virtualization ως επιλογή IP και την συμπεριέλαβαν στους τελευταίους επεξεργαστές καθώς και στις εκδόσεις αρχιτεκτονικής τους, όπως είναι το ARM Cortex-A15 MPCore και το ARMv8 EL2.

Άλλες διαφορές μεταξύ του Virtualization σε servers, desktop υπολογιστές και ενσωματωμένα περιβάλλοντα συμπεριλαμβάνουν τις απαιτήσεις για την αποτελεσματική κατανομή των πόρων μεταξύ των εικονικών μηχανών (Virtual Machines - VMs), υψηλό εύρος ζώνης (bandwidth), χαμηλή καθυστέρηση στην εσωτερική επικοινωνία των εικονικών μηχανών, γενική προβολή του προγραμματισμού και της διαχείρισης ισχύος των συστημάτων και απόλυτο έλεγχο των ροών πληροφορίας.

1.2.2. Network Virtualization

Στον τομέα της πληροφορικής, όταν αναφερόμαστε στο Network Virtualization εννοούμε τη διαδικασία συνδυασμού υλικών και λογισμικών πόρων καθώς και δικτυακής λειτουργικότητας πάνω σε μία ενιαία οντότητα η οποία βασίζεται σε λογισμικό, ένα εικονικό δίκτυο.

Το Network Virtualization κατηγοριοποιείται είτε ως εξωτερικό Virtualization, το οποίο συνδυάζει πολλά δίκτυα ή μέρη πόρων δικτύων σε μία εικονική μονάδα ή σε εσωτερικό Virtuallization, το οποίο παρέχει λειτουργίες δικτύου σε δοχεία λογισμικού (software containers) και εικονικές μηχανές (virtual machines) σε έναν και μόνο server. Τα software containers είναι πιο "ελαφριές" κατασκευές, πιο minimal από τα minimal operating systems.

Χρήση του Network Virtualization έχουμε για παράδειγμα σε δοκιμές λογισμικού, όπου και οι προγραμματιστές χρησιμοποιούν το Network Virtualization για να ελέγξουν το λογισμικό του οποίου η παραγωγή είναι προς εξέλιξη σε προσομοιώσεις περιβαλλόντων δικτύου όπου τα λογισμικά αυτά προορίζονται να λειτουργούν. Ως συστατικό εφαρμοσμένων τεχνικών, το Network Virtualization επιτρέπει στους προγραμματιστές να μιμούνται συνδέσεις μεταξύ εφαρμογών, υπηρεσιών, εξαρτήσεων και τελικών χρηστών μέσα σε ένα δοκιμαστικό περιβάλλον χωρίς να χρειάζεται να εξετάσουν φυσικά όλους τους πιθανούς τρόπους συνδυασμού τύπων υλικού και λογισμικού. Η εγκυρότητα των δοκιμών αυτών εξαρτάται από την ακρίβεια με την οποία έχει υλοποιηθεί το εικονικό δίκτυο και εξομοιώνεται με το πραγματικό υλικό και τα λειτουργικά συστήματα.

1.2.2.1. Συνδυασμοί Network Virtualization

Διάφοροι προμηθευτές εξοπλισμού και λογισμικού προσφέρουν Network Virtualizatiton συνδυάζοντας οποιοδήποτε από τα παρακάτω:

- Υλικό δικτύου, όπως τα switches και οι προσαρμογείς δικτύου, επίσης γνωστά ως κάρτες διασύνδεσης δικτύου (Network Interface Controllers – NICs).
- Στοιχεία δικτύου, όπως είναι τα firewalls και οι balancers φορτίου.
- Δίκτυα, όπως τα εικονικά δίκτυα LAN (Virtual Local Area Network – VLAN) και containers, όπως είναι οι εικονικές μηχανές (VMs).
- Δικτυακές συσκευές αποθήκευσης.
- Στοιχεία δικτύου από μηχανή σε μηχανή, όπως οι συσκευές τηλεπικοινωνίας.
- Στοιχεία κινητής τηλεφωνίας δικτύου, όπως φορητοί υπολογιστές, tablet υπολογιστές και έξυπνα τηλέφωνα.
- Μέσα δικτύου, όπως τα Ethernet και Fiber Channel.

Εξωτερικό Virtualization: Το Virtualiization εξωτερικού δικτύου συνδυάζει ή υποδιαιρεί ένα ή περισσότερα τοπικά δίκτυα (Local Area Network – LAN) σε εικονικά δίκτυα για τη βελτίωση της αποτελεσματικότητας ενός μεγάλου δικτύου ή ενός κέντρου δεδομένων. Ένα εικονικό τοπικό δίκτυο (Virtual Local Area Network – VLAN) και το switch του δικτύου αποτελούν τα βασικά του στοιχεία. Μέσω αυτής της τεχνολογίας ένας διαχειριστής συστήματος μπορεί να διαμορφώσει συστήματα τα οποία είναι φυσικά συνδεδεμένα στο ίδιο τοπικό δίκτυο σε ξεχωριστά εικονικά δίκτυα. Επίσης, ένας διαχειριστής μπορεί να συνδυάσει συστήματα από ξεχωριστά τοπικά δίκτυα (LAN) σε ένα και μόνο VLAN που καλύπτει τμήματα ενός μεγάλου δικτύου.

Εσωτερικό Virtualization: Μια εσωτερικού δικτύου εικονικοποίηση διαμορφώνει είτε ένα ενιαίο σύστημα με δοχεία λογισμικού, όπως τα προγράμματα ελέγχου του Xen hypervisor, ή ψευδο-διασυνδέσεις, όπως είναι ένα VNIC (Virtual Network Interface Card), ώστε να μπορούν να μιμηθούν ένα φυσικό δίκτυο με λογισμικό. Αυτό μπορεί να βελτιώσει την αποτελεσματικότητα ενός ενιαίου συστήματος με την απομόνωση των εφαρμογών σε χωριστά δοχεία ή ψευδο-διασυνδέσεις.

1.2.2.2. Παραδείγματα

Τα Citrix και Vyatta έχουν δημιουργήσει μια εικονική στοίβα πρωτοκόλλων δικτύων συνδυάζοντας τη δρομολόγηση του Vyatta, firewall, και λειτουργείες VPN με το Netscaler load balancher της Citrix, τη βελτιστοποίηση των αναμεταδοτών (repeaters) δικτύου ευρείας περιοχής (Wide Area Network – WAN) και το Secure Sockets Layer VPN.

Το Network Virtualization του OpenSolaris παρέχει ένα λεγόμενο "δίκτυο σε πλαίσιο" ή αλλιώς “network in a box” (βλ. OpenSolaris Network Virtualization and Control Resource).

Ο Microsoft Virtual Server χρησιμοποιεί εικονικές μηχανές για να δημιουργήσει ένα “network in a box” για συστήματα x86. Αυτά τα container μπορούν να εκτελούν διαφορετικά λειτουργικά συστήματα, όπως τα Microsoft Windows ή το Linux, είτε είναι συνδεδεμένα ή ανεξάρτητα από έναν συγκεκριμένο ελεγκτή διεπαφής δικτύου (Network Interface Controller – NIC).

Η εικονικοποίηση δικτύων μπορεί να χρησιμοποιηθεί στην ανάπτυξη και δοκιμή εφαρμογών για να μιμηθεί λογισμικό υλικού και συστήματος πραγματικού κόσμου. Η μηχανοργάνωση επιδόσεων, η εικονικοποίηση δικτύων επιτρέπει την εξομοίωση των συνδέσεων μεταξύ εφαρμογών, υπηρεσιών, εξαρτήσεων και τελικών χρηστών για δοκιμές λογισμικού.

Για παράδειγμα, στην περίπτωση του OpenStack, το δίκτυο παρέχεται από την Neutron η οποία παρέχει πολλά χαρακτηριστικά και εργαλεία του πυρήνα των Linux που αφορούν τη δικτύωση (iptables, iproute2, L2 bridge, L3 routing, Open vSwitch – OVS). Σε μεγάλες εφαρμογές υπάρχει τεράστιος αριθμός εικονικών μηχανών και φυσικά πολλά διαφορετικά virtual LANs (VLANs) πάνω σε πολλούς Hypervisors. Οπότε χρειαζόταν ένας μηχανισμός διαχείρισης αυτών των δικτύων που “ζουν” στο virtual space. Πριν το OVS είχαμε μόνο το bridging το οποίο είναι μια απλή αρχιτεκτονική που σε μεγάλη κλίμακα αποτυγχάνει γιατί η κίνηση ανάμεσα στις εικονικές μηχανές πρέπει να “κατέβει” στο hardware δίκτυο και μετά να “ανέβει” πάλι επίπεδο, καθώς σε μεγάλα Data Center, Cloud ή μεγάλες εφαρμογές ένα μεγάλο μέρος της κίνησης είναι εσωτερικό.

1.2.3. Hardware Virtualization

Το Hardware Virtualization (Εικονικοποίηση Υλικού) ή αλλιώς Platform Virtualization (Εικονικοποίηση Πλατφόρμας) είναι η δημιουργία μίας εικονικής μηχανής η οποία λειτουργεί ως ένας πραγματικός υπολογιστής με λειτουργικό σύστημα. Το λογισμικό που εκτελείται σε αυτές τις εικονικές μηχανές είναι ξεχωριστό από τους υποκείμενους σε αυτό πόρους υλικού. Για παράδειγμα, ένας υπολογιστής που έχει εγκατεστημένο επάνω του λειτουργικό Windows, μπορεί να φιλοξενήσει μία εικονική μηχανή η οποία προσομοιάζει έναν υπολογιστή με λειτουργικό CentOS Linux, χωρίς τίποτα να εμποδίζει την κανονική εκτέλεση του CentOS. Τα Virtualization Platforms είναι πλατφόρμες οι οποίες μας προσφέρουν τη δυνατότητα δημιουργίας μίας εικονικής μηχανής, είτε όσον αφορά το υλικό της μηχανής είτε το λογισμικό της. Είναι ένας πολύ καλός τρόπος για να εξετάζουμε νέα λειτουργικά συστήματα, καθώς και τον τρόπο που λειτουργούν ανάλογα με διαφορετικό υλικό στο υπόβαθρο.

Τα Virtualization Platforms, και κυρίως οι emulators (προσομοιωτές) και οι hypervisors (επόπτες), είναι πακέτα λογισμικού τα οποία εξομοιώνουν ολόκληρη μία φυσική μηχανή και συνήθως μπορούν και μας παρέχουν πολλαπλές εικονικές μηχανές (Virtual Machines).

Στο Hardware Virtualization, το μηχάνημα του κεντρικού υπολογιστή είναι το πραγματικό μηχάνημα στο οποίο πραγματοποιείται το virtualization και αποκαλείται host machine ενώ το μηχάνημα που φιλοξενεί την εικονική μηχανή είναι το εικονικό μηχάνημα και αποκαλείται guest machine. Οι λέξεις host και guest χρησιμοποιούνται για να μπορεί να γίνει η διάκριση μεταξύ του λογισμικού που τρέχει στο φυσικό μηχάνημα από αυτό που τρέχει στην εικονική μηχανή. Το software ή το firmware που δημιουργεί μία

εικονική μηχανή στο υλικό του host αποκαλείται submenu (υπομενού) ή Virtual Machine Manager (Διαχειριστής Εικονικής Μηχανής). Το virt-manager, για παράδειγμα, είναι ο Virtual Machine Manager που έχει αναπτυχθεί από την Red Hat και είναι διαθέσιμο για όλες τις εκδόσεις των Linux, στα οποία μπορεί να υποστηριχθεί το Virtualization.

Διαφορετικοί τύποι Hardware Virtualization περιλαμβάνουν:

Full Virtualization – σχεδόν πλήρης προσομοίωση του πραγματικού υλικού, ώστε το λογισμικό, το οποίο συνήθως αποτελείται από ένα guest λειτουργικό σύστημα, να τρέχει χωρίς τροποποιήσεις.

Paravirtualization – ένα περιβάλλον υλικού δεν μπορεί να προσομοιώνεται. Ωστόσο, τα προγράμματα φιλοξενούμενων εκτελούνται σε δικούς τους απομονωμένους τομείς, σαν να τρέχουν σε ξεχωριστό σύστημα. Τα φιλοξενούμενα προγράμματα πρέπει να τροποποιηθούν ειδικά για να εκτελούνται σε αυτό το περιβάλλον.

Το Hardware Virtualization είναι ένας τρόπος βελτίωσης της συνολικής αποδοτικότητας του virtualization. Περιλαμβάνει κεντρικές μονάδες επεξεργασίας (CPUs) που παρέχουν υποστήριξη για virtualization στο υλικό και άλλα στοιχεία υλικού που συμβάλλουν στη βελτίωση της απόδοσης ενός περιβάλλοντος φιλοξενουμένων.

Το Hardware Virtualization μπορεί να θεωρηθεί ως μέρος μιας γενικής τάσης στον τομέα των επιχειρήσεων IT που περιλαμβάνει αυτόνομες ηλεκτρονικές υποδομές επεξεργασίας πληροφορίας, ένα σενάριο στο οποίο το περιβάλλον πληροφορικής θα είναι σε θέση να διαχειριστεί τον εαυτό του με βάση την αντιλαμβανόμενη δραστηριότητα και την υπολογιστική χρησιμότητα, χρησιμότητα που οι πελάτες μπορούν να πληρώσουν μόνο για όσο χρειάζεται.

Ο συνηθισμένος στόχος του Hardware Virtualization είναι να συγκεντρώσει τα διοικητικά καθήκοντα ενώ παράλληλα να βελτιώσει την δυνατότητα κλιμάκωσης και τη συνολική χρήση των πόρων του υλικού. Με το Virtualization, διάφορα λειτουργικά συστήματα μπορούν να λειτουργούν παράλληλα σε μία κεντρική μονάδα επεξεργασίας (CPU). Αυτός ο παραλληλισμός τείνει να μειώσει τα γενικά έξοδα και διαφέρει από το multitasking, το οποίο περιλαμβάνει την εκτέλεση πολλών προγραμμάτων στο ίδιο λειτουργικό σύστημα κατά την ίδια χρονική περίοδο. Με τη χρήση του virtualization, μια επιχείρηση μπορεί να διαχειρίζεται καλύτερα τις ενημερώσεις και γρήγορες αλλαγές στο λειτουργικό σύστημα και τις εφαρμογές χωρίς να διαταράσσει την ομαλή λειτουργία του συστήματος για τον χρήστη. «Εν τέλει, το virtualization βελτιώνει δραματικά την αποδοτικότητα και τη διαθεσιμότητα πόρων και εφαρμογών σε έναν οργανισμό ή μια επιχείρηση. Αντί να βασιζόμαστε στο παλιό μοντέλο "ένας server, μία εφαρμογή" που οδηγεί σε ανεπαρκείς πόρους, δουλεύουμε με εικονικούς πόρους οι οποίοι εφαρμόζονται δυναμικά για να μπορούν να καλύψουν τις ανάγκες των επιχειρήσεων χωρίς πλεονάζον λίπος» (ConsonusTech).

Η Εικονικοποίηση υλικού δεν είναι η ίδια με την εξομοίωση υλικού. Στην εξομοίωση υλικού, ένα κομμάτι υλικού μιμείται ένα άλλο, ενώ στον τομέα της Εικονικοποίησης υλικού, ένας hypervisor (που είναι ένα κομμάτι του λογισμικού) μιμείται ένα συγκεκριμένο κομμάτι υλικού υπολογιστή ή ολόκληρο τον υπολογιστή. Επιπλέον, ένας hypervisor δεν είναι ο ίδιος με έναν εξομοιωτή. Και τα δύο είναι προγράμματα υπολογιστών που μιμούνται υλικό, αλλά ο τομέας χρήσης τους στη γλώσσα διαφέρει.

1.2.3.1. Υποστήριξη Λειτουργικού Συστήματος

Οι βασικοί λόγοι οι οποίοι οδήγησαν στην ανάγκη για χρήση του Virtualization στα λειτουργικά συστήματα UNIX και Linux ήταν:

- Επέκταση των ικανοτήτων του υλικού, επιτρέποντας σε κάθε μία μηχανή να μπορεί να εκτελεί ακόμα περισσότερες εργασίες ταυτόχρονα.
- Προσπάθεια ελέγχου κόστους και απλοποίησης της διαχείρισης μέσω της ενοποίησης των servers.
- Ανάγκη για τον έλεγχο των μεγάλων εγκαταστάσεων πολυεπεξεργαστών και συμπλεγμάτων (Clusters).
- Η παροχή αυξημένης και βελτιωμένης ασφάλειας, αξιοπιστίας και ανεξαρτησίας της συσκευής, οι οποίς είναι δυνατές χάρη στις αρχιτεκτονικές των Hypervisors.
- Δυνατότητα εκτέλεσης σύνθετων εφαρμογών εξαρτώμενων από συγκεκριμένα λειτουργικά συστήματα σε διαφορετικά περιβάλλοντα υλικού ή ακόμα και διαφορετικά λειτουργικά συστήματα.

Μεγάλοι προμηθευτές UNIX συστημάτων, συμπεριλαμβανομένων των Sun Microsystems, HP, IBM και SGI, διέθεταν στην αγορά εικονικοποιημένο υλικό αρκετά πριν το 2000. Συνήθως, αυτό αφορούσε μεγάλα και ακριβά συστήματα, αξίας αρκετών εκατομμυρίων δολαρίων, αν και υπήρχε διαθέσιμο Virtualization σε χαμηλό ή και μεσαίο κόστος, όπως ήταν τα συστήματα της IBM με τους pSeries server, της Sun/Oracle με τους T-Series CoolThreads servers και της HP με τις μηχανές Superdomeseries.

Παρά το γεγονός ότι η Solaris ήταν πάντα το μόνο guest domain λειτουργικό σύστημα που υποστηρίχθηκε επίσημα από την Sun/Oracle με τον Logical Domains Hypervisor τους, στα τέλη του 2006, τα Linux (Ubuntu και Gentoo) καθώς και το FreeBSD μεταφέρθηκανεκτελούνται πάνω από τον Hypervisor και μπορούν όλα να τρέχουν ταυτόχρονα στον ίδιο επεξεργαστή, ως πλήρως εικονικοποιημένα ανεξάρτητα guest λειτουργικά συστήματα. Η πλήρης εικονικοποίηση στους επεξεργαστές SPARC αποδείχθηκε απλή: από την δημιουργία της στα μέσα της δεκαετίας του 1980, η Sun διατήρησε σκόπιμα την αρχιτεκτονική SPARC καθαρή από στοιχεία τα οποίαθα εμπόδιζαν την εικονικοποίηση.

Η HP με τα συστήματα Itanium "Integrity Virtual Machines" (Integrity VM) μπορεί να φιλοξενήσει τεχνολογία πολλαπλών λειτουργικών συστημάτων. Το Itanium μπορεί να εκτελέσει HP-UX, Linux, Windows και OpenVMS συστήματα. Εκτός από το OpenVMS, το οποίο υποστηρίζεται σε μεταγενέστερη έκδοση, αυτά τα περιβάλλοντα υποστηρίζονται επίσης και ως εικονικοί servers στην πλατφόρμα Integrity VM της HP. Το λειτουργικό σύστημα HP-UX φιλοξενεί το Integrity VM Hypervisor layer το οποίο επιτρέπει την αξιοποίηση πολλών σημαντικών χαρακτηριστικών του HP-UX και παρέχει σημαντική διαφοροποίηση μεταξύ αυτής της πλατφόρμας και άλλων πλατφορμών, όπως η λειτουργία hotswap της μνήμης, η λειτουργία hotswap του επεξεργαστή και ο δυναμικές ενημερώσεις του πυρήνα χωρίς επανεκκίνηση του συστήματος. Παρότι αξιοποιεί πολύ το HP-UX λειτουργικό σύστημα, ο Hypervisor Integrity VM είναι στην πραγματικότητα ένα υβρίδιο που τρέχει σε γυμνό μέταλλο ενώ οι επισκέπτες εκτελούν τις εργασίες τους. Η εκτέλεση των κανονικών εφαρμογών HP-UX σε έναν κεντρικό υπολογιστή Integrity VM αποθαρρύνεται μερικές φορές, επειδή ο Integrity VM εφαρμόζει τις δικές του ρυθμίσεις διαχείρισης μνήμης, προγραμματισμού και Εισόδου/Εξόδου (Input/OutputI/O) που συντονίζονται για εικονικές μηχανές και δεν είναι τόσο αποτελεσματικές για κανονικές εφαρμογές. Η HP παρέχει επίσης πιο άκαμπτο διαχωρισμό των συστημάτων Integrity και HP9000 μέσω τεχνολογιών VPAR και nPar, ενώ η πρώτη προσφέρει κοινόχρηστο διαμοιρασμό πόρων και η τελευταία προσφέρει πλήρες I/O και επεξεργασία απομόνωσης. Η ευελιξία του περιβάλλοντος εικονικού server (Virtual Server Environment – VSE) έδωσε τη δυνατότητα χρήσης του πιο συχνά σε νεότερες υλοποιήσεις.

Η IBM παρέχει τεχνολογία virtualization των partitions, γνωστή και ως λογική διαμέριση (Logical PARtitioning – LPAR) στα συστήματα System/390, zSeries, pSeries και iSeries. Για τα Power Systems της IBM, ο POWER Hypervisor (PHYP) είναι ένας εγγενής (bare-metal) Hypervisor στο υλικολογισμικό και παρέχει απομόνωση μεταξύ των LPAR. Η χωρητικότητα του επεξεργαστή παρέχεται στους LPAR είτε με ειδικό τρόπο είτε με βάση τα δικαιώματα, όπου η αχρησιμοποίητη χωρητικότητα συλλέγεται και μπορεί να ανακατανεμηθεί σε φορτία εργασίας. Οι ομάδες LPAR μπορούν να έχουν τη χωρητικότητα του επεξεργαστή

τους σαν να βρίσκονταν σε μια "ομάδα" – η IBM αναφέρεται σε αυτήν την δυνατότητα ως Multiple Shared – Processor Pools (MSPPs) και την υλοποιεί σε servers με τον επεξεργαστή POWER6. Οι κατανομές δυνατοτήτων LPAR και MSPP μπορούν να αλλάξουν δυναμικά. Η μνήμη κατανέμεται σε κάθε LPAR (στην εκκίνηση του LPAR ή δυναμικά) και ελέγχεται από τον POWER Hypervisor.

Για διευθυνσιοδότηση σε πραγματικό χρόνο λειτουργιών από τα λειτουργικά συστήματα (AIX, Linux, IBM i), οι επεξεργαστές POWER (POWER4 και εξής) έχουν σχεδιασμένες δυνατότητες εικονικοποίησης όπου η αντιστοίχιση των διευθύνσεων του υλικού αντιστοιχούνται με τις διευθύνσεις του λειτουργικού συστήματος για να φθάσει στη φυσική διεύθυνση μνήμης. Οι προσαρμογείς Εισόδου/Εξόδου (I/O) μπορούν να ανήκουν αποκλειστικά σε ένα LPAR ή να μοιράζονται το LPAR μέσω ενός partition μιας συσκευής γνωστού και ως Virtual I/O Server (VIOS). Το Power Hypervisor παρέχει υψηλά επίπεδα αξιοπιστίας, διαθεσιμότητας και συντήρησης (Reliability, Availability, Serviceability – RAS), διευκολύνοντας την άμεση προσθήκη ή και αντικατάσταση πολλών εξαρτημάτων (αναλόγως με το μοντέλο: επεξεργαστές, μνήμη, προσαρμογείς εισόδου/εξόδου, ανεμιστήρες, μονάδες ισχύος, δίσκοι, ελεγκτές συστημάτων κλπ.).

Παρόμοιες τάσεις συνέβησαν και με τις πλατφόρμες servers x86 / x86-64, όπου έργα ανοιχτού κώδικα όπως το Xen οδήγησαν σε προσπάθειες για Virtualization. Αυτές περιλαμβάνουν τους hypervisors που βασίζονται σε πυρήνες Linux και Solaris καθώς και σε προσαρμοσμένους πυρήνες.

1.2.3.2. x86 Συστήματα

Από το 2005, οι προμηθευτές CPU έχουν προσθέσει Hardware Virtualization Assistance στα προϊόντα τους, όπως για παράδειγμα, το Intel VT-x (με κωδικό όνομα Vanderpool) και το AMD-V (με την κωδική ονομασία Pacifica).
Μια εναλλακτική προσέγγιση απαιτεί την τροποποίηση του guest λειτουργικού συστήματος για την πραγματοποίηση κλήσεων συστήματος προς τον Hypervisor, αντί για την εκτέλεση εντολών I/O του μηχανήματος που εξομοιώνει ο Hypervisor. Αυτό ονομάζεται paravirtualization στο Xen, "hypercall" στο Parallels Workstation, και κώδικας "DIAGNOSE" στο VM της IBM. Όλα είναι στην πραγματικότητα το ίδιο πράγμα, μια κλήση συστήματος στον παρακάτω Hypervisor. Μερικοί μικρο-πυρήνες όπως είναι ο Mach και το L4 είναι αρκετά ευέλικτοι, έτσι ώστε να είναι δυνατό το "paravirtualization" των guest λειτουργικών συστημάτων.

1.2.4. Nested Virtualization

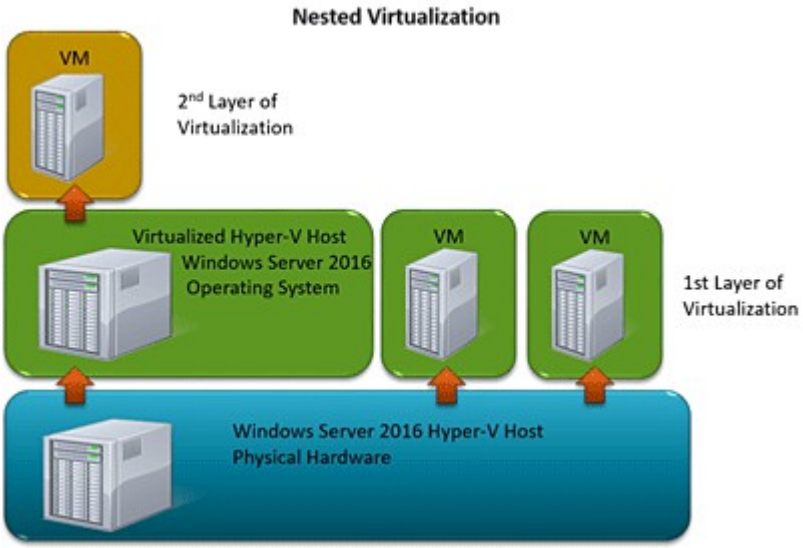
Το Nested Virtualization (Ενσωματωμένη Εικονικοποίηση), είναι η διαδικασία κατά την οποία αναπτύσσεται, δημιουργείται και εκτελείται μια εικονική μηχανή μέσα σε μία άλλη εικονική μηχανή. Αφορά στην εκτέλεση ενός ή και περισσοτέρων Hypervisors μέσα σε έναν άλλον Hypervisor, την εικονικοποίηση μέσα σε ένα ήδη εικονικοποιημένο περιβάλλον. Η φύση μιας ενσωματωμένης εικονικής μηχανής δεν χρειάζεται να είναι ομογενής της εικονικής μηχανής η οποία φιλοξενεί την πρώτη. Για παράδειγμα, μια εφαρμογή εικονικοποίησης μπορεί να αναπτυχθεί μέσα σε μία άλλη εικονική μηχαή η οποία έχει δημιουργηθεί με χρήση Hardware Virtualization.

Ο Hypervisor ο οποίος φιλοξενεί την κεντρική εικονική μηχανή, αναφέρεται ως Hypervisor επιπέδου 0 ή L0 (Layer 0) και ο Hypervisor ο οποίος φιλοξενείται στον L0 αναφέρεται ως Hypervisor επιπέδου 1 ή L1(Layer 1).

Το Nested Virtualization δεν υποστηρίζεται από όλους τους Hypervisors. Μερικοί από αυτούς που υποστηρίζουν το Nested Virtualization είναι το KVM των Linux, το VMware ESXi της VMware και το Hyper-V της Microsoft για τον Windows Server 2016.

Το Nested Virtualization χρησιμοποιείται για λόγους ανάπτυξης, δοκιμών και εκπαίδευσης στο Virtualization και η χρήση του αρχίζει να γίνεται απαραίτητη καθώς γνωστά λειτουργικά συστήματα αναπτύσσουν αυτή τη λειτουργικότητα και σε ένα εικονικοποιημένο περιβάλλον μπορεί να χρησιμοποιηθεί μόνο εάν ο L0 Hypervisor υποστηρίζει τη λειτουργία του Nested Virtualization. Επίσης, η μετακίνση ενός ήδη υπάρχοντος εικονικού περιβάλλοντος μέσα σε ένα cloud, ακολουθώντας το Infrastructrure as a Service (IaaS) σαν οδηγό, γίνεται ευκολότερη όταν η πλατφόρμα προορισμού υποστηρίζει το Nested Virtualization.

Η υλοποίηση του Nested Virtualization πάνω σε μία συγκεκριμένη αρχιτεκτονική εξαρτάται από τις βοηθητικές δυνατότητες που προσφέρει το υλικό. Εάν μία αρχιτεκτονική δεν υποστηρίζει την απαίτηση για Nested Virtualization, τότε υπάρχουν διάφορες τεχνικές, σε επίπεδο λογισμικού πλέον, οι οποίες χρησιμοποιούνται για να κατασταθεί ικανή η υποστήριξή του σε αυτές. Με το πέρας του χρόνου, όλο και περισσότερες αρχιτεκτονικές αποκτούν την κατάλληλη υποστήριξη από το υλικό καθώς το Nested Virtualization είναι μία από τις επιταχυνόμενα εξελισσόμενες τεχνολογίες.



Εικόνα 2: Nested Vritualization με 3 επίπεδα στον Hyper-V της Microsoft

1.2.5. Άλλοι τύποι Virtualization

- **Λογισμικό**
  1. **Εικονικοποίηση σε επίπεδο λειτουργικού συστήματος**: φιλοξενία πολλαπλών εικονικοποιημένων περιβαλλόντων σε μια μοναδική παρουσία OS
  2. **Εικονικοποίηση εφαρμογών και εικονικοποίηση χώρου εργασίας**: απομόνωση μεμονωμένων εφαρμογών από το υποκείμενο λειτουργικό σύστημα και άλλες εφαρμογές. που συνδέονται στενά με την έννοια των φορητών εφαρμογών
  3. **Εικονικοποίηση υπηρεσιών**: εξομοίωση της συμπεριφοράς συγκεκριμένων στοιχείων σε ετερογενείς εφαρμογές που βασίζονται σε στοιχεία, όπως API-driven εφαρμογές, εφαρμογές που βασίζονται σε cloud και αρχιτεκτονικές προσανατολισμένες στις υπηρεσίες

- **Μνήμη**
  1. **Εικονικοποίηση μνήμης**: συγκέντρωση πόρων μνήμης τυχαίας προσπέλασης (RAM) από δικτυωμένα συστήματα σε μια ενιαία μνήμη μνήμης

2. **Εικονική μνήμη**: δίνοντας σε μια εφαρμογή την εντύπωση ότι έχει συνεχή μνήμη εργασίας, απομονώνοντάς την από την υλοποίηση της φυσικής μνήμης

- **Αποθήκευση**
  1. **Εικονικοποίηση αποθήκευσης**: η διαδικασία της πλήρους αφαίρεσης της λογικής αποθήκευσης από την φυσική αποθήκευση
  2. **Κατανεμημένο σύστημα αρχείων**: οποιοδήποτε σύστημα αρχείων που επιτρέπει την πρόσβαση σε αρχεία από πολλούς κεντρικούς υπολογιστές που μοιράζονται μέσω δικτύου υπολογιστών
  3. **Εικονικό σύστημα αρχείων**: ένα στρώμα αφαίρεσης πάνω από ένα πιο συγκεκριμένο σύστημα αρχείων, επιτρέποντας στις εφαρμογές των πελατών να έχουν πρόσβαση σε διαφορετικούς τύπους συστημάτων αρχείων σκυροδέματος με ομοιόμορφο τρόπο
  4. **Υποδοχέας αποθήκευσης**: το λογισμικό που διαχειρίζεται την εικονικοποίηση αποθήκευσης και συνδυάζει τους φυσικούς πόρους αποθήκευσης σε μία ή περισσότερες εύκαμπτες ομάδες λογικής αποθήκευσης.
  5. **Εικονικός δίσκος**: ένα πρόγραμμα υπολογιστή που εξομοιώνει μια μονάδα δίσκου, όπως μια μονάδα σκληρού δίσκου ή μια μονάδα οπτικού δίσκου (δείτε τη σύγκριση λογισμικού εικόνας δίσκου)

- **Δεδομένα**
  1. **Εικονικοποίηση δεδομένων**: η παρουσίαση των δεδομένων ως αφηρημένη στρώση, ανεξάρτητα από τα υποκείμενα συστήματα βάσεων δεδομένων, τις δομές και την αποθήκευση
  2. **Βελτιστοποίηση βάσης δεδομένων**: η αποσύνδεση του στρώματος της βάσης δεδομένων, το οποίο βρίσκεται μεταξύ των επιπέδων αποθήκευσης και εφαρμογών μέσα στη στοίβα εφαρμογών πάνω από όλα

- **Δίκτυο**
  1. **Εικονικοποίηση δικτύων**: Δημιουργία ενός εικονικού δικτύου που καλύπτει το χώρο εντός ή μεταξύ των υποδικτύων δικτύου
  2. **Εικονικό ιδιωτικό δίκτυο (VPN):** πρωτόκολλο δικτύου που αντικαθιστά το πραγματικό καλώδιο ή άλλα φυσικά μέσα σε ένα δίκτυο με αφηρημένο επίπεδο, επιτρέποντας τη δημιουργία ενός δικτύου μέσω του Διαδικτύου

1.3. Ansible

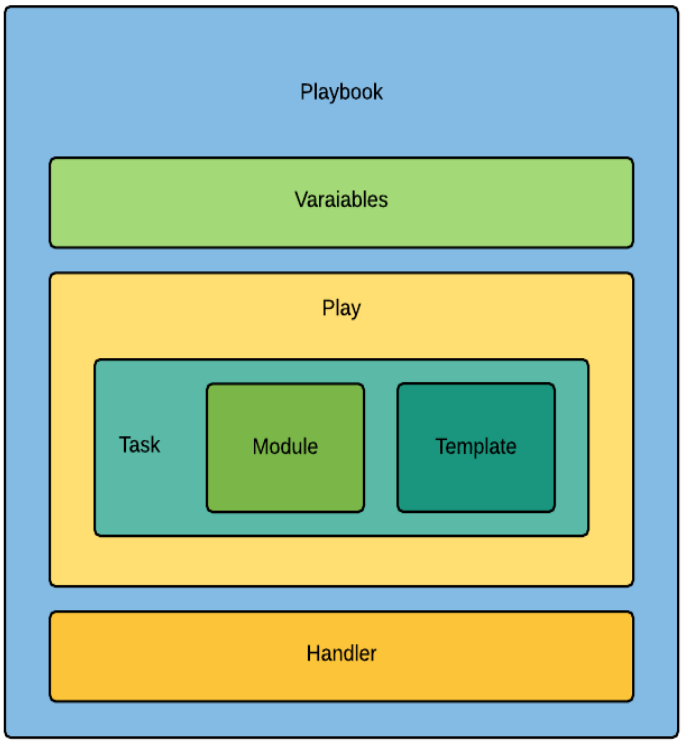
Το Ansible είναι ένα ανοιχτού τύπου λογισμικό το οποίο παρέχει τη δυνατότητα εφαρμογής της εγκατάστασης, διαχείρισης ή την εφαρμογή αλλαγών σε μεγάλης κλίμακας συστήματα. Το Ansible σχεδιάστηκε από τον Michael DeHaan και δημιουργήθηκε η εταιρία Ansible Inc. για την υποστήριξη του λογισμικού. Η Red Hat απέκτησε αργότερα την εταιρία και το λογισμικό και συνεχίζει μέχρι σήμερα να το διαθέτει και να υποστηρίζει την λειτουργία του. [1]

Το Ansible υποστηρίζει Python και είναι συνήθως σε YAML (γλώσσα συσχετισμού δεδομένων, εύκολα αναγνωρίσιμη από τον άνθρωπο και συνήθως χρησιμοποιείται για τα configuration αρχεία, αλλά και σε πολλές εφαρμογές που αποθηκεύουν δεδομένα) ή σε ειδικά DSL (Domain Specific Language) που ονομάζονται PyDSL. Είναι σχεδιασμένο να δουλεύει με χρήση του SSH πρωτοκόλλου και των ήδη εγκατεστημένων στα συστήματά μας κλειδιών του για ασφαλή μεταφορά.

Το Ansible βρίσκεται σε έναν κεντρικό server τον οποίο ονομάζουμε *control server* και μπορεί να εκτελεί εντολές και λειτουργίες στον server που επιθυμούμε, ο οποίος ονομάζεται *target server.*

Το Ansible λειτουργεί στέλνοντας Ansible scripts, τα λεγόμενα *Playbooks,* μέσω SSH στον target server. Τα scripts αυτά είναι γραμμένα σε YAML και εκτελούνται στον target server. Το Ansible μπορεί να εγκαταστήσει αρχεία και να διαχειριστεί πακέτα καθώς και πολλούς άλλους πόρους, συμπεριλαμβανομένων και πόρων για τη δημιουργία της δομής ενός Cloud, εκτελώντας τα Playbooks.

Το Playbook του Ansible αποτελείται από συνεχόμενα modules τα οποία και εκτελούνται στον target server. Τα modules αυτά μπορούν να διαχειριστούν αρχεία, πακέτα αλλά και υπηρεσίες. Ουσιαστικά τα Playbooks του Ansible περιέχουν “παιχνίδια” (plays) που με τη σειρά τους περιέχουν ξεχωριστές εργασίες (tasks). Τα tasks εκτελούνται με τη σειρά που είναι γραμμένα και με τη σειρά που καλούνται από το Playbook. Οι χειριστές (handlers) ενεργοποιούνται από τα tasks και εκτελούνται μία φορά, στο τέλος των plays. Η δομή ενός Playbook φαίνεται στην παρακάτω εικόνα.



Εικόνα 3: Δομή του Playbook της Ansible.

Για να εγκαταστήσουμε το Ansible στο σύστημά μας, μπορούμε να εκτελέσουμε την παρακάτω εντολή σε Debian based συστήματα:

```
#  sudo apt-get install -y ansible
```

Ή την παρακάτω εντολή σε RHEL/CentOS συστήματα:

```
#  yum -y install ansible
```

Για παράδειγμα, εάν θέλουμε να εγκαταστήσουμε έναν DNS server με χρήση Playbook της Ansible, τότε θα πρέπει στον Control Server όπου βρισκόμαστε και είναι εγκατεστημένο το Ansible, να φτιάξουμε και να εκτελέσουμε το Playbook, στο οποίο θα έχουν δηλωθεί τα plays, τα tasks, οι handlers, οι ρόλοι και τα groups (όταν ενδιαφερόμαστε για συγκεκριμένους hosts και τους έχουμε δηλωμένους ξεχωριστά) και ό,τι ενέργειες εκτελέσει εντέλει το Playbook θα εκτελεστεί, εγκατασταθεί και ρυθμιστεί πλήρως στον Target Server που μας ενδιαφέρει.

Ένα παράδειγμα ενός απλού Playbook είναι το παρακάτω.

```
---
- hosts: hostB
  tasks:
      - name: Create file
        file:
            path: /tmp/yallo
            state: touch

–hosts: hostB
  sudo: yes
  tasks:
      - name: Create user
        user:
            name: mario
            shell: /bin/zsh

      - name: Install zlib
        yum:
            name: zlib
            state: latest
```

Περιέχει δύο (2) plays και τρία (3) tasks. Το κάθε play ξεκινάει με – **hosts** και αφορά συγκεκριμένους hosts στους ή στον οποίο εκτελείται. Εδώ, το πρώτο play θα εκτελεστεί στον hostB και απλά δημιουργεί ένα

αρχείο που ονομάζεται yallo και βρίσκεται στον φάκελο /tmp. Με την εντολή `state: touch` λέμε στο Ansible ότι αν υπάρχει το αρχείο ήδη, να μην κάνει τίποτα, αλλιώς, αν δεν υπάρχει, να το δημιουργήσει.

Στο δεύτερο play, το οποίο και αυτό εκτελείται στον hostB, ενεργοποιούμε κατευθείαν την επιλογή ότι οι παρακάτω εντολές θα εκτελεστούν με το `sudo` ενεργό ώστε να εκτελεστούν οι εντολές που περιέχονται στα δύο tasks. Στο πρώτο task δημιουργούμε χρήστη που ονομάζεται mario και του δίνουμε επιλεγμένο πρόγραμμα για την κονσόλα το `zsh`. Στο δεύτερο task εγκαθίσταται στο σύστημα το πακέτο `zlib` στην τελευταία του έκδοση.

Γενικά, κάθε YAML αρχείο ξεκινάει με τρείς παύλες (---) και κάθε play ξεκινάει με **`- hosts :.`**

Περισσότερες πληροφορίες για το Ansible μπορούμε να βρούμε στην ιστοσελίδα https://docs.ansible.com/

1.4. Gdeploy

Το gdeploy είναι ένα εργαλείο που κάνει χρήση των λειτουργιών του Ansible και αναπτύχθηκε αρχικά με σκοπό να κάνει πιο εύκολη την δημιουργία και εγκατάσταση GlusterFS clusters, αλλά με τον καιρό εξελίχθηκε στο να είναι ικανό να εκτελεί πολλές ακόμα λειτουργίες. Τώρα πια το gdeploy μπορεί, σε αριθμό από hosts, να κάνει τα εξής:

- να δημιουργεί physical ή logical volumes,
- να ομαδοποιεί τα volumes σε groups,
- να εγκαθιστά πακέτα,
- να κάνει εγγραφή σε κανάλια RHN (Red Hat Network),
- να εκτελεί εντολές στην κονσόλα του συστήματος,
- να δημιουργεί GlusterFS volumes και πολλά άλλα

[2]

1.5. GlusterFS

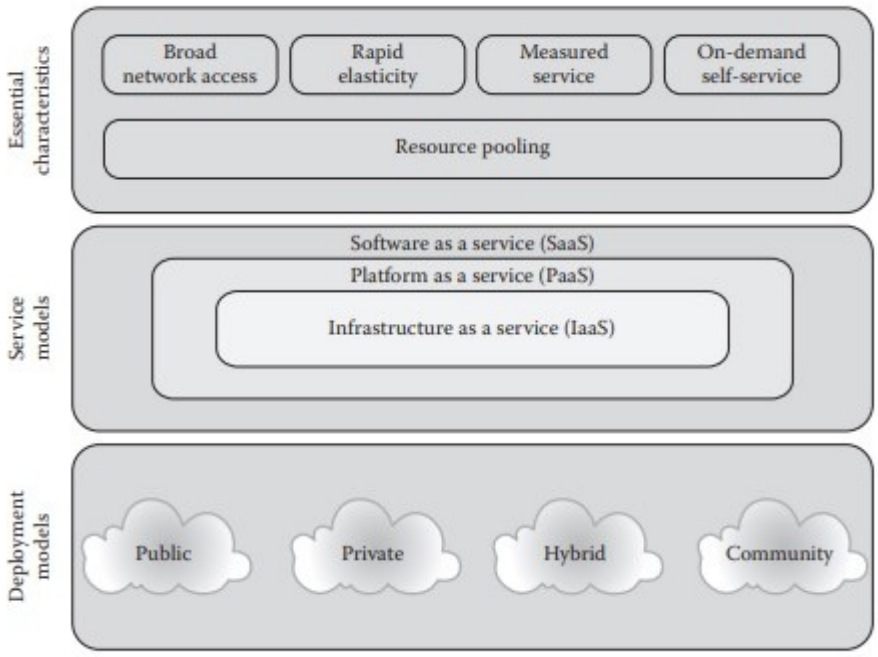
Το GlusterFS είναι ένας τύπος κατανεμημένου συστήματος αρχείων δικτύου, ή αλλιώς ένας τύπος Distributed Network File System (DNFS), με το οποίο, όπως και με οποιοδήποτε άλλον τύπο DNFS, μπορούμε να δημιουργήσουμε ένα σύστημα αρχείων που μπορεί να επεκταθεί και να κατανεμηθεί σε άλλα συστήματα, κάτω από το ίδιο global namespace.

Το GlusterFS είναι ένα ανοιχτού τύπου λογισμικό και μπορεί να συγκεντρώνει τους αποθηκευτικούς χώρους από διαφορετικούς server και να τους διαθέτει κάτω από το ίδιο global namespace, βοηθώντας έτσι στη δημιουργία χώρων που μπορούν να χρησιμοποιηθούν για παροχή αποθηκευτικού χώρου νέφους (Cloud) αλλά και για streaming πολυμέσων για τους χρήστες μας. Επίσης, με το GlusterFS μπορούμε να κάνουμε χρήση απλού υλικού (hardware) και όχι εξειδικευμένου εξοπλισμού που συνήθως κοστίζει ακριβά.

Με το GlusterFS μπορούμε να έχουμε έναν αριθμό servers οι οποίοι θα προσφέρουν τον αποθηκευτικό τους χώρο χωρίς ο χρήστης να χρειάζεται να γνωρίζει σε ποιον ακριβώς server είναι αποθηκευμένα τα δεδομένα του, αλλά να είναι σίγουρος ότι τα δεδομένα του δεν πρόκειται να υποστούν καμία διαφθορά ή να κατακερματιστούν και να μην μπορεί να έχει πρόσβαση σε αυτά. Επίσης, για να είμαστε σίγουροι για την ασφάλεια των δεδομένων και με σκοπό να μπορεί μόνο ο ιδιοκτήτης τους να τα προσπελάσει, μπορούν να εφαρμοστούν τεχνικές κρυπτογράφησης πάνω τους ώστε να είναι ασφαλή και απροσπέλαστα από τρίτους. [3]

1.6. Cloud και CloudStack

Υπολογιστικό Νέφος ή αλλιώς Cloud Computing είναι ένα μοντέλο που επιτρέπει την πρόσβαση στο δίκτυο σε μία κοινόχρηστη ομάδα από διαμορφωμένους υπολογιστικούς πόρους (π.χ. δίκτυα, διακομιστές αποθηκευτικούς χώρους, εφαρμογές και υπηρεσίες) που μπορούν να παρασχεθούν γρήγορα και να κυκλοφορήσουν με ελάχιστη προσπάθεια διαχείρισης ή αλληλεπίδραση παρόχου υπηρεσιών. Αυτό το μοντέλο Cloud προωθεί τη διαθεσιμότητα και αποτελείται από πέντε βασικά χαρακτηριστικά, τρία μοντέλα υπηρεσιών και τέσσερα μοντέλα ανάπτυξης. [4]



Εικόνα 4: Τα βασικά χαρακτηριστικά, μοντέλα υπηρεσιών και τα μοντέλα ανάπτυξης του Cloud.

Με το Cloud οι χρήστες μπορούν να αποθηκεύουν, επεξεργάζονται δεδομένα, λογισμικό αλλά και υπηρεσίες απομακρυσμένα μέσω του Διαδικτύου και ενώ τα δεδομένα, το λογισμικό και οι υπηρεσίες προσφέρονται από κεντρικά Data Centers. Υπηρεσίες όπως η On-Demand παροχή εικονικών μηχανών, το ηλεκτρονικό ταχυδρομείο ή τα κοινωνικά δίκτυα συχνά βασίζονται στην τεχνολογία του Cloud. Οι χρήστες μπορούν να εξοικονομούν πόρους από την αγορά και συντήρηση λογισμικού, ακριβών routers και μεγάλων εγκαταστάσεων αποθήκευσης δεδομένων.[6][7]

Τα βασικά μοντέλα υπηρεσιών που προσφέρει το Cloud είναι τα εξής:

- Λογισμικό ως Υπηρεσία ή Software as a Service (SaaS), π.χ. είναι η Amazon, το Azure Cloud και το Google Cloud,
- Δομή ως Υπηρεσία ή Infrastructure as a Service (IaaS), π.χ. είναι το Gmail,
- Πλατφόρμα ως Υπηρεσία ή Platform as a Service (PaaS), π.χ. είναι η δυνατότητα να εκτελούμε το Salesforce απομακρυσμένα.

Επιπλέον, έχουν πια οριστεί και άλλες υπηρεσίες από την ITU-T Y3500 στο Cloud Computing Overview and Vocabulary (Αύγ. 2014), όπως είναι οι:

- CaaS (Communication as a Service),
- CompaaS (Compute as a Service),
- DSaaS (Data Storage as a Service),
- NaaS (Network as a Service),
- DbaaS (DataBase as a Service),
- DesktopaaS (Desktop as a Service),
- EmailaaS (Email as a Service),
- IdentityaaS (Identity as a Service),
- MngtaaS (Management as a Service),
- SecurityaaS (Security as a Service),
- XaaS (Anything or Everything or XaaS), να είναι η τελευταία πρόσφατη υπηρεσία που αναδύεται.

Με την εμφάνιση της τεχνολογίας του Cloud, εμφανίστηκε και η επιθυμία για την τεχνολογία του Open Cloud, με το οποίο θα μπορούσαμε να αποφύγουμε τον εγκλωβισμό σε μεγάλες εταιρείες και να εξαρτιόμαστε από αυτές εξ' ολοκλήρου. Αυτή ήταν και η αιτία που οδήγησε την NASA σε συνεργασία με την RACKSPACE, σε μία προσπάθεια δημιουργίας Open Cloud και τελικά στη δημιουργία του λογισμικού OpenStack το οποίο προσφέρει πληθώρα εργαλείων και τεχνολογιών και είναι εντελώς ανοιχτό και προσπελάσιμο από οποιονδήποτε ενδιαφέρεται να προσφέρει στην κοινότητα. Με το OpenStack μπορούμε να έχουμε και να παρέχουμε υπηρεσίες Data Center και αποτελεί ένα μεγάλο project από μόνο του. Λόγω της δυσκολίας που συναντάται στην εγκατάστασή του, έχει δημιουργηθεί μία πιο ελαφριά έκδοσή του, πολύ πιο γρήγορη στην εγκατάσταση, που είναι ικανή να προσφέρει την Cloud τεχνολογία και αρκετές επιπλέον υπηρεσίες και ονομάζεται CloudStack.

Λόγω χρόνου, στην πτυχιακή αυτή θα εξετάσουμε το λογισμικό CloudStack.

Το CloudStack είναι ένα ανοιχτό (open source) λογισμικό της Apache και σχεδιάστηκε για να είναι εφικτή η δημιουργία και διαχείριση μεγάλων δικτύων εικονικών μηχανών, ως εξαιρετικά διαθέσιμων και επεκτάσιμων Iaas υπηρεσιών νέφους. Το CloudStack χρησιμοποιείται από αριθμό πάροχων υπηρεσιών με σκοπό την παροχή υπηρεσιών δημόσιου νέφους, αλλά και από πολλές εταιρείες για παροχή υπηρεσιών ιδιωτικού νέφους ή ως μέρος μία ενδιάμεσης λύσης Cloud που ονομάζεται Hybrid και μπορεί να προσφέρει ιδιωτικές ή δημόσιες υπηρεσίες ανάλογα με τις ρυθμίσεις του. Συγκεκριμένα, μία εταιρία επιλέγει να

φτιάξει ένα Hybrid Cloud γιατί δεν θέλει να έχει τα συστήματά της σε κάποιο Δημόσιο Νέφος (AMAZON, Google, Azure), αλλά σε δικό της Ιδιωτικό Νέφος. Ωστόσο, θέλει όταν προκύπτει δυναμικά η ανάγκη επέκτασης να μπορεί να "ξεχυλώνει" στο Δημόσιο Νέφος μιας και δεν διαθέτει τους πόρους που είναι αναγκαίοι για να κάνει αυτή την επέκταση (π.χ. Καινούριο Data Center, 50 νέοι servers κ.λπ.)

Το CloudStack προσφέρει ένα καλό πακέτο υπηρεσιών που ενδιαφέρει αρκετές εταιρείες και οργανισμούς όπως είναι η οργάνωση εργασιών που πρέπει να γίνονται σε συγκεκριμένους χρόνους, το Δίκτυο ως Υπηρεσία (Network as a Service, NaaS) και τη διαχείριση των λογαριασμών, παρέχει ένα πλήρες και ανοιχτό εγγενές API, υπολογισμό των πόρων του συστήματος, καθώς και μία πολύ καλή διεπαφή χρήστη (User Interface, UI).

Το CloudStack υποστηρίζει τους πιο δημοφιλείς Hypervisors, δηλαδή τα Vmware, KVM, Citrix Xen Server, Xen Cloud Platform (XCP), Oracle VM Server και τον Microsoft Hyper-V. [8]

ΚΕΦΑΛΑΙΟ 2 – oVirt

2.1. Περίληψη Κεφαλαίου

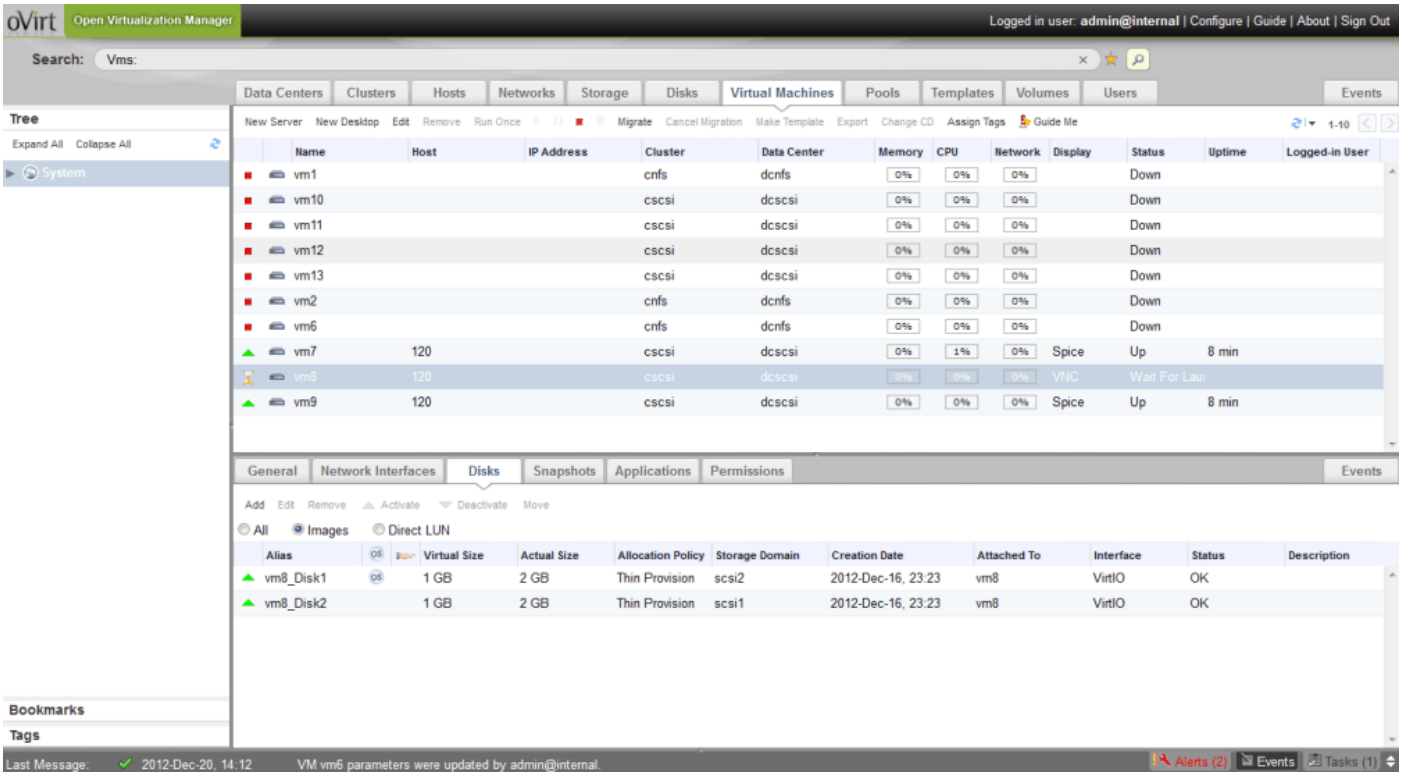
Σε αυτό το κεφάλαιο θα ανακαλύψουμε την σχετικά νέα, ανοιχτού τύπου πλατφόρμα εικονικοποίησης oVirt. Θα εξετάσουμε:

- το γραφικό αλλά και το text-based περιβάλλον εικονικοποίησης του oVirt
- το Live CD που παρέχεται για μια demo εμπειρία του περιβάλλοντος, αλλά και των δυνατοτήτων του oVirt
- την self-hosted engine του oVirt
- τα oVirt Nodes που προσφέρουν την δυνατότητα δημιουργίας χώρου Gluster
- τις Gluster δυνατότητες και πώς ο Gluster χώρος μπορεί να αποτελέσει το λεγόμενο Software Defined Storage (SDS) και να διατελέσει χρέη χώρου Cloud για τους χρήστες μας
- το εργαλείο Gdeploy που κάνει χρήση των Playbooks του Ansible
- τον τρόπο δημιουργίας χρηστών με συγκεκριμένα δικαιώματα (π.χ. πρόσβαση στο Cockpit για έλεγχο του δικού τους μικρού Data Center)

2.2. Τι Είναι το oVirt

Το oVirt είναι μία ολοκληρωμένη πλατφόρμα διαχείρισης Virtualization και μπορούμε μέσω της διεπαφής διαχείρισης (management interface), η οποία ονομάζεται oVirt Engine, να διαχειριστούμε κόμβους υλικού, αποθηκευτικούς και δικτυακούς πόρους των συστημάτων μας καθώς και να αναπτύξουμε και να παρακολουθήσουμε τις εικονικές μηχανές καθώς εκτελούνται στο κέντρο δεδομένων μας. Αναπτύσσεται ως λογισμικό ανοιχτού κώδικα και είναι licensed. Το oVirt βασίζεται πάνω στον ισχυρό KVM Hypervisor (Kernel Virtual Machine Hypervisor) της Linux και στον RHEV-M Management Server, που διατίθεται από την Red Hat στην κοινότητα ανοιχτού κώδικα.

Περιλαμβάνει πλούσιες διεπαφές σε μορφή παραθυρικού περιβάλλοντος, και συγκεκριμένα web-browser interface, τόσο για τους διαχειριστές του όσο και για τους απλούς χρήστες. Προσφέρει υποστήριξη για ζωντανή μετάδοση και «μετανάστευση» των εικονικών μηχανών από τους κεντρικούς υπολογιστές στους χώρους αποθήκευσής τους καθώς και ολοκληρωμένη διαχείριση των κεντρικών υπολογιστών (hosts), του χώρου αποθήκευσης και της διαχείρισης δικτύου. Επίσης, όπως θα δούμε και παρακάτω, σε περίπτωση βλάβης του κεντρικού υπολογιστή υπάρχει υψηλή διαθεσιμότητα των εικονικών μηχανών, χωρίς να κινδυνεύει το σύστημά μας ή η ομαλή λειτουργία του χρήστη.



Εικόνα 3: Παραθυρικό web-browser περιβάλλον oVirt Virtualization Manager

Το oVirt είναι εννοιολογικά παρόμοιο με το vSphere της VMware. Το oVirt χρησιμεύει ως υπόβαθρο για τα προϊόντα Virtualization της Red Hat και είναι το "upstream" project όπου αναπτύσσονται νέα χαρακτηριστικά πριν από την ένταξή τους σε αυτή την υποστηριζόμενη προσφορά προϊόντων.

Το oVirt είναι από τα πιο πρόσφατα project Virtualization. Με το oVirt η κοινότητα προσπαθεί να ανταγωνιστεί την καθιερωμένη επιλογή στο Virtualization Management, VMware. Νέες εκδόσεις, σταθερές ή προς παραγωγή οι οποίες ακόμα εξετάζονται, μπορούν και γίνονται διαθέσιμες άμεσα στους χρήστες του σχεδόν σε μηνιαία βάση. Συγκεκριμένα το oVirt αναπτύσσεται έτσι ώστε να μπορεί να μας προσφέρει τα εξής:

- Διαχείριση πολλαπλών εικονικών μηχανών.
- Εξελιγμένη διεπαφή χρήστη επιτρέπει τη διαχείριση όλων των πτυχών του κέντρου δεδομένων σας.
- Επιλογή μέσων κατανομής VM σε κεντρικούς υπολογιστές: εγχειρίδιο, "βελτιστοποιημένο", καρφωμένο.
- Ζωντανή μετανάστευση VMs από έναν hypervisor σε άλλο.
- Δυνατότητα πρόσθεσης νέων κόμβων Hypervisor εύκολα και κεντρικά.
- Δυνατότητα παρακολούθησης τη χρήση πόρων σε VM.
- Διαχείριση των ποσοστώσεων για τη χρήση των πόρων των κεντρικών συστημάτων (αποθήκευση, υπολογισμός, δίκτυο).
- Παροχή κονσόλας αυτοεξυπηρέτησης τόσο για απλές όσο και για προηγμένες περιπτώσεις χρήσης.
- Είναι χτισμένο πάνω στον KVM Hypervisor της Linux, οπότε και μοιράζεται πολλά στοιχεία με το, πιο γνωστό στους χρήστες, KVM.
- Είναι ανοιχτού τύπου λογισμικό (Open source), οπότε και μπορούμε να συμμετέχουμε στον σχεδιασμό και την ανάπτυξη του έργου αυτού, καταθέτοντας είτε bugs είτε προσφέροντας λύση για κάποιο πρόβλημα ή συμμετέχοντας ενεργά στον περαιτέρω σχεδιασμό του.

2.2.1. Συστατικά Στοιχεία του oVirt

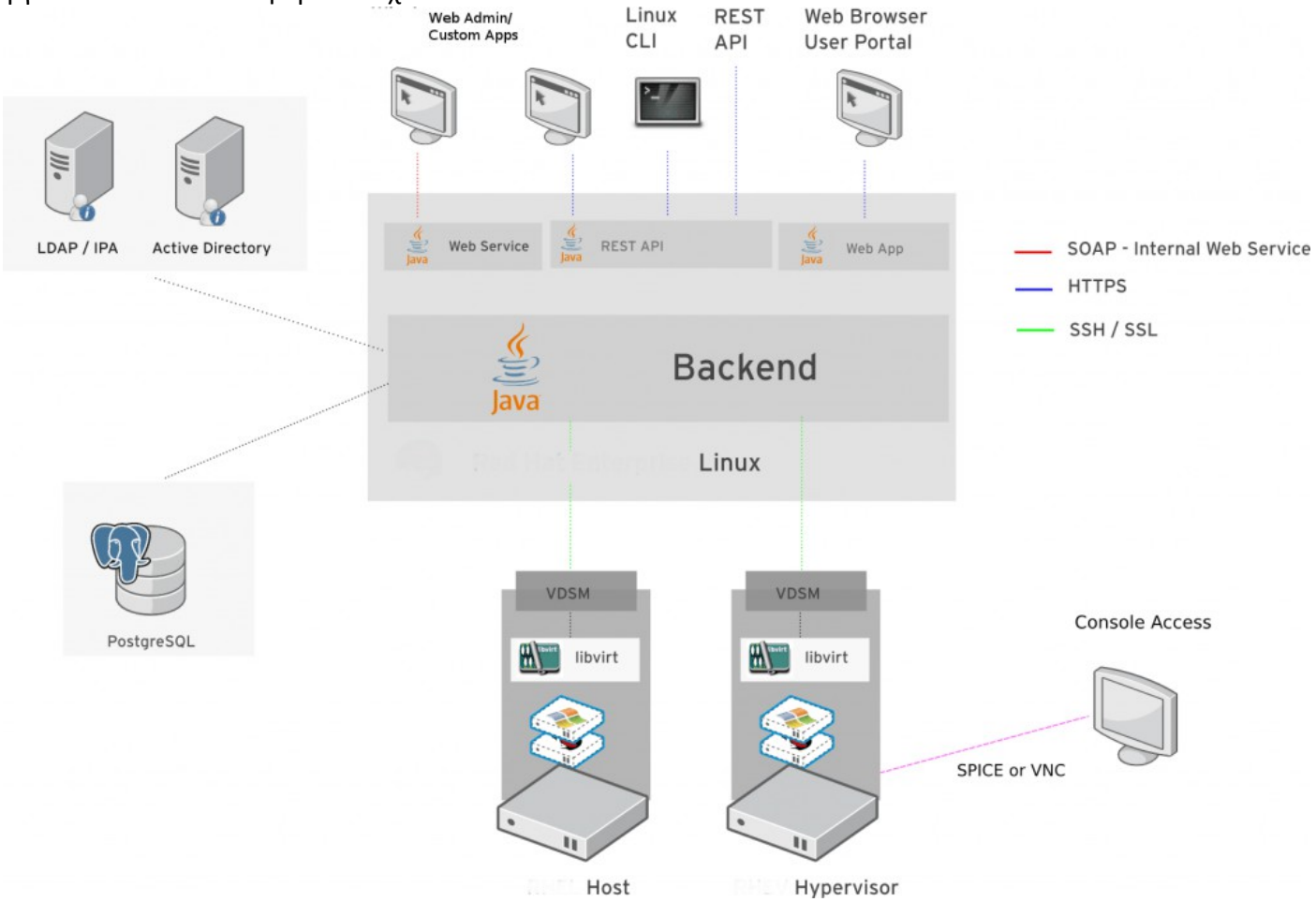
2.2.1.1. Αρχιτεκτονική του oVirt

Μια τυπική εγκατάσταση oVirt αποτελείται από τρία στοιχεία:

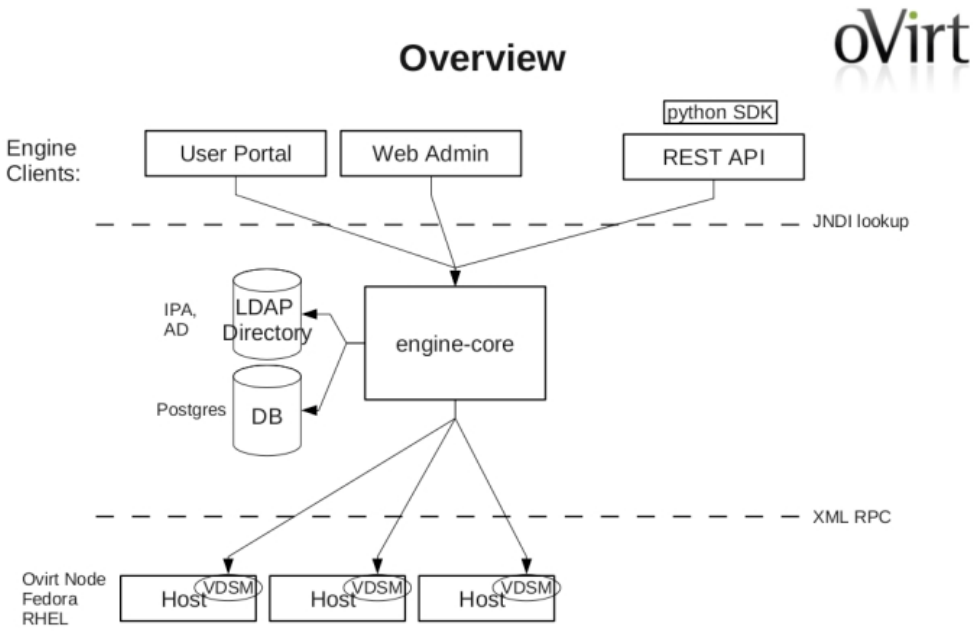
1. Το oVirt-Engine το οποίο χρησιμοποιείται για την ανάπτυξη, δημιουργία, παρακολούθηση, μετακίνηση και διακοπή εικόνων VM, τη ρύθμιση παραμέτρων αποθήκευσης, δικτύου κ.λπ.
2. Έναν ή περισσότερους κεντρικούς υπολογιστές (κόμβοι), ή αλλιώς hosts ή nodes, στους οποίους και τρέχουμε τις εικονικές μηχανές (VM)
3. Έναν ή περισσότερους κόμβους αποθήκευσης, οι οποίοι και συγκρατούν τις εικόνες και τα ISO αρχεία που αντιστοιχούν σε αυτά τα VM

Επίσης, συνήθως αναπτύσσεται μια υπηρεσία ταυτότητας πέρα από το oVirt-Engine, το οποίο χρησιμοποιείται για την εξακρίβωση της ταυτότητας των χρηστών και των διαχειριστών για το oVirt-Engine.

Τα διαφορετικά διαγράμματα και περιγραφές που ακολουθούν αντιπροσωπεύουν την αρχιτεκτονική του έργου oVirt και τα διάφορα στοιχεία του.



Εικόνα 4: Αρχιτεκτονική του oVirt



Εικόνα 5: Απλοποιημένη αρχιτκτονική του oVirt

2.2.1.2. oVirt Engine

Κεντρική μηχανή διαχείρισης εικονικοποίησης για επιχειρήσεις με γραφική κονσόλα διαχείρισης και προγραμματιστικές διεπαφές.

Το backend του oVirt-Engine είναι γραμμένο σε Java, ενώ το frontend αναπτύσσεται με το GWT web toolkit. Το oVirt-Engine τρέχει πάνω από τον server εφαρμογών WildFly (πρώην γνωστό ως JBoss). Το frontend μπορεί να προσπελαστεί μέσω μιας πύλης webadmin για διαχείριση ή μιας πύλης χρήστη με προνόμια, και διαθέτει λειτουργίες οι οποίες μπορούν να ρυθμιστούν με ακρίβεια από τον διαχειριστή του συστήματος. Η διαχείριση των χρηστών μπορεί να γίνει τοπικά ή ενσωματώνοντας το oVirt με υπηρεσίες LDAP ή AD. Το oVirt-Engine αποθηκεύει δεδομένα σε μια βάση δεδομένων PostgreSQL. Η αποθήκευση δεδομένων και οι δυνατότητες αναφορών εξαρτώνται από πρόσθετες βάσεις δεδομένων ιστορικού και αναφορών που μπορούν να δημιουργηθούν κατ' επιλογή κατά τη διάρκεια της διαδικασίας εγκατάστασης του oVirt. Το RESTful API είναι διαθέσιμο για την προσαρμογή ή την προσθήκη χαρακτηριστικών του κινητήρα. [9]

2.2.1.3. oVirt Node

Εξαιρετικά κλιμακωτός, image-based Hypervisor μικρής αποτύπωσης (<200MB), με ελάχιστο αποτύπωμα ασφαλείας.

Οι κόμβοι (nodes) είναι διανομές Linux με εγκατεστημένα τα VDSM και libvirt, μαζί με μερικά επιπλέον πακέτα για εύκολη ενεργοποίηση της δικτύωσης και άλλων υπηρεσιών συστήματος. Οι διανομές Linux που υποστηρίζει το oVirt μέχρι και σήμερα είναι το Fedora 17 ή το oVirt-node, το οποίο είναι ουσιαστικά μια απογυμνωμένη διανομή που περιέχει μόνο αρκετά στοιχεία από τη διανομή CentOS της Red Hat που επιτρέπουν το Virtualiation.

Οι κόμβοι αποθήκευσης μπορούν να χρησιμοποιηθούν για αποθήκευση σε μορφή μπλοκ ή αρχείων και μπορούν να είναι τοπικοί ή απομακρυσμένοι, προσπελάσιμοι μέσω του NFS (Network File System). Οι τεχνολογίες αποθήκευσης όπως είναι το Gluster, στο οποίο θα αναφερθούμε ξανά αργότερα και θα το

εξετάσουμε, υποστηρίζονται μέσω του τύπου αποθήκευσης POSIXFS. Οι κόμβοι αποθήκευσης ομαδοποιούνται σε πισίνες αποθήκευσης (storage pools), οι οποίες εξασφαλίζουν υψηλή διαθεσιμότητα και πλεόνασμα σε αποθηκευτικούς πόρους.

Ένα oVirt node είναι ένας server που εκτελεί RHEL, CentOS, Fedora ή πειραματικά Debian, με ενεργοποιημένο τον KVM Hypervisor και έναν δαίμονα VDSM (Virtual Desktop and Server Manager) γραμμένο σε Python. Η διαχείριση των πόρων που ξεκινούν από μια πύλη web admin αποστέλλεται μέσω του backend του Engine που εκπέμπει κατάλληλες κλήσεις στον δαίμονα VDSM. Το VDSM ελέγχει όλους τους πόρους που είναι διαθέσιμοι στον node (υπολογισμός, αποθήκευση, δικτύωση) και εικονικές μηχανές που εκτελούνται σε αυτόν και είναι επίσης υπεύθυνος για την παροχή ανατροφοδότησης στο oVirt Engine για όλες τις λειτουργίες που έχουν ξεκινήσει. Πολλαπλοί oVirt node μπορούν να συγκεντρωθούν από την πύλη web admin του oVirt Engine για την ενίσχυση του RAS.

Ο κινητήρας oVirt μπορεί να εγκατασταθεί σε έναν αυτόνομο server ή μπορεί να φιλοξενηθεί σε ένα σύμπλεγμα nodes μέσα σε μια virual machine (Self-Hosted Engine). Η Self-Hosted Engine μπορεί να εγκατασταθεί χειροκίνητα ή να αναπτυχθεί αυτόματα μέσω μιας εικονικής συσκευής. [10]

2.2.2. Προϋποθέσεις και Απαιτούμενα Εγκατάστασης και Χρήσης

Στη συνέχεια θα αναφερθούμε στις προϋποθέσεις και στα απαιτούμενα που πρέπει να έχουν τα συστήματά μας για να εγκατασταθούν αντίστοιχα το oVirt Engine, oVirt Node, του Hypervisor, αλλά και του Firewall.

2.2.2.1. oVirt Engine

2.2.2.1.1. Απαιτήσεις Υλικού

Οι ελάχιστες και συνιστώμενες απαιτήσεις σε υλικό για το oVirt Engine που περιγράφονται εδώ αφορούν τυπικές μικρού ή μεσαίου μεγέθους εγκαταστάσεις και ποικίλουν ανάλογα με το μέγεθος αλλά και τον φόρτο εργασίας που επιθυμούμε να εκτελούν.
Το oVirt Engine εκτελείται σε λειτουργικά περιβάλλοντα Enterprise Linux όπως είναι τα CentOS Linux, Scientific Linux και Red Hat Enterprise Linux.

Πίνακας 1: Απαιτούμενοι πόροι συστήματος

Πόροι Συστήματος	**Ελάχιστα Απαιτούμενα**	**Προτιμώμενα Απαιτούμενα**
CPU	2-πύρηνος CPU	4-πύρηνος CPU
Μνήμη	4 GB διαθέσιμη μνήμη RAM εάν το Data Warehouse δεν έχει εγκατασταθεί και η μνήμη δεν καταναλώνεται από άλλες διαδικασίες του συστήματος	16 GB μνήμη RAM
Σκληρός Δίσκος	25 GB τοπικά προσπελάσιμου, εγράψιμου χώρου στο δίσκο	50 GB τοπικά προσπελάσιμου, εγράψιμου χώρου στο δίσκο
Interface Δικτύου	1 κάρτα δικτύου (Network Interface Card -NIC) με διαθέσιμο εύρος ζώνης (bandwidth) τουλάχιστον 1 Gbps	1 κάρτα δικτύου (Network Interface Card -NIC) με διαθέσιμο εύρος ζώνης (bandwidth) τουλάχιστον 1 Gbps

2.2.2.1.2. Απαιτήσεις Προγράμματος Περιήγησης

Το oVirt ακολουθεί τους κανονισμούς του Red Hat Customer Portal Browser Support Policy. Συνίσταται η χρήση των κάτωθι προγραμμάτων περιήγησης, καθώτι αναβαθμίζονται αυτόματα:

- Mozilla Firefox
- Google Chrome
- Apple Safari
- Microsoft Internet Explorer 11
- Microsoft Edge

2.2.2.1.3. Απαιτήσεις Συστήματος Πελάτη

Οι κονσόλες των εικονικών μηχανών μπορούν μόνο να προσπελαστούν από το υποστηριζόμενο Remote Viewer (`virt-viewer`) είτε μέσω Enterprise Linux ή Windows συστήματα. Για την εγκατάσταση του `virt-viewer` απαιτούνται δικαιώματα διαχειριστή.

Η πρόσβαση στην κονσόλα SPICE είναι διαθέσιμη μόνο μέσω άλλων λειτουργικών συστημάτων όπως είναι το OS X, μέσω του μη υποστηριζόμενου SPICE HTML5 browser client.
Οι υποστηριζόμενοι QXL drivers είναι διαθέσιμοι για Enterprise Linux, Windows XP και Windows 7.
Το SPICE υποδιαιρείται στα εξής επίπεδα:

- Επίπεδο 1: Λειτουργικά συστήματα στα οποία ο remote-viewer έχει δοκιμαστεί πλήρως.
- Επίπεδο 2: Λειτουργικά συστήματα στα οποία ο remote-viewer έχει δοκιμαστεί μερικώς και είναι πιθανό ότι θα λειτουργήσει όπως πρέπει.

Πίνακας 2: Επίπεδα SPICE

Υποστηριζόμενο Επίπεδο	**Λειτουργικό Σύστημα**	**Υποστήριξη SPICE**
Επίπεδο 1	Enterprise Linux 7	Πλήρως υποστηριζόμενο σε Enterprise Linux 7.2 και άνω
	Microsoft Windows 7	Πλήρως υποστηριζόμενο σε Microsoft Windows 7
Επίπεδο 2	Microsoft Windows 8	Υποστηριζόμενο όταν το `spice-vdagent` εκτελείται στο λειτουργικό σύστημα του πελάτη
	Microsoft Windows 10	Υποστηριζόμενο όταν το `spice-vdagent` εκτελείται στο λειτουργικό σύστημα του πελάτη

2.2.2.1.4 Απαιτήσεις Λειτουργικού Συστήματος

Το oVirt Engine εγκαθίσταται «επάνω» σε μία βασική εγκατάσταση των Enterprise Linux (είτε RHEL ή CentOS) 7. Δεν εγκαθιστούμε κανένα από τα επιπλέον πακέτα μετά την εγκατάσταση της βασικής (`base`) έκδοσης, καθότι μπορεί να προκληθούν προβλήματα εξαρτήσεων όταν προσπαθήσουμε να κάνουμε την εγκατάσταση του `oVirt-Engine` και των δικών του, απαιτούμενων, πακέτων.

2.2.2.2. Hypervisor

2.2.2.2.1. Απαιτήσεις CPU

Όλες οι CPUs πρέπει να έχουν υποστήριξη για Intel® 64 ή AMD64 επεκτάσεις, το AMD-V™ ή το Intel VT® με ενεργοποιημένες τις επεκτάσεις για virtualization υλικού. Επίσης, απαιτείται υποστήριξη για το No eXecute flag (NX).

Πίνακας 3: Υποστηριζόμενα Μοντέλα CPU για τον Hypervisor

AMD	Intel	IBM
AMD Opteron G1	Intel Conroe	IBM POWER8
AMD Opteron G2	Intel Penryn AMD Opteron AMD Opteron G4 AMD Opteron G5	
AMD Opteron G3	Intel Nehalem	
AMD Opteron G4	Intel Westmere	
AMD Opteron G5	Intel Sandybridge	
	Intel Haswell	
	Intel Haswell-no TSX	
	Intel Skylake	

Δοκιμές Εάν ο Επεξεργαστής Υποστηρίζει τις Απαιτούμενες Σημαίες

1. Ενεργοποιούμε τις δυνατότητες Virtualization από το BIOS. Απενεργοποιούμε και επανεκκινούμε τον host για να εξασφαλίσουμε ότι η αλλαγή που κάναμε εμμένει.
2. Στα Enterprise Linux ή στο oVirt Node στην οθόνη του boot, πιέζουμε οποιοδήποτε πλήκτρο και διαλέγουμε την επιλογή `Boot` ή την `Boot with serial console`, με την οποία όταν ανοίγει ο host με την κονσόλα του συστήματος ενεργοποιημένη.
3. Πιέζουμε το Tab και επεξεργαζόμαστε τις παραμέτρους για τον πυρήνα του συστήματος. Παραδείγματος χάριν, για τους server HP G5 σε λειτουργικό συστημα CentOS ή oVirt Node χρειαζεται να δώσουμε εδώ τις παραμέτρους **`hpsa.hpsa_simple_mode=1 hpsa.hpsa_allow_any=1`** καθώς οι drivers για τα μηχανήματα HP δεν παρέχονται από τα CentOS και αυτόματα δίνονται από το σύστημα οι drivers cciss, οι οποίοι είναι πλέον παρωχημένοι αλλά δεν παρέχονται από τα CentOS.
4. Σιγουρεύουμε ότι μετά την τελευταία παράμετρο, δίνουμε ένα κενό ακόμα, και προσθέτουμε την παράμετρο rescue.
5. Πατάμε Enter και κάνουμε το boot σε rescue mode.
6. Όταν εμφανίζεται το prompt, παρατηρούμε εάν ο επεξεργαστής μας έχει τις απαραίτητες επεκτάσεις και εάν είναι ενεργοποιημένες εκτελώντας την παρακάτω εντολή :

```
# grep -E 'svm|vmx' /proc/cpuinfo | grep nx
```

Εάν εμφανιστεί οποιοδήποτε αποτέλεσμα, τότε ο επεξεργαστής μας έχει το απαραίτητο υλικό, ικανό για virtualization. Εάν δεν εμφανιστεί κανένα αποτέλεσμα, τότε ακομα υπάρχει πιθανότητα να έχουμε το κατάλληλο υλικό, καθώς σε μερικές περιπτώσεις οι εταιρείες παραγωγής τους απενεργοποιούν τις επεκτάσεις του virtualization στο BIOS. Εάν για οποιονδήποτε λόγο πιστεύουμε ότι κάτι παρόμοιο συμβαίνει, συμβουλευόμαστε το BIOS του συστήματος και το βιβλίο οδηγιών της μητρικής μας κάρτας που παρέχεται από τον κατασκευαστή.

2.2.2.2.2. Απαιτήσεις Μνήμης

Η μνήμη RAM που απαιτείται εξαρτάται από τις απαιτήσεις του λειτουργικού συστήματος του guest συστήματος, τις απαιτήσεις του guest συστήματος για τις εφαρμογές του, την χρήση μνήμης του και την χρήση που θα έχει από τους επισκέπτες. Επιπλέον, μπορεί να χρειαστεί να συνυπολογίσουμε ότι το KVM

είναι ικανό να κάνει overcommit την φυσική μνήμη RAM για τους virtualized guests. Αυτό μας επιτρέπει να προμηθεύουμε τους επισκέπτες με πόρους μνήμης οι οποίοι δεν είναι φυσικά υπαρκτοί, εφόσον τα συστήματα δεν βρίσκονται σε μέγιστο φόρτο εργασίας. Το KVM καταφέρνει να το κάνει αυτό διαμοιράζοντας τη μνήμη στους επισκέπτες όπως απαιτείται και μεταφέροντας τις αχρησιμοποίητες θέσεις στο swap.

Πίνακας 4: Απαιτήσεις Μνήμης

Ελάχιστη Μνήμη	Μέγιστη Μνήμη
2 GB RAM 2 TB RAM	2 TB RAM

2.2.2.2.3. Απαιτήσεις Αποθηκευτικού Χώρου

Οι hosts απαιτούν να υπάρχει αποθηκευτικός χώρος για να αποθηκεύονται οι ρυθμίσεις (configuration), τα logs, τα kernel dumps αλλά και για χρήση ώς χώρο swap. Οι απαιτήσεις για αποθηκευτικό χώρο στα Enterprise Linux, εξαρτάται από τη ποσότητα του αποθηκευτικού χώρου που χρησιμοποιείται από το ήδη υπάρχον configuration του συστήματος. Ωστόσο, οι απαιτήσεις αυτές αναμένεται να ξεπερνούν αυτές που έχει ο oVirt Node.

Πίνακας 5: Ελάχιστες Απαιτήσεις Αποθηκευτικού Χώρου του oVirt Node

/	/boot	/var	swap	Ελάχιστο Σύνολο
6 GB	1 GB	15 GB	1 GB	23 GB

Εάν εγκαθιστούμε και το oVirt Engine Virtual Appliance ώστε να εγκαταστήσουμε την self-hosted Engine, τότε το `/var` partition πρέπει να είναι μεγέθους τουλάχιστον 60GB .

2.2.2.2.4. Απαιτήσεις Συσκευών PCI

Οι hosts μας πρέπει να διαθέτουν τουλάχιστον ένα interface δικτύου με ελάχιστο εύρος ζώνης (bandwidth) 1Gbps. Προτείνεται ο κάθε ένας από τους hosts να έχουν δύο κάρτες δικτύου (NICs), οπότε και θα μπορούμε να έχουμε μία κάρτα αφιερωμένη στην υποστήριξη των εντατικών δραστηριοτήτων του δικτύου όπως είναι η διαδικασία του εικονικού machine migration. Η εκτέλεση τέτοιων λειτουργιών είναι περιορισμένες απο το εύρος ζώνης που είναι διαθέσιμο.

Μελέτη Υλικού Για Device Assignment

Εάν σκοπεύουμε να εφαρμόσουμε device assignment και PCI passthrough ώστε η εικονική συσκευή να μπορέσει να χρησιμοποιήσει μια συγκεκριμένη συσκευή PCIe από κάποιον host, πρώτα βεβαιωνόμαστε ότι οι παρακάτω προδιαγραφές είναι σε ισχύ:

- Η CPU πρέπει να υποστηρίζει IOMMU (Input-Output Memory Management Unit), όπως είναι για παράδειγμα οι VT-d ή AMD-Vi. Η IBM POWER8 CPU υποστηρίζει το IOMMU από προεπιλογή.
- Το firmware πρέπει να υποστηρίζει IOMMU.
- Τα CPU root ports που χρησιμοποιούνται πρέπει να υποστηρίζουν ACS (Access Control Services) ή παρόμοιας ικανότητας με αυτή του ACS.
- Η συσκευή PCIe πρέπει να υποστηρίζει ACS or παρόμοιας ικανότητας με του ACS.
- Συνιστάται ότι όλα τα switches και οι γέφυρες PCIe μεταξύ της συσκευής PCIe και του root port θα πρέπει να μπορούν να υποστηρίζουν ACS. Για παράδειγμα, αν ένα switch δεν υποστηρίζει ACS, τότε

όλες οι συσκευές πίσω από αυτό το switch μοιράζονται την ίδια ομάδα IOMMU, και μπορεί να ανατεθεί μόνο στο ίδιο εικονικό μηχάνημα.

- Για την υποστήριξη GPU, το Enterprise Linux 7 υποστηρίζει συσκευές PCI εκχώρησης απο NVIDIA K-Series Quadro (μοντέλο σειράς 2000 ή ανώτερο), GRID, και Tesla ως μη-VGA συσκευές γραφικών. Επί του παρόντος, έως δύο GPUs μπορούν να προσκολήθουν σε μία εικονική μηχανή επιπρόσθετες των πρότυπων, εξομοιωμένων VGA interfaces. Τα εξομοιωμένα VGA χρησιμοποιούνται για προ-reboot και εγκαταστάσεις και η NVIDIA GPU αναλαμβάνει όταν οι NVIDIA graphics drivers έχουν φορτώσει. Να σημειωθεί οτι η NVIDIA Quadro 2000 δεν είναι υποστηριζόμενη, ούτε και η Quadro K420 κάρτα.

Πάντα μπορούμε να ανατρέξουμε στους προσδιορισμούς κατασκευαστή και τα datasheets για να επιβεβαιώσουμε ότι το υλικό απαντά στις απαιτήσεις μας. Αφότου έχουμε εγκαταστήσει έναν host, μπορούμε να ανατρέξουμε στο "*Configuring a Hypervisor Host for PCI Passthrough*", στο documentation του oVirt για περισσότερες πληροφορίες πάνω στο πώς να επιτρέψουμε στο υλικο και λογισμικό του host να μπορεί να κάνει το passthrough της συσκευής. Η εντολή

```
# lspci -v
```

μπορεί να χρησιμοποιηθεί για να μας εκτυπώσει στην οθόνη πληροφορίες για τις συσκευές PCI οι οποίες ειναι ήδη εγκατεστημένες στο σύστημα.

2.2.2.3. Firewalls

2.2.2.3.1. Απαιτήσεις oVirt Engine Firewall

Το oVirt Engine απαιτεί οτι ένας αριθμός port θα ανοίξει ώστε να επιτέψει την κίνηση δικτύου μέσω του firewall του συστημάτος. Το script Engine-setup είναι ικανό να διαμορφώσει το firewall αυτόματα, αλλά αυτή η δράση αντικαθιστά οποιαδήποτε προϋπάρχουσα διαμόρφωση του firewall. Εν αντιθέση με την περίπτωση όπου υπάρχει ένα ήδη υπάρχον firewall, και πρέπει να εισάγουμε χειροκίνητα τους κανόνες του firewall που απαιτούνται απο το Engine.
Η εντολή `Engine-setup` αποθηκεύει μία λίστα των iptables κανόνων που απαιτούνται στο αρχείο `/usr/share/oVirt-Engine/conf/iptables.example`. Η διαμόρφωση firewall που έχει καταγραφεί εδώ προϋποθέτει μία προκαθορισμένη διαμόρφωση. Όπου μη προκαθορισμένες HTTP και HTTPS πύλες έχουν επιλεχθεί κατά την διάρκεια της εγκατάστασης, προσαρμόζουμε τους κανόνες του firewall ώστε να επιτρέπουν την κίνηση στο δικτύο στις πύλες που έχουν επιλεχθεί – όχι τις προκαθορισμένες πύλες (80 και 443 αντίστοιχα για τα πρωτόκολλα HTTP και HTTPS) που αναφέρονται εδώ.

Πίνακας 6: Απαιτήσεις του Firewall για το oVirt Engine

Πύλη(-ες)	Πρωτόκολλο	Πηγή	Προορισμός	Σκοπός
-	ICMP	oVirt Node(s) Enterprise Linux host(s)	oVirt Engine	Όταν εγγραφόμαστε στην oVirt Engine, οι εικονικοποιημένοι hosts στέλνουν σήματα ICMP μέσω του ping request στο Engine για να επιβεβαιώσουν ότι είναι ενεργό και διαθέσιμο.
22	TCP	Σύστημα(-τα) χρησιμοποιούμενα για	oVirt Engine	Secure Shell (SSH) access. Προαιρετικό.

		συντήρηση του Engine συμπεριλαμβανομένων των ρυθμίσεων παρασκηνίου και αναβαθμίσεων λογισμικού.		
2222	TCP	Πρόσβαση πελατών στην εικονική μηχανή μέσω σειριακής κονσόλας.	oVirt Engine	Secure Shell (SSH) πρόσβαση για να ενεργοποιηθεί σύνδεση με την σειριακή κονσόλα της εικονικής μηχανής.
80, 443	TCP	Administration Portal clients User Portal clients oVirt Node(s) Enterprise Linux host(s) REST API clients	oVirt Engine	Παροχή πρόσβασης HTTP και HTTPS στο Engine.
6100	TCP	Administration Portal clients User Portal clients	oVirt Engine	Παροχή πρόσβασης websocket proxy για τους πελάτες που χρησιμοποιούν την βασισμένη στο πρόγραμμα περιήγησης κονσόλα όταν το websocket proxy είναι ενεργό στο. Αν, παρ'όλα αυτά το websocket proxy είναι ενεργό σε έναν διαφορετικό host, αυτή η θύρα δεν χρησιμοποιείται.
7410	UDP	oVirt Node(s) Enterprise Linux host(s)	oVirt Engine	Πρέπει να είναι προσβάσιμο απο το Engine για να λαμβάνει τις αναφορές του Kdump.

Σε περιβάλλοντα όπου το oVirt Engine επίσης απαιτείται να μπορεί να εξάγει NFS αποθηκευτικό χώρο, όπως είναι το ISO Storage Domain, επιπλέον θύρες πρέπει να επιτραπούν δια μέσω του firewall. Επιτρέπουμε εξαιρέσεις firewall για τις θύρες που χρησιμοποιούμε ανάλογα με τις εκδόσεις NFS που χρησιμοποιούνται στο σύστημά μας:

Πίνακας 7: Εξαιρέσεις θυρών στο firewall ανάλογα με την έκδοση του NFS που έχουμε

Έκδοση NFS	**Πύλες**
NFSv4	• TCP πύλη `2049` για το NFS
NFSv3	• TCP και UDP πύλη `2049` για το NFS • TCP και UDP πύλη `111` (`rpcbond/sunrpc`) • TCP και UDP πύλη που καθορίζεται με `MOUNTD_PORT=”port”` • TCP και UDP πύλη που καθορίζεται με `STATD_PORT="port"` • TCP πύλη που καθορίζεται με `LOCKD_TCPPORT="port"` • UDP πύλη που καθορίζεται με `LOCKD_UDPPORT="port"`

Τα `MOUNTD_PORT`, `STATD_PORT`, `LOCKD_TCPPORT`, και `LOCKD_UDPPORT` πύλες ορίζονται στο αρχείο `/etc/sysconfig/nfs`.

2.2.2.3.2. Απαιτήσεις Firewall του Hypervisor

Στους hosts που διαθέτουν Enterprise Linux και στα oVirt Nodes, απαιτείται να επιτρέπεται η κίνηση του δικτύου από έναν αριθμό θυρών από το firewall του συστήματος. Στην περίπτωση του oVirt Node αυτές οι θύρες ρυθμίζονται και προστίθεται στη λίστα των κανόνων που επιτρέπει την κίνηση το firewall μας αυτόματα. Για τους hosts με Enterprise Linux όμως αυτοί οι κανόνες πρέπει να δοθούν χειροκίνητα ώστε να ρυθμιστεί σωστά το firewall.

Πίνακας 8: Απαιτήσεις του Firewall για τον Hypervisor

Πύλη(-ες)	**Πρωτόκολλο**	**Πηγή**	**Προορισμός**	**Σκοπός**
22	TCP	oVirt Engine	oVirt Node(s) Enterprise Linux	Secure Shell (SSH) access. Προαιρετικό.
2223	TCP	oVirt Engine	oVirt Node(s) Enterprise Linux Host(s)	Secure Shell (SSH) πρόσβαση για να ενεργοποιηθεί σύνδεση με τις σειριακές κονσόλες των εικονικών μηχανών.
161	UDP	oVirt Node(s) Enterprise Linux Host(s)	oVirt Engine	Απλό πρωτόκολλο διαχείρισης δικτύου (Simple Network Management Protocol - SNMP). Απαιτείται μόνο εάν θέλουμε το SNMP να στέλνει πληροφορίες σε έναν οι περισσότερους εξωτερικούς διαχειριστές SNMP. Προαιρετικό.
5900-6923	TCP	Χρήστες μέσω του Administration Portal Χρήστες μέσω του User Portal	oVirt Node(s) Enterprise Linux Host(s)	Απομακρυσμένη πρόσβαση στην κονσόλα μέσω των VNC και SPICE πρωτοκόλλων. Αυτές οι πύλες πρέπει να είναι ανοιχτές και να επιτρέπεται η κίνηση διαμέσω τους ώστε να διευκολύνεται η πρόσβαση των χρηστών στις εικονικές μηχανές.
5989	TCP, UDP	Common Information Model Object Manager (CIMOM)	oVirt Node(s) Enterprise Linux Host(s)	Χρησιμοποιείται από το Common Information Model Object Manager (CIMOM) για παρακολούθηση και έλεγχο των εκτελούμενων εικονικών μηχανών στον host. Απαιτείται μόνο εάν θέλουμε να χρησιμοποιήσουμε το CIMOM για έλεγχο των VMs στο εικονικό περιβάλλον. Προαιρετικό.
9090	TCP	oVirt Engine Υπολογιστές Πελατών	oVirt Node(s) Enterprise Linux Host(s)	Πρόσβαση χρηστών στο interface του Cockpit. Προαιρετικό.
16541	TCP	oVirt Node(s) Enterprise Linux Host(s)	oVirt Node(s) Enterprise Linux Host(s)	Μετανάστευση εικονικών μηχανών (virtual machine migration) με χρήση της βιβλιοθήκης `libvirt`.

Πύλη(-ες)	Πρωτόκολλο	Πηγή	Προορισμός	Σκοπός
49152 -49216	TCP	oVirt Node(s) Enterprise Linux Host(s)	oVirt Node(s) Enterprise Linux Host(s)	Λειτουργίες migration και fencing εικονικων μηχανών με χρήση του VDSM. Αυτές οι πύλες πρέπει να είναι ανοιχτές ώστε να επιτρέπεται η αυτόματη και μη μετανάστευση των VMs.
54321	TCP	oVirt Engine oVirt Node(s) Enterprise Linux Host(s)	oVirt Node(s) Enterprise Linux Host(s)	Επικοινωνία του VDSM με το Engine και άλλους virtualization hosts.

2.2.2.4. Απαιτήσεις Firewall για τον Directory Server

Το oVirt χρειάζεται έναν directory server για να μπορεί να υποστηρίξει τη δυνατότητα της αυθεντικοποίησης των χρηστών του. Ένα σύνολο θυρών πρέπει να επιτρέπονται στους κανόνες του firewall στον directory server ώστε να υποστηρίζεται η αυθεντικοποίηση GSS-API, όπως χρησιμοποιείται από το oVirt Engine.

Πίνακας 9: Απαιτήσεις Firewall στον Directory Server για Αυθεντικοποίηση των Χρηστών

Πύλη(-ες)	Πρωτόκολλο	Πηγή	Προορισμός	Σκοπός
88, 464	TCP, UDP	oVirt Engine	Directory Server	Αυθεντικοποίηση του Kerberos.
389, 636	TCP	oVirt Engine	Directory Server	Lightweight Directory Access Protocol (LDAP) μέσω SSL

2.2.2.5. Απαιτήσεις Firewall για τον Database Server

Το oVirt υποστηρίζει την χρήση απομακρυσμένου database server. Εάν σχεδιάζουμε να χρησιμοποιήσουμε έναν database server στο oVirt σύστημά μας, πρέπει να σιγουρέψουμε ότι ο απομακρυσμένος server επιτρέπει τις συνδέσεις από το Engine του oVirt.

Πύλη(-ες)	Πρωτόκολλο	Πηγή	Προορισμός	Σκοπός
5432	TCP, UDP	oVirt Engine	PostgreSQL database server	Η προκαθορισμένη πύλη για τις συνδέσεις της PostgreSQL database.

2.3. LiveCD Περιβάλλον

Το oVirt προσφέρει τη δυνατότητα να το λειτουργήσουμε χωρίς να χρειαστεί απαραίτητα να το κάνουμε εγκατάσταση στο σύστημά μας διαθέτοντας τις εκδόσεις του σε Live CD. Το Live CD δεν μας προσφέρει το πλήρες φάσμα των δυνατοτήτων που έχει το oVirt, αλλά εκδόσεις demo δίνοντας έτσι μία προεπισκόπιση των δυνατοτήτων και λειτουργιών του, ώστε να μπορέσουμε να πειραματιστούμε με αυτό χωρίς να

χρειαστεί να πειράξουμε το σύστημά μας και να αποφασίσουμε αμερόληπτα εάν, μετά την εμπειρία χρήσης του, θέλουμε να το εγκαταστήσουμε στο σύστημά μας. Σε αυτό το στάδιο, η εγκατάσταση του oVirt είναι απλοποιημένη για τους νέους χρήστες, ώστε να μην χρειαστεί να δώσουμε παραπάνω χρόνο από όσο θα θέλαμε απλά και μόνο στην εγκατάστασή του, αλλά στην εμπειρία χρήσης του.

Το LiveCD του oVirt βρίσκεται στη σελίδα https://www.oVirt.org/download/oVirt-live/ και διαθέτονται οι εκδόσεις από την 0.8 έως και την 4.1.0, beta αλλά και σταθερές εκδόσεις.

Εμείς, κάναμε download την έκδοση 4.1.0 σε μορφή `iso` αρχείου η οποία βρίσκεται στο link http://plain.resources.oVirt.org/pub/oVirt-4.1/iso/oVirt-live/4.1.0/oVirt-live-4.1.iso και είναι βασιμένη στα Enterprise Linux 7.3 (EL7.3). Έπειτα, χρησιμοποιώντας ένα πρόγραμμα CD/DVD Burner (για παράδειγμα το K3b των Linux), εγγράφουμε το `iso` αρχείο σε ένα DVD. Δεν χρειάζεται να κάνουμε κάποια επιπλέον ενέργεια ώστε να κάνουμε το DVD να είναι bootable, καθώς είναι έρχεται έτοιμο με αυτή τη δυνατότητα.

Έχουμε φροντίσει ώστε να υπάρχει ένας υπολογιστής ο οποίος θα πληροί όλες τις προϋποθέσεις και απαιτήσεις που έχει το oVirt από ένα host μηχάνημα. Επιλέξαμε έναν DELL desktop υπολογιστή με 1TB σκηρό δίσκο, 16GB μνήμη RAM και τετραπύρηνο επεξεργαστή Intel® Xeon® E5420 στα 2,50GHz και δοκιμάσαμε σε αυτόν το Live CD του oVirt.

Αρχικά, βάζουμε το Live CD του oVirt και κάνουμε boot από αυτό στον υπολογιστή μας. Όταν ανοίγουμε τον υπολογιστή για πρώτη φορά, πριν να γίνει το boot από το Live CD, μπαίνουμε στο BIOS και παρατηρούμε αν είναι σωστές οι ρυθμίσεις της ημερομηνίας και της ώρας. Εάν δεν είναι, του δίνουμε τις σωστές τιμές, αλλιώς βγαίνουμε από το μενού του BIOS και συνεχίζουμε με το boot του υπολογιστή μας. Άλλος τροπος για να δώσουμε τις ρυθμίσεις αυτές μπορούμε να τις δώσουμε και από την κονσόλα του συστήματος, εφόσον έχουμε ήδη μπει στο λειτουργικό περιβάλλον του oVirt. Οι σωστές τιμές στην ημερομηνία και την ώρα είναι πολύ σημαντικές καθώς, αρχικά, αν είναι λάθος ή πολύ παλιές δεν θα λειτουργήσει σωστά το εργαλείο yum. Επίσης, είναι πολύ σημαντικό, ειδικά σε περιπτώσεις με nodes να έχουν όλοι τις ίδιες ακριβώς τιμές για να είναι απόλυτα σωστή η επικοινωνία μεταξύ τους.

Όταν βρισκόμαστε πια στο περιβάλλον του Live CD, μας εμφανίζεται ένα παράθυρο για να κάνουμε εγκατάσταση του oVirt στο Live CD environment.Επιλέγουμε να συνεχίσουμε και να γίνει η εγκατάσταση του oVirt Engine, ώστε να μπορέσουμε να δούμε το περιβάλλον του. Η εγκατάσταση συνεχίζεται σε CLI mode και κατά τη διάρκειά της μας δίνονται διάφορες επιλογές, στις οποίες αφήνουμε τις προεπιλεγμένες τιμές, εκτός ίσως από την επιλογή που βρίσκεται στον τομέα του System Configuration “`Configure NFS share server to be used as ISO domain? (yes/no) [yes]: `”, με την οποία εάν είναι καταφατικώς επιλεγμένη να χρησιμοποιήσουμε τον κοινόχρηστο server με NFS, ως έναν τομέα που μας προσφέρει τα ISO αρχεία τα οποία θα χρειαστούμε για την εγκατάστη των εικονικών μηχανών αργότερα.

Στο τέλος της εγκατάστασης, ενημερωνόμαστε ότι μπορούμε πλέον να έχουμε πρόσβαση στο oVirt-Engine μας στη διεύθυνση https://livecd.localdomain/oVirt-Engine με τα στοιχεία χρήστη:

```
User name: admin

Password: qwer
```

Εφόσον έχουμε δώσει τις σωστές ρυθμίσεις ημερομηνίας και ώρας, μπορούμε πλέον να μπούμε στην κονσόλα του συστήματος και μέσω της υπηρεσίας Network Manager, που εκτελείται στα Linux, και των εντολών `nmctl` και `nmtui` να ρυθμίσουμε το δίκτυο της μηχανής μας, αφού δούμε πρώτα μέσω της εντολής `ifconfig` ποιο είναι το interface που χρησιμοποιούμε, ώστε να ρυθμίσουμε την σωστή κάρτα δικτύου. Με

την εντολή `hostname` ή μέσω του αρχείου που βρίσκεται στο μονοπάτι `/etc/resolv.conf` μπορούμε να δούμε ποιο είναι το hostname της μηχανής μας, καθώς χρειάζεται να το γνωρίζουμε και να μπορούμε να το παραθέσουμε αργότερα όταν θα κάνουμε εγκατάσταση το oVirt.

Με την εντολή `netstat -nPR` για να δούμε τον πίνακα δρομολόγησης του τοπικού μας δικτύου και εάν υπάρχει κάποιο πρόβλημα με τη σύνδεσή μας στο διαδίκτυο. Εάν μπορούμε να κάνουμε `ping` στον εαυτό μας και στη διεύθυνση gateway αλλά δεν μπορούμε να συνδεθούμε στο internet, μπορούμε να εκτελέσουμε την εντολή, για παράδειγμα,

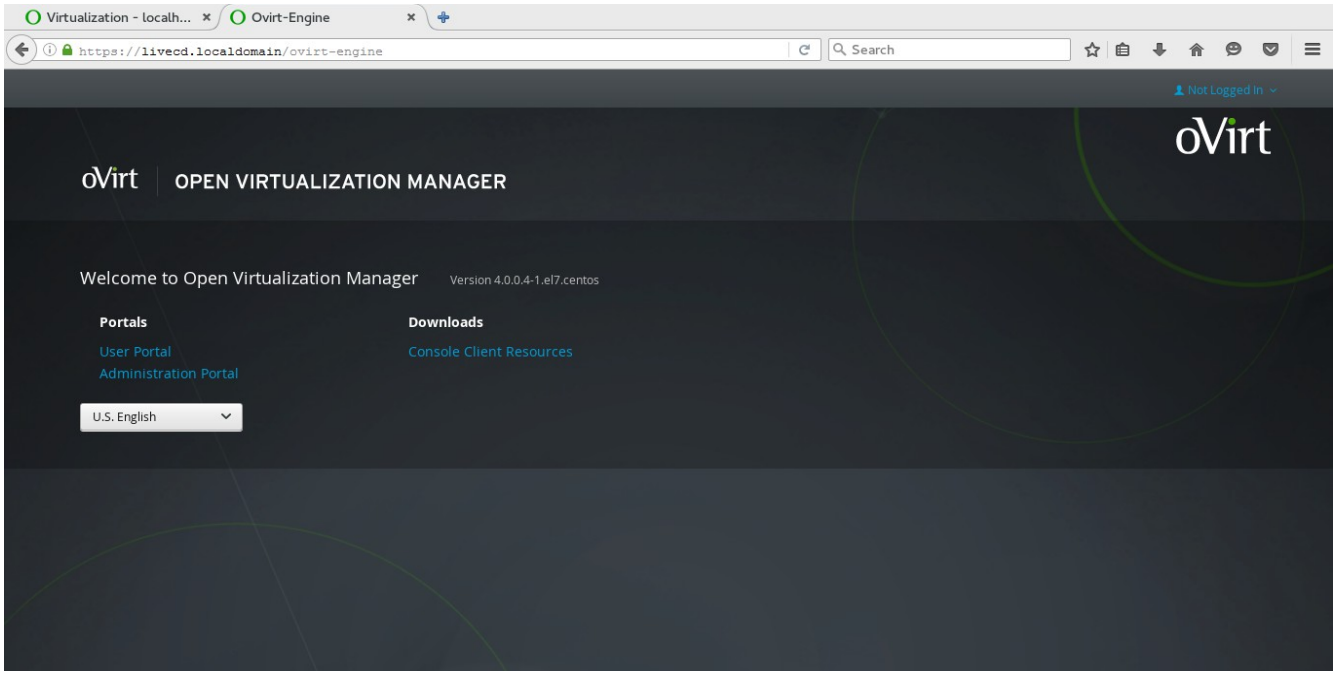
```
# route del -net 169.254.0.0 gw 0.0.0.0 oVirtmgmt
```

ή την

```
# route del -net 169.254.0.0/24
```

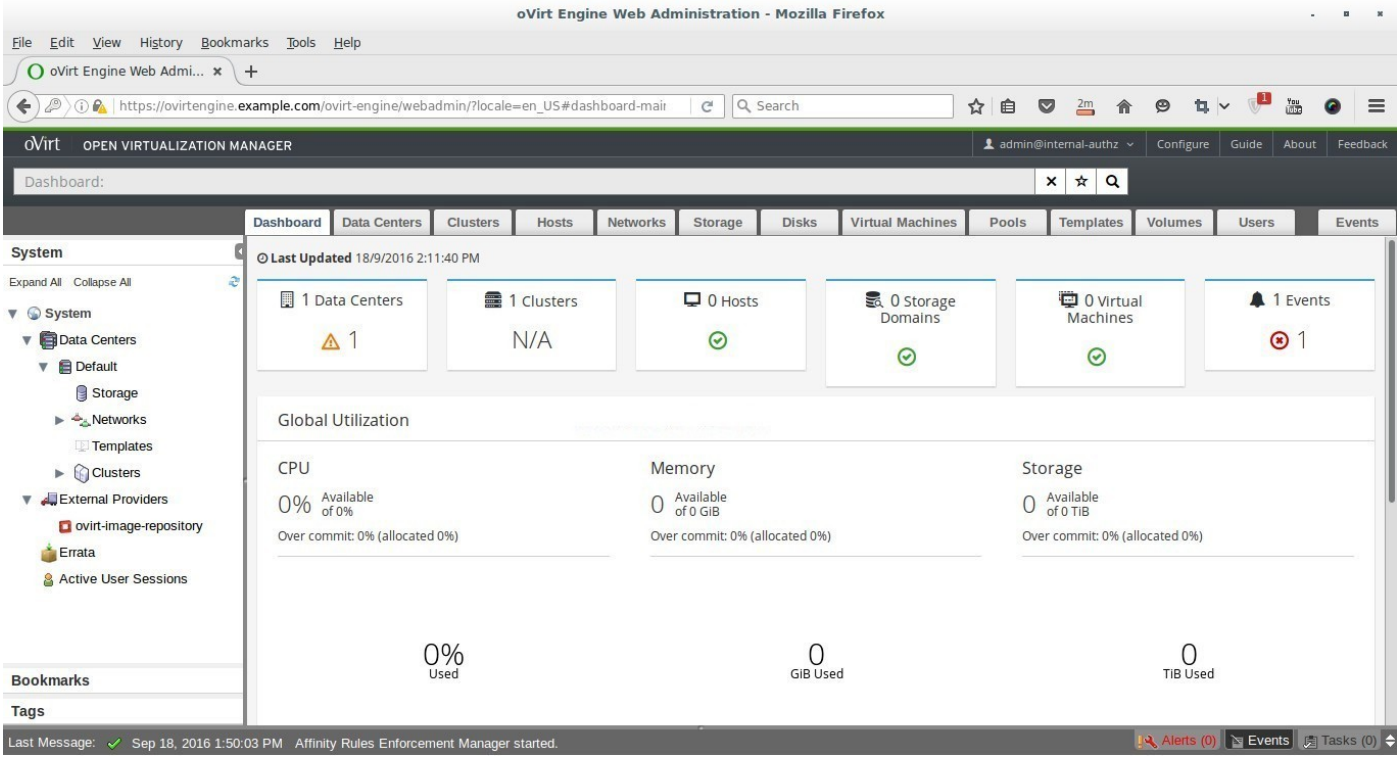
και να αφαιρέσουμε κάποια από τις εγγραφές 169 250.0.0/16 από τον πίνακα δρομολόγησης του συστήματος.

Όπως προείπαμε, με το τέλος της εγκατάστασης του oVirt μπορούμε να επισκεφτούμε πλέον την ιστοσελίδα https://livecd.localdomain/oVirt-Engine και να διαχειριστούμε το Virtualization περιβάλλον μας, είτε μέσω του user portal ή μέσω του administrator portal, τα οποία παρέχουν διαφορετικές δυνατότητες, με το administration portal να είναι αυτό που προσφέρει τις περισσότερες, φυσικά.



Εικόνα 5: Αρχική Σελίδα oVirt Engine όπου επιλέγουμε σε ποιο portal θέλουμε να μπούμε.

Από τη σελίδα που εμφανίζεται μπαίνουμε στο administrator portal, με δικαιώματα διαχειριστή της μηχανής του oVirt, όπου και μας ζητείται να δώσουμε τα στοιχεία μας (username/password), ώστε να μπορέσουμε να διαχειριστούμε το νέο μας σύστημα εικονικών μηχανών και υπηρεσιών.



Εικόνα 6: Αρχική Σελίδα oVirt Engine

Με χρήση μόνο του Live CD και χωρίς να γίνει εγκατάστασή του στον server μας, καταφέραμε μόνο να δημιουργήσουμε τις εικονικές μηχανές χωρίς ωστόσο να δημιουργηθούν πραγματικά, καθώς η διαδικασία δεν μπορούσε να προχωρήσει και ο αποθηκευτικός χώρος που είχαμε ήταν ουσιαστικά εικονικός και αυτός.

Το Live CD είναι μία μη επίσημη έκδοση είτε σε Fedora ή σε CentOS, χρησιμοποιεί το “All In One” plugin και ξεκίνησε τον Οκτώβριο του 2012. Ωστόσο μετά την 1η Φεβρουαρίου 2017 καθώς το “All In One” plugin είχε ήδη σταματήσει να αναπτύσσεται και είχε βγει εκτός παραγωγής (έκδοση 3.6 του oVirt), οπότε λίγο καιρό αργότερα και σταμάτησε η διάθεση νέων Live CD (έκδοση 4.1.0 του oVirt).

Σε κάθε έκδοση Live CD υπήρχαν αρκετά προβλήματα και bugs τα οποία διορθώνονταν με κάθε νέα έκδοση, αλλά η συνέχιση της χρήσης των Live CD δεν κρίθηκε απαραίτητη και ωφέλιμη οπότε και παρατήθηκε.

Μετά από μερικές μέρες χρήσης του Live CD του oVirt καταλήξαμε στα εξής συμπεράσματα:

1. Το Live CD αποτελεί μόνο μία έκδοση demo και δεν μας δίνει την πλήρη εμπειρία του oVirt.

2. Αντιμετωπίσαμε αρκετά προβλήματα μέχρι να καταφέρουμε να δημιουργηθεί μία σταθερή σύνδεση μεταξύ του host μας και του δικτύου λόγω των εικονικών interfaces και του λάθος routing table που δημιουργούσε από μόνο του το σύστημα, αυτόματα, καθώς «βλέπει» τα εικονικά interfaces και όχι τα πραγματικά interfaces της μηχανής στην οποία τρέχει το oVirt-Engine (host).

3. Το Live CD είναι πλέον παρωχημένο, καθώς υπήρχαν αρκετά bugs και το plugin που χρησιμοποιόταν ως wrapper όλων των επιλογών που μας δίνονταν σταμάτησε και αυτό να υποστηρίζεται από τον Μάιο του 2016 (έκδοση 3.6.6 του oVirt).

Το Live CD δεν ήταν σταθερό και είχε αρκετά προβλήματα η χρήση του, οπότε και για όλους τους παραπάνω λόγους κρίθηκε ανώφελη η περαιτέρω χρήση του.

2.4. Βασικό Single Hosted Περιβάλλον

Μια self-hosted Engine είναι ένα εικονικό περιβάλλον στο οποίο το oVirt Engine τρέχει ως μια εικονική μηχανή στους κεντρικούς υπολογιστές τους οποίους διαχειρίζεται το Engine. Η εικονική μηχανή δημιουργείται ως μέρος των ρυθμίσεων του κεντρικού υπολογιστή (host) και το Engine εγκαθίσταται και διαμορφώνεται παράλληλα με τη διαδικασία διαμόρφωσης host. Το κύριο πλεονέκτημα της self-hosted Engine είναι ότι απαιτεί λιγότερο υλικό για την ανάπτυξη ενός instance του oVirt καθώς το Engine λειτουργεί ως εικονική μηχανή και όχι με φυσικό υλικό. Επιπλέον, το Engine είναι ρυθμισμένο ώστε να έχει υψηλή διαθεσιμότητα. Εάν ο κεντρικός υπολογιστής που εκτελεί την εικονική μηχανή Engine μεταβαίνει σε λειτουργία συντήρησης ή αποτυγχάνει απροσδόκητα, η εικονική μηχανή θα μεταφερθεί αυτόματα σε έναν άλλον κεντρικό υπολογιστή ο οποίος υπάρχει στο περιβάλλον μας. Για την υποστήριξη της δυνατότητας υψηλής διαθεσιμότητας απαιτούνται τουλάχιστον δύο self-hosted κεντρικοί υπολογιστές.

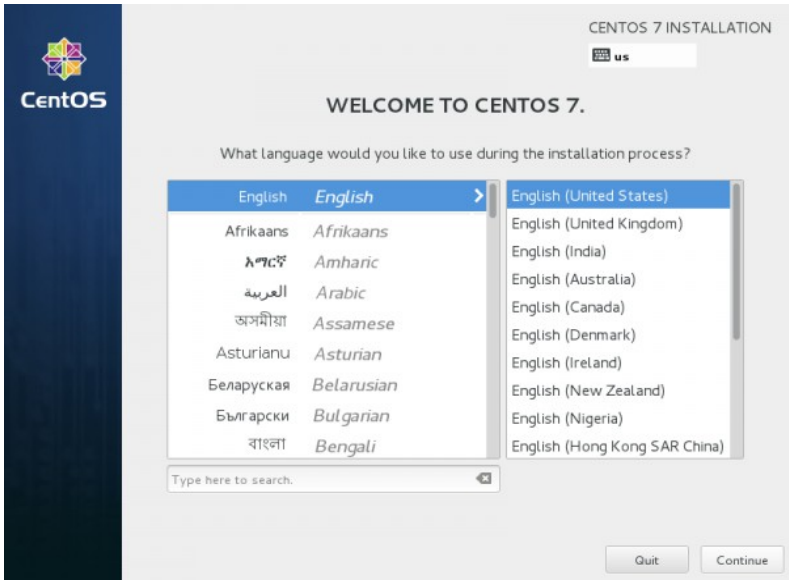
Η self-hosted Engine χρειάζεται τουλάχιστον δύο υπολογιστές ώστε εάν ο host στον οποίο εκτελείται για κάποιο λόγο βγει εκτός λειτουργίας να μην σταματήσει να εκτελείται και το self-hosted Engine μας. Επειδή η self-hosted Engine δεν μπορεί να δημιουργηθεί χειροκίνητα, αλλά είναι μία ρύθμιση ουσιαστικά που εγκαθίσταται μαζί με το υπόλοιπο λογισμικό του oVirt, αποφασισάμε ότι ο καλύτερος τρόπος για να δοκιμάσουμε το προϊόν του oVirt, θα ήταν μέσω της εγκατάστασής του σε έναν και μόνο host ο οποίος θα ανταποκρίνεται πλήρως στις προϋποθέσεις που θέτουν οι παραγωγοί του oVirt.

Καθότι δε διαθέταμε δύο ίδια μηχανήματα, αλλά θέλαμε να ελέγξουμε το oVirt πριν να προχωρήσουμε στη δημιουργία και εγκατάσταση των oVirt Nodes, αποφασίσαμε να δημιουργήσουμε ένα βασικό single hosted περιβάλλον, το οποίο θα μοιάζει αρκετά με την single-hosted Engine, αφού μιλάμε για εγκατάσταση του oVirt Engine στον ίδιο host με αυτόν που υπάρχει το oVirt.

2.4.1. Εγκατάσταση των CentOS 7

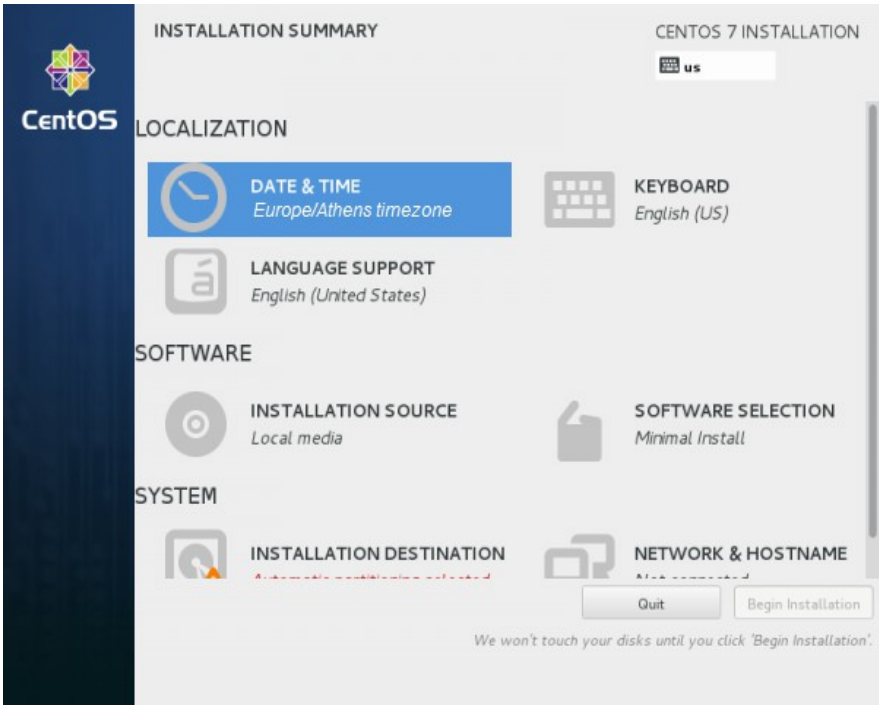
Προχωράμε κανονικά στην εγκατάσταση των CentOS 7 στον υπολογιστή που θα έχουμε ως host. Ο υπολογιστής αυτός έχει 16GB μνήμη RAM, 1TB αποθηκευτικό χώρο και τετραπύρηνο επεξεργαστή Intel® Xeon® E5420 στα 2,50GHz.

Επιλέγουμε την γλώσσα που θέλουμε να χρησιμοποιήσουμε κατά τη διάρκεια της εγκατάστασης των CentOS.



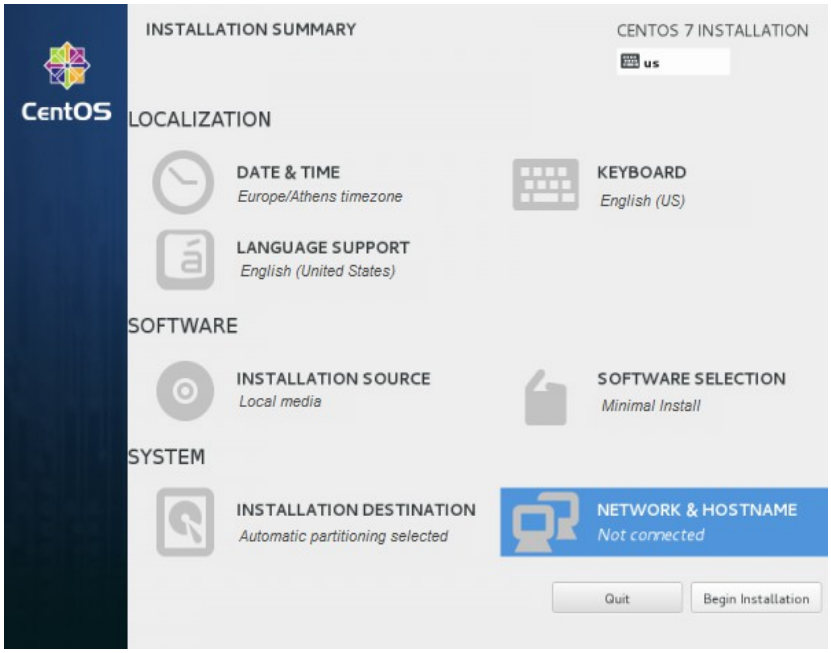
Εικόνα 7: Επιλογή γλώσσας.

Δίνουμε τις σωστές ρυθμίσεις για την ημερομηνία και την ώρα.



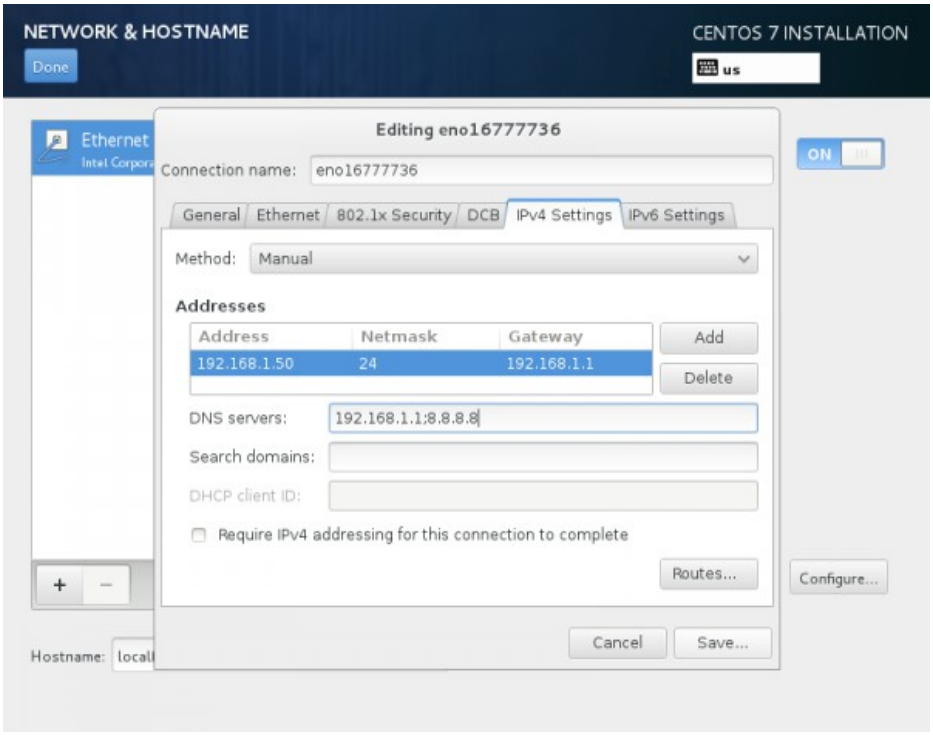
Εικόνα 8: Ρύθμιση ημερομηνίας και ώρας

Ρυθμίζουμε το δίκτυο του υπολογιστή, καθώς κατά την εγκατάσταση των CentOS χρειάζεται να είναι ικανό το σύστημα να κατεβάσει τα απαραίτητα πακέτα για τον τύπο εγκατάστασης που έχουμε επιλέξει.



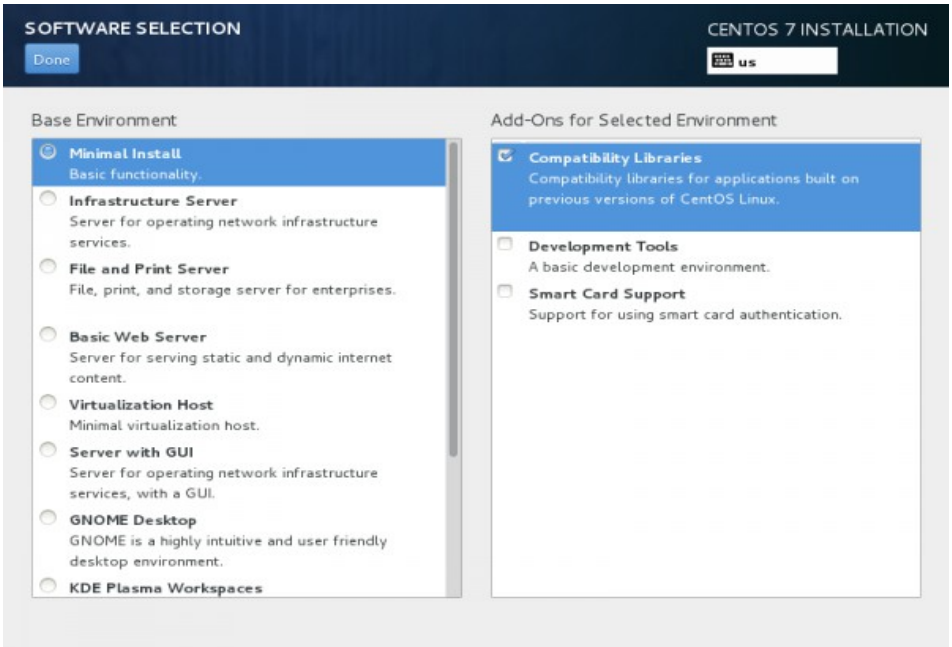
Εικόνα 9: Ρύθμιση δικτύου [1/2].

Δίνουμε την διεύθυνση IP, την μάσκα του δικτύου, τις gateway και DNS διευθύνσεις. Εάν το επιθυμούμε, μπορούμε από εδώ να αλλάξουμε το hostname του υπολογιστή μας σε κάτι άλλο από το default `localhost.localdomain`. Αν δεν θέλουμε να ορίσουμε τώρα το hostname του υπολογιστή μας, μπορούμε να το κάνουμε αργότερα μέσω της κονσόλας του συστήματος.

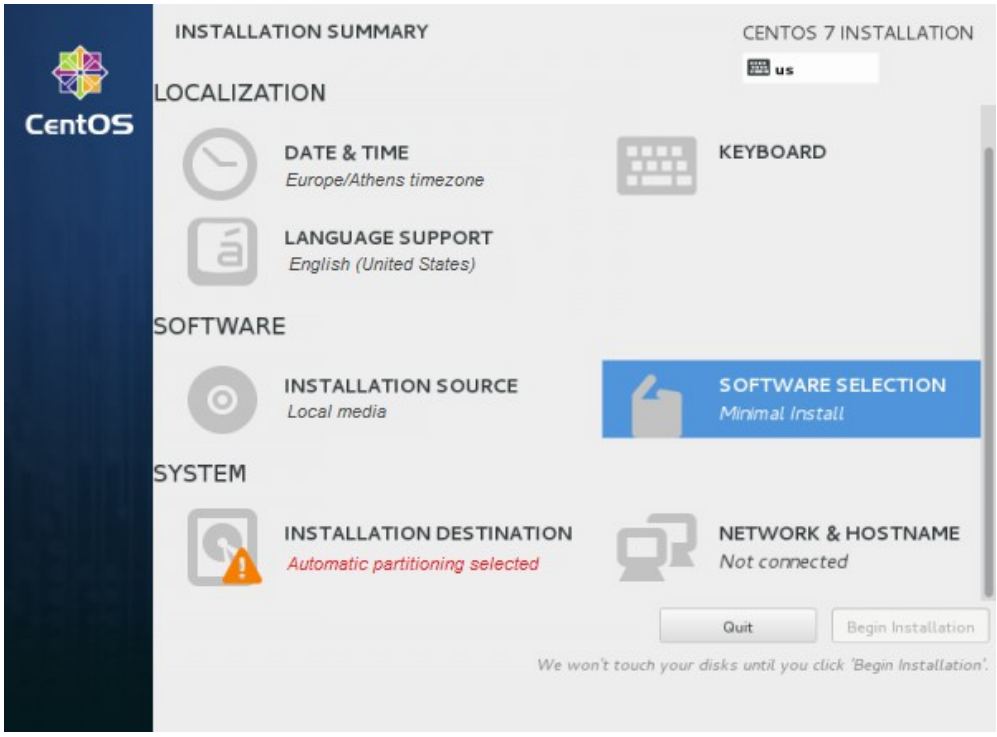


Εικόνα 10: Ρύθμιση δικτύου [2/2]

Αφού έχει ρυθμιστεί σωστά το δίκτυο, μπορούμε να προχωρήσουμε και να επιλέξουμε τον τύπο εγκατάστασης που θέλουμε να κάνουμε στα CentOS. Από τη διαθέσιμη λίστα που εμφανίζεται επιλέγουμε να κάνουμε εγκατάσταση την μινιμαλιστική έκδοση των CentOS, με την οποία εγκαθίστανται τα απολύτως απαραίτητα πακέτα του συστήματος και δεν θα υπάρξει πρόβλημα με κάποιο από τα πακέτα που θα γίνει εγκατάσταση μαζί με το oVirt.

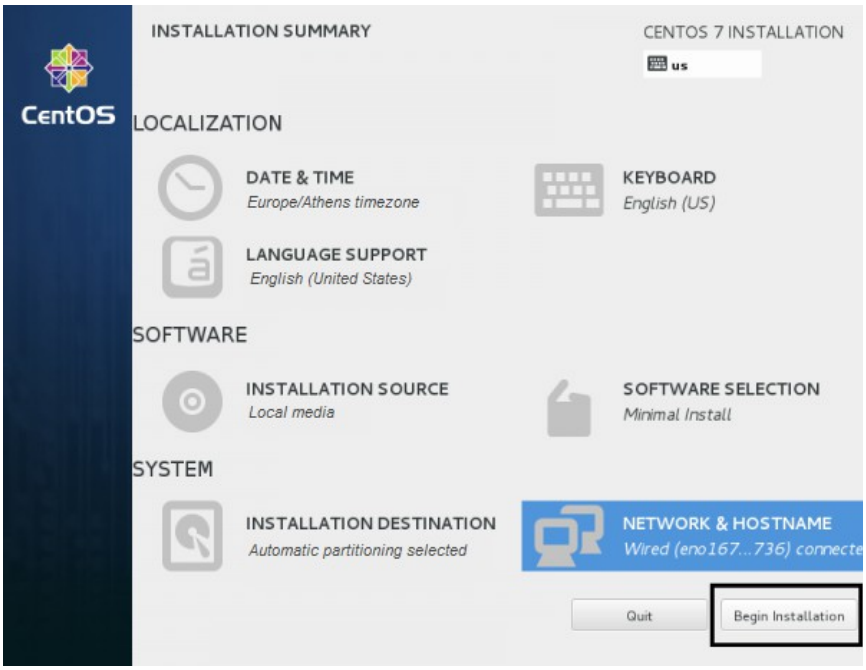


Εικόνα 11: Επιλογή τύπου εγκατάστασης [1/2].



Εικόνα 12: Επιλογή τύπου εγκατάστασης [2/2].

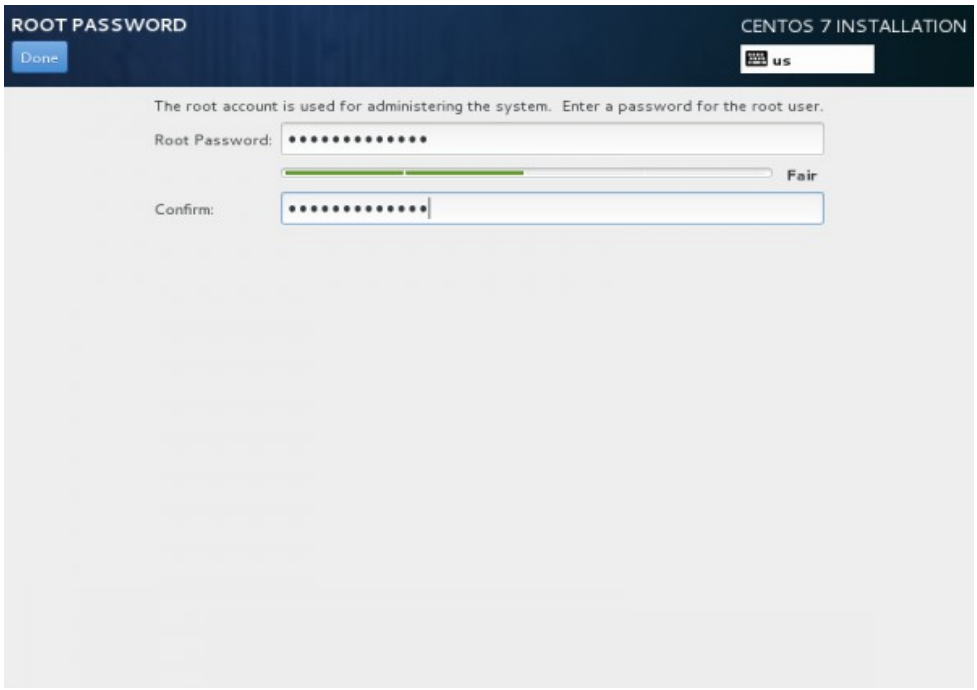
Στην επιλογή “INSTALLATION DESTINATION” επιλέγουμε την επιλογή “Automatically configure partitioning”. Εάν ο σκληρός δίσκος περιείχε οποιουδήποτε τύπου δεδομένα ή κάποιο άλλο λειτουργικό σύστημα τότε θα ερωτούμασταν από το πρόγραμμα εγκατάστασης αν θέλουμε να κάνουμε αναμόρφωση στον δίσκο. Σε μία τέτοια περίπτωση, απαντάμε καταφατικά, αναμορφώνουμε τον δίσκο ώστε να προετοιμαστεί για το νέο λειτουργικό σύστημα και επιλέγουμε και πάλι το “Automatically configure partitioning”.



Εικόνα 13: Εκκίνση εγκατάστασης των CentOS

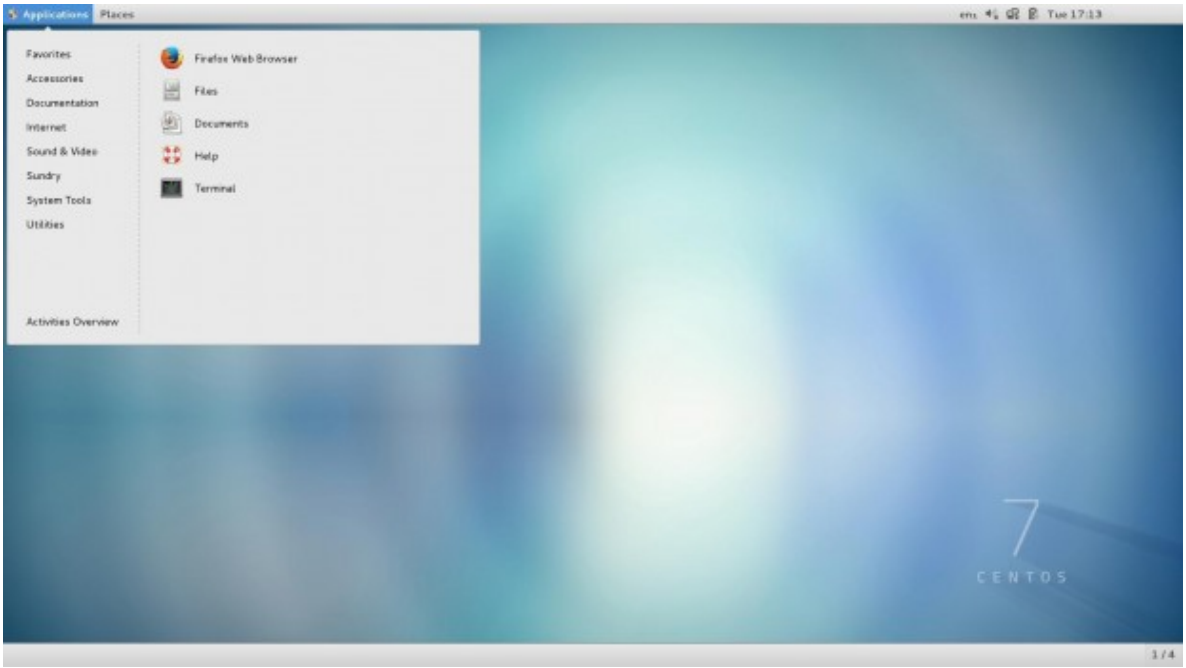
Αφού έχουμε δώσει όλες τις απαραίτητες ρυθμίσεις για να γίνει σωστά η εγκατάσταση των CentOS, προχωράμε στην εκτέλεσή της, πατώντας το *“Begin Installation”*. Κατά τη διάρκεια της εγκατάστασης μπορούμε να δώσουμε password για τον root χρήστη μας και αν το επιθυμούμε να δημιουργήσουμε έναν χρήστη ακόμα με ή δίχως δικαιώματα διαχειριστή. Το προτεινόμενο εδώ είναι να μην φτιάξουμε άλλον

χρήστη, αλλά να υπάρχει μόνο ο root χρήστης. Στην περίπτωσή μας, όπου δεν εμπλέκονται άλλα άτομα, δεν τέθηκε τέτοιο θέμα.



Εικόνα 14: Ορισμός κωδικού του χρήστη root

Με το πέρας της εγκατάστασης, το σύστημά μας είναι έτοιμο και μπορούμε να ξεκινήσουμε την εγκατάσταση του oVirt.



Εικόνα 15: Περιβάλλον CentOS

2.4.2. Προετοιμασία για την Εγκατάσταση του oVirt Engine

Αρχικά ελέγχουμε τα network interfaces του συστήματός μας μέσω της κονσόλας και της υπηρεσίας Network Manager με την εντολή:

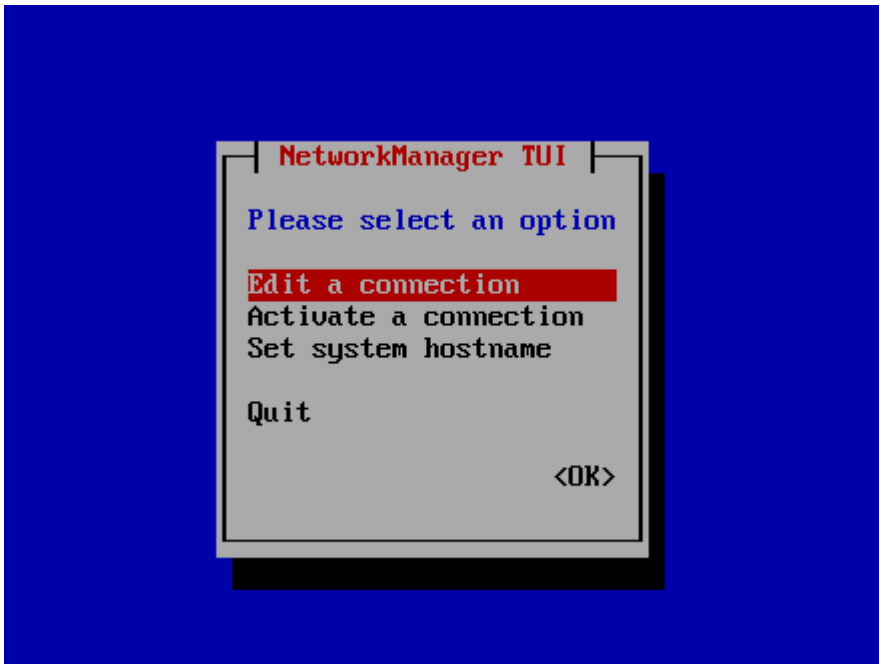
```
# nmtui
```

Αν, για οποιονδήποτε λόγο, δεν δουλεύει αμέσως θα το διορθώσουμε ως εξής:

```
# systemctl enable NetworkManager
# systemctl start NetworkManager
```

Με τις οποίες εντολές ενεργοποιούμε την υπηρεσία Network Manager (`systemctl enable NetworkManager`) και κάνουμε εκκίνησή της (`systemctl start NetworkManager`). Μέσω της υπηρεσίας Network Manager μπορούμε να κάνουμε τη διαχείριση όλων των interfaces δικτύου του συστήματός μας.

Με την εκτέλεση της εντολής `nmtui` εμφανίζεται μία διεπαφή χρήστη κειμένου με την οποία μπορούμε να διαμορφώσουμε το δίκτυο του μηχανήματός μας.



Εικόνα 16: Text-based User Interface του Network Manager.

Επιλέγουμε το *"edit a connection"* για το πρώτο interface. Αλλάζουμε την ρύθμιση από αυτόματο σε manual και δίνουμε τις παρακάτω τιμές:

```
IP: 192.168.1.27/24
GW: 192.168.1.1
DNS: 192.168.1.16
(DNS 2: 8.8.8.8 )
```

Και στη συνέχεια επανεκκινούμε το δίκτυο ώστε να είμαστε σίγουροι ότι έχει πάρει πλέον τις ρυθμίσεις που του δώσαμε (`systemctl restart network`). Μπορούμε να δούμε τις διευθύνσεις και τα interfaces του

συστήματός μας με την εντολή `ip addr show` και επαληθεύουμε ότι υπάρχει σύνδεση με το Internet με την εντολή `ping www.google.com`.

```
# systemctl restart network
# IP addr show
# ping www.google.com
```

Εάν δεν είχαμε ορίσει το `hostname` του υπολογιστή μας κατά τη διάρκεια της εγκατάστασης του λειτουργικού συστήματος, μπορούμε να το ορίσουμε τώρα, μέσω της κονσόλας με την εξής εντολή:

```
# hostnamectl set-hostname thch.rhel7domain
```

Με την εντολή:

```
# date
```

Μπορούμε να επαληθεύσουμε ότι ημερομηνία και ώρα του συστήματος είναι οι σωστές, καθώς χρειάζεται ο σέρβερ μας και τα VMs να έχουν την ίδια ώρα και ημερομηνία ώστε να μπορούν να επικοινωνούν μεταξύ τους χωρίς προβλήματα.

Κάνουμε αναβάθμιση στο σύστημά μας (CentOS Minimal), ώστε όλα τα πακέτα που έχουν κατέβει με την εγκατάσταση του συστήματος να είναι ενημερωμένα.

```
# yum update
# reboot
```

Και κάνουμε `reboot` για να αναβαθμιστεί ο πυρήνας των Linux. Συνδεόμαστε ως `root`, δίνουμε το password και μπορούμε πλέον να προχωρήσουμε στην εγκατάσταση του oVirt Engine.

2.4.3. Εγκατάσταση oVirt Engine

Για να κάνουμε εγκατάσταση το oVirt Engine, πρέπει πρώτα να προσθέσουμε το repository του.

```
# yum install http://resources.oVirt.org/pub/repo/release-oVirt.rpm
```

Κάνουμε πάλι

```
# yum update
```

Και στην περίπτωση που αναβαθμιστεί κάποιο πακέτο που αφορά τον πυρήνα των CentOS, κάνουμε πάλι reboot.

Με εκτέλεση της εντολής:

```
# yum install oVirt-Engine
```

Προχωράμε στην εγκατάσταση του πακέτου του oVirt Engine και των εξαρτήσεών του.

Με την εγκατάσταση του oVirt Engine μας δίνονται οι εξής επιλογές:

```
Total download size: 492 M

Installed size: 1.1G

Is this ok [y/d/N] : y
```

Με το πέρας αυτής της εντολής μπορούμε να προχωρήσουμε και να εκτελέσουμε την εντολή με την οποία ρυθμίζουμε όλες τις παραμέτρους του oVirt Engine.

```
# engine-setup
```

Οι επιλογές που μας παρατέθηκαν ήταν οι εξής:

```
                                    =CONFIGURE ENGINE=

Configure Engine on this host (Yes, No) [Yes]:

Configure Image I/O Proxy on this host? (Yes, No) [Yes]:

Configure WebSocket Proxy on this machine? (Yes, No) [Yes]:

Please note: Data Warehouse is required for the Engine. If you choose to not configure it on
this host, you have to configure it on a remote host, and then configure the Engine on this
host so that it can access the database of the remote Data Warehouse host.

Configure Data Warehouse on this host (Yes, No) [Yes]:

Configure VM Console Proxy on this host (Yes, No) [Yes]:

Host fully qualified DNS name of this server [*autodetected host name*]:

Setup can automatically configure the firewall on this system.

 Note: automatic configuration of the firewall may overwrite current settings.

Do you want Setup to configure the firewall? (Yes, No) [Yes]:


                                  =DATABASE CONFIGURATION=

Where is the DWH database located? (Local, Remote) [Local]:
```

```
Setup can configure the local postgresql server automatically for the DWH to run. This may
conflict with existing applications.

Would you like Setup to automatically configure postgresql and create DWH database, or prefer
to perform that manually? (Automatic, Manual) [Automatic]:

Where is the Engine database located? (Local, Remote) [Local]:

Setup can configure the local postgresql server automatically for the Engine to run. This may
conflict with existing applications.

Would you like Setup to automatically configure postgresql and create Engine database, or
prefer to perform that manually? (Automatic, Manual) [Automatic]:


                                =OVIRT ENGINE CONFIGURATION=

Engine admin password:

Confirm Engine admin password:

Application mode (Both, Virt, Gluster) [Both]


                                   =STORAGE CONFIGURATION=

Default SAN wipe after delete (Yes, No) [No]:


                                     =PKI CONFIGURATION=

Organization name for certificate [*autodetected domain-based name*]:

Setup can configure the default page of the web server to present the application home page.
This may conflict with existing applications.


                                   =APACHE CONFIGURATION=

Do you wish to set the application as the default web page of the server? (Yes, No) [Yes]:

Setup can configure apache to use SSL using a certificate issued from the internal CA.

Do you wish Setup to configure that, or prefer to perform that manually? (Automatic, Manual)
[Automatic]:


                                   =SYSTEM CONFIGURATION=

Configure an NFS share on this server to be used as an ISO Domain? (Yes, No) [Yes]:


                                =MISCELANEOUS CONFIGURATION=

Please choose Data Warehouse sampling scale:

(1) Basic

(2) Full

(1, 2)[1]:
```

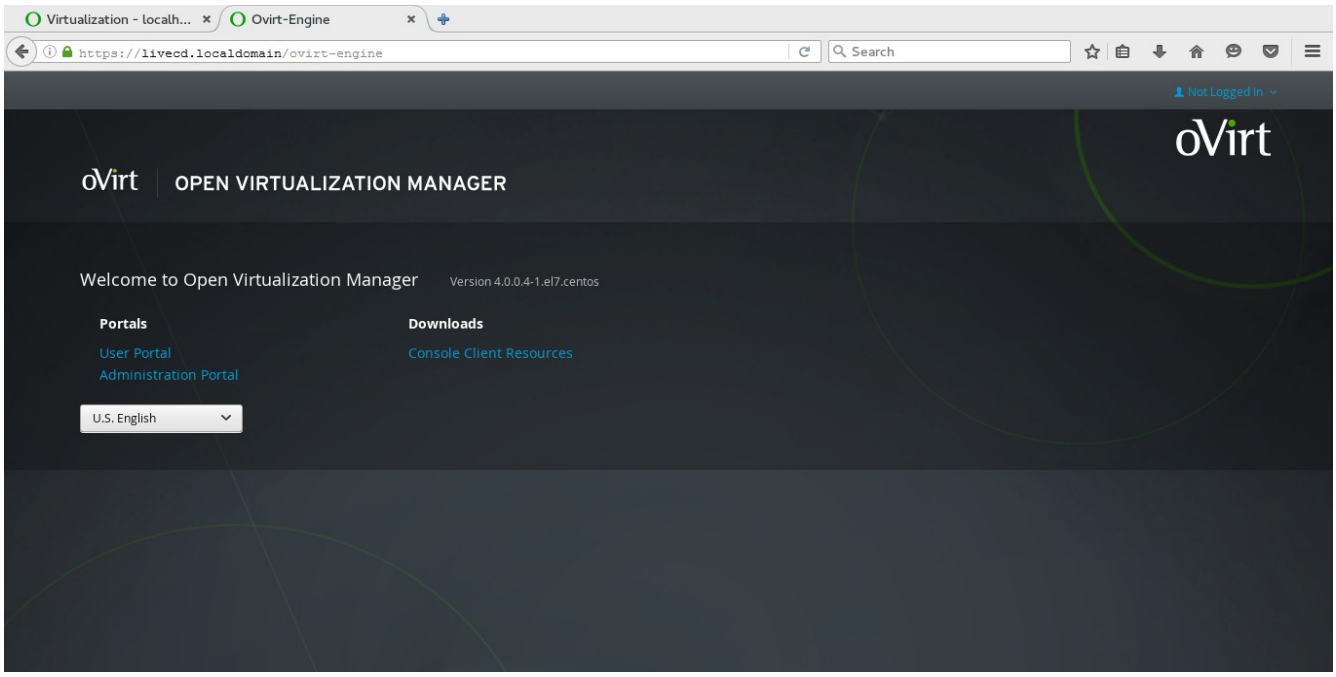
```
Please confirm installation settings (OK, Cancel) [OK]:
```

Όπου όλες οι επιλογές που βρίσκονται ανάμεσα στις αγκύλες είναι οι προεπιλογές της εγκατάστασης.

Δοκιμάσαμε μερικούς συνδυασμούς στις επιλογές αυτές, αλλά καταλήξαμε ότι οι προεπιλεγμένες τιμές ήταν και οι καταλληλότερες.

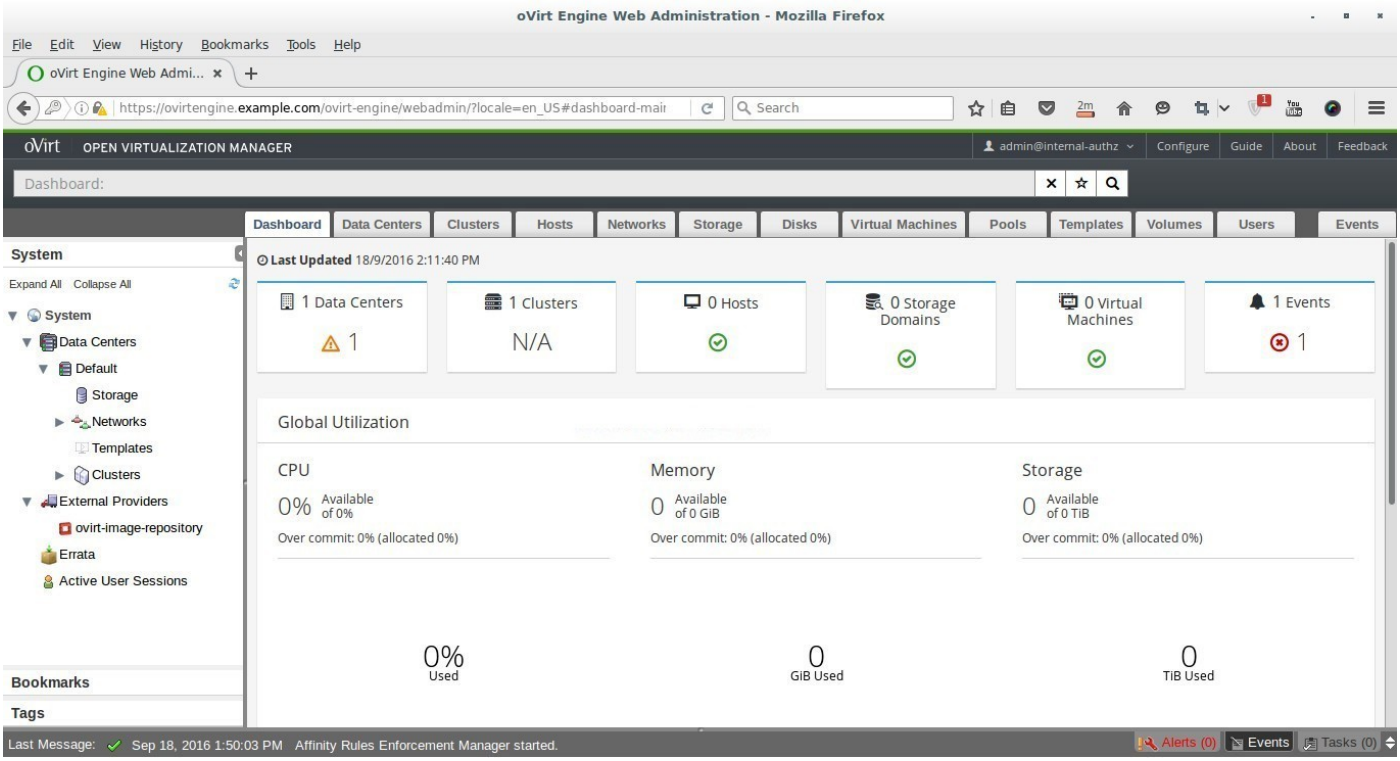
Μετά το τέλος της εγκατάστασης του oVirt Engine, μπορούμε πλεον να επισκεφτούμε μέσω ενός προγράμματος περιήγησης (πχ Firefox Mozilla), το URL https://thch.rhel7domain/oVirt-Engine και επιλέγοντας το *Adminisrtator Portal* να συνδεθούμε στο Engine, να δημιουργήσουμε VMs, να αρχίσουμε να παρέχουμε υπηρεσίες όπως Firewall, DNS, Software-Defined Data Center (SDDC) κ.λπ.

Το *User Portal* παρέχει περιορισμένες δυνατότητες και μπορούμε να κάνουμε περιορισμένα πράγματα στο oVirt. Στο *Adminisrtator Portal* παρέχεται ολόκληρο το φάσμα των δυνατοτήτων του oVirt, και με αυτό συνδεόμαστε.



Εικόνα 17: Αρχική οθόνη web interface του oVirt

Είμαστε πλέον στο web περιβάλλον του oVirt, και μπορούμε να διαχειριστούμε το σύνολο των μηχανών μας, να φτιάξουμε νέες Virtual Machines, να ορίσουμε Data Center, Clusters και πολλά άλλα που θα δούμε εν συνεχεία.



Εικόνα 18: Αρχική οθόνη περιβάλλοντος oVirt

Καθότι βρισκόμαστε σε εξωτερικό υπολογιστή ο οποίος έχει λειτουργικό σύστημα Windows και δεν βρισκόμαστε στον host με τα CentOS και το oVirt Engine, μπορούμε, για να έχουμε καλύτερη επικοινωνία και να έχουμε τη δυνατότητα να διαχειριζόμαστε το σύστημα απομακρυσμένα από τον υπολογιστή Windows, να εγκαταστήσουμε την υπηρεσία Samba στον host με τα CentOS ώστε να μπορούμε να διαμοιράζομαστε τα αρχεία σε συγκεκριμένες τοποθεσίες του CentOS συστήματος. Στην περίπτωση που έχουμε φάκελο με ISO με λειτουργικά συστήματα στον υπολογιστή Windows, μπορούμε να τα διαχειριζόμαστε κατά τη βούλησή μας, για παράδειγμα να αντιγράψουμε το αρχείο εγκατάστασης των Mint Linux σε συγκεκριμένο φάκελο στο CentOS σύστημά μας.

2.4.3.1. Samba

Ενεργοποιούμε τις δυνατότητες του samba για την περίπτωση που για οποιονδήποτε λόγο δεν μπορούμε να είμαστε φυσικά πάνω από το μηχάνημα που έχει τα CentOS και το oVirt Engine, να μπορούμε να προσπελάσουμε αυτόν τον φάκελο και να τον διαχειριστούμε. Επίσης, είναι μια πολύ καλή λύση για την περίπτωση που έχουμε τα iso αρχεία που χρειαζόμαστε για τις εικονικές μηχανές περασμένα σε υπολογιστή με Windows, να μπορούμε να κάνουμε εύκολα (drag and drop) την αντιγραφή τους σε διαφορετικό λειτουργικό σύστημα.

Για να εγκαταστήσουμε την υπηρεσία samba στο CentOS ακολουθούμε τα εξής βήματα:

```
# yum install -y samba
# systemctl stop iptables
# vi /etc/samba/smb.conf
```

Όπου κατεβάζουμε και εγκαθιστούμε το `samba` με χρήση του `yum` και σταματάμε την υπηρεσία των `iptables`.

Φτιάχνουμε φάκελο στο /samba directory και το ονομάζουμε shared-ISO:

```
# mkdir -p /samba/shared-ISO
```

Αφού πρώτα πάρουμε backup το αρχείο smb.conf, με την εντολή:

```
# cp -of /etc/samba/smb.conf /etc/samba/smb.conf.back
```

Αν θέλουμε μπορούμε να αδειάσουμε το αρχικό αρχείο smb.conf αντιγράφοντας το κενό αρχείο null στο αρχείο ρυθμίσεων του samba smb.conf δίνουμε την εξής εντολή:

```
# cat /Dev/null > etc/samba/smb.conf
```

Και στη συνέχεια για να το κάνουμε επεξεργαστούμε το ανοίγουμε με τον επεξεργαστή κειμένου vi:

```
# vi etc/samba/smb.conf
```

Οπότε και προσθέτουμε στα περιεχόμενα του αρχείου smb.conf τα κάτωθι:

```
[Shared-ISO]
path=/samba/shared-iso
browsable=yes
writeable=yes
guest ok=yes
read-only=no
```

Προσθέτουμε την υπηρεσία samba στην δημόσια ζώνη του firewall ώστε να μην μπλοκάρει το firewall την υπηρεσία του samba. Αυτό γίνεται δίνοντας την εξής εντολή:

```
 # firewall-cmd --permanent --zone=public --add-service=samba
```

Επαναφορτώνουμε το firewall-cmd για να φορτωθούν οι αλλαγές που κάναμε με την προηγούμενη εντολή:

```
# firewall-cmd --reload
```

Αλλάζουμε τον ιδιοκτήτη του φακέλου /samba και δίνουμε δικαιώματα, ώστε αργότερα να μπορούμε να προσπελάσουμε τα αρχεία που θα βρίσκονται σε αυτόν τον φάκελο από άλλον υπολογιστή με λειτουργικό σύστημα Windows.

```
# cd /samba
# chmod –R 0755 shared-iso/
# chown –R nobody:nobody shared-iso/
# ls -l shared-iso/
```

Επιπλέον πρέπει να επιτρέψουμε στον SElinux την αλλαγή των ρυθμίσεων ασφαλείας αρχείων για το samba με την εξής εντολή:

```
 # chcon -t samba_share_t shared-iso/
```

Για να αποκτήσουμε πρόσβαση στα CentOS από τα Windows, πάμε στον υπολογιστή που τρέχει Windows, Start>Run>Εκτέλεση (cmd) και πληκτρολογούμε \\CentOS. Ο χρήστης των Windows μπορεί να δει τον κοινόχρηστο φάκελο και να τον προσπελάσει, να δημιουργήσει δικά του περιεχόμενα στον φάκελο και άλλα.

2.4.3.1.1. Secured Samba Server

Δημιουργούμε:

1. Group που το ονομάζουμε smbgroup.
2. Χρήστη που τον ονομάζουμε smbmar για να μπορούμε να προσπελαύνουμε τον Samba server με χρήση της κατάλληλης αυθεντικοποίησης χρηστών.
3. Και δημιουργούμε έναν κωδικό (password) για τον χρήστη samba που δημιουργήσαμε, smbmar.

```
# groupadd smbgroup
# useradd smbmar –G smbgroup
# smbpasswd –a smbmar
New smb password: gr00+
Confirm password: gr00+
Added user smbmar
```

Δημιουργούμε φάκελο /samba/secured με τις κάτωθι άδειες:

```
# mkdir -p /samba/secured
# cd /samba
# chmod -R 0777 secured/
# chcon -t samba_share_t secured/
```

Μέσω του vi επεξεργαστή κειμένου, επεξεργαζόμαστε πάλι τις ρυθμίσεις του samba στο αρχείο smb.conf:

```
# vi etc /samba/smb.conf
```

και προσθέτουμε τα εξής:

```
[secured]
path=/samba/secured
valid users =@smbgroup
guest ok = no
writable = yes
browsable = yes
```

Βγαίνουμε από το αρχείο και σώζουμε τις αλλαγές με ESC+ZZ. Εν συνεχεία, επανεκκινούμε την υπηρεσία samba.

```
# systemctl restart smb.service
# systemctl restart nmb. service
```

Μπορούμε να εξετάσουμε τις ρυθμίσεις και να δούμε ότι οι αλλαγές που κάναμε έχουν φορτωθεί στο σύστημα με την εντολή:

```
# testparm
```

Τώρα στον host με τα Windows μας εμφανίζεται παράθυρο για να συνδεθούμε ως χρήστης Samba, όπου και μας ζητούνται το όνομα χρήστη και ο κωδικός μας:

```
user: smbmar
password: gr00+
```

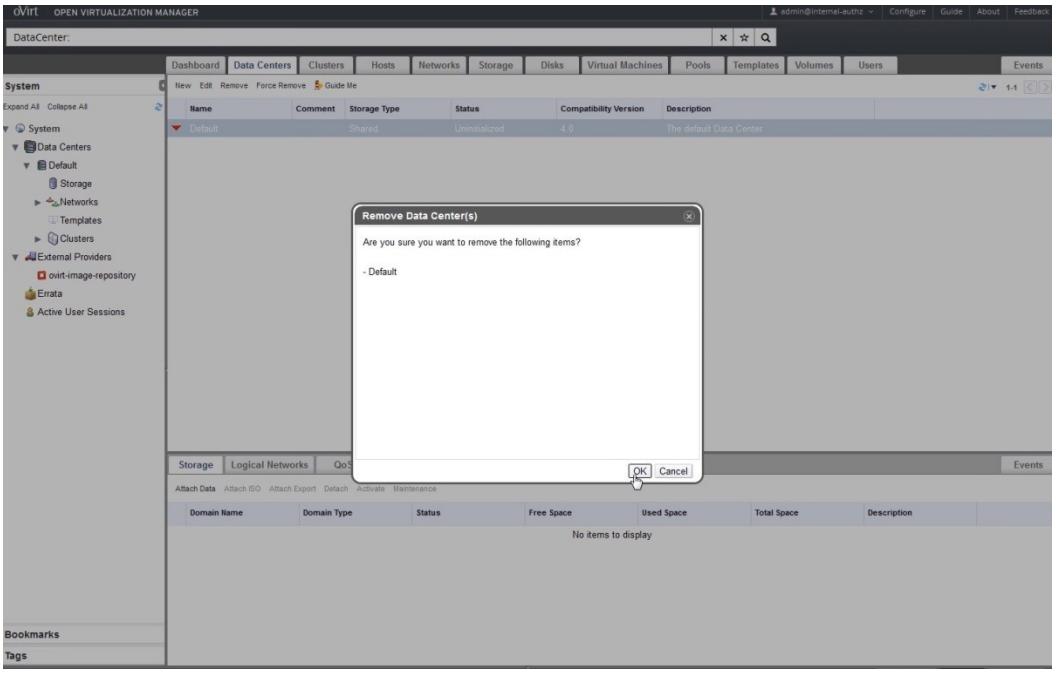
Από τον host με τα CentOS δίνουμε άδεια στον χρήστη smbmar:

```
# chown -R smbmar:smbgroup secured/
```

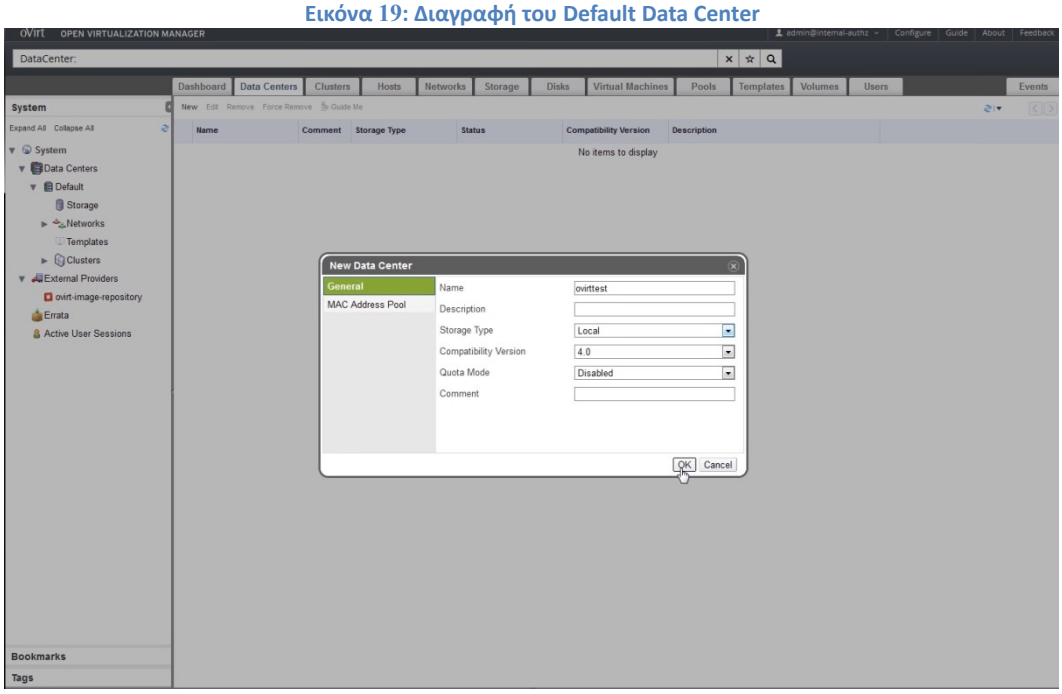
Οπότε πλέον ο χρήστης που μπορεί να συνδεθεί στο σύστημά μας είναι ελεγμένος και απαιτείται η παροχή του κωδικού του για την είσοδό του και τις ενέργειές του σε αυτό. Εφόσον πια έχει δημιουργηθεί σωστά το samba μπορούμε πια να συνεχίσουμε με τη δοκιμή του oVirt και των υπηρεσιών που προσφέρει.

2.4.3.2. Δημιουργία νέου Domain

Πρώτο βήμα μας είναι να διαγράψουμε το default Data Center που δημιουργείται μαζί με την εγκατάσταση του oVirt καθώς δεν το χρειαζόμαστε και θα δημιουργήσουμε δικό μας Data Center.



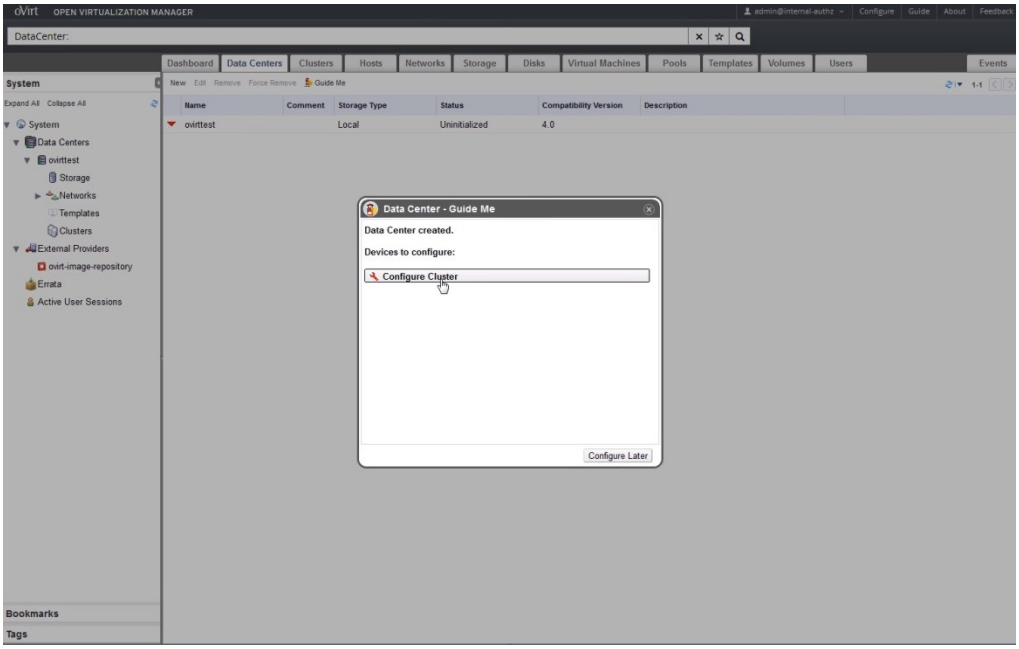
Εικόνα 19: Διαγραφή του Default Data Center



Εικόνα 20: Δημιουργία Νέου Data Center

Δημιουργούμε νέο Data Center και στα πεδία με τις ρυθμίσεις του δίνουμε όνομα για το Data Center και ορίζουμε τον τύπο του αποθηκευτικού χώρου ο οποίος μπορεί να είναι τοπικός (local) ή Gluster. Για τον Gluster χρειαζόμαστε τα oVirt Nodes. Επειδή εδώ εξετάζουμε μία διαφορετική κατεύθυνση, δεν θα το δούμε τώρα, αλλά θα επιλέξουμε την επιλογή Local. Το Gluster θα το δούμε παρακάτω όταν θα εξετάζουμε τα oVirt Nodes. Οι υπόλοιπες επιλογές παραμένουν ως έχουν.

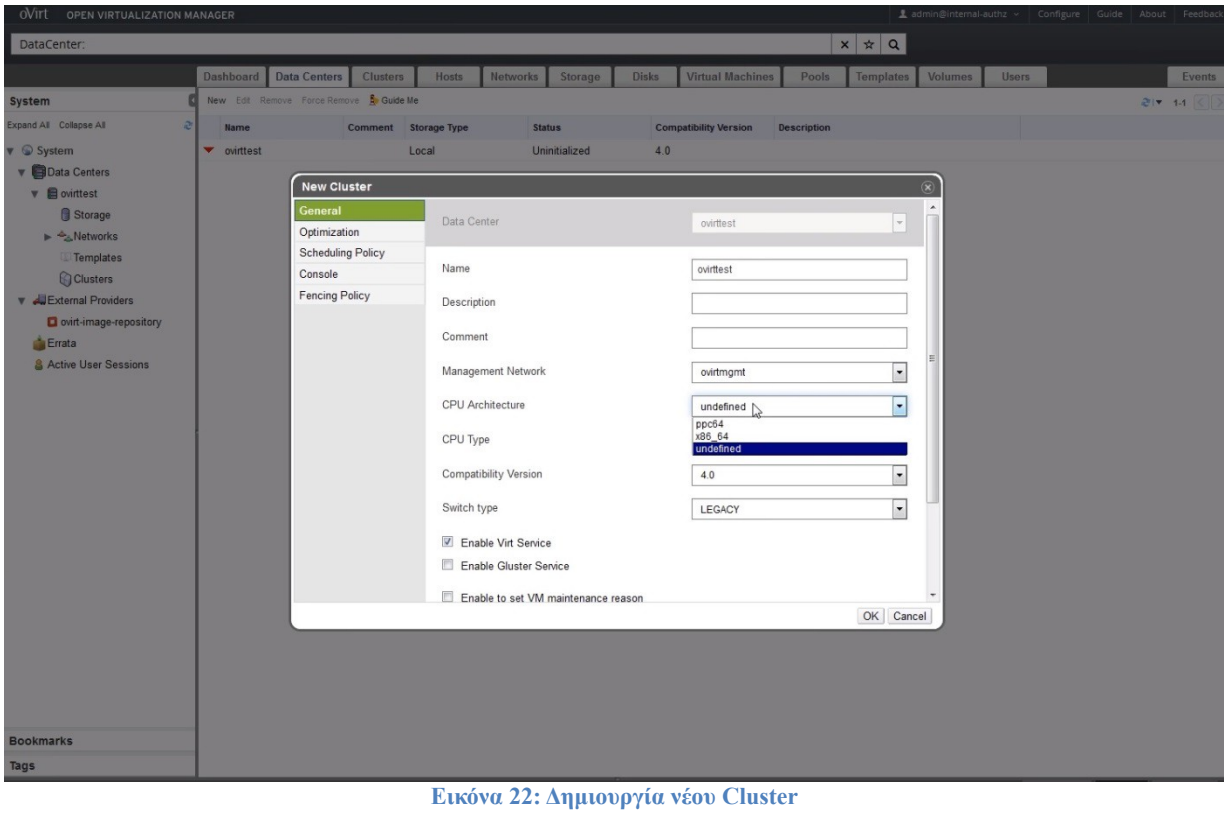
Με το που δημιουργείται το καινούριο Data Center, ερωτώμαστε αν θέλουμε να δημιουργήσουμε cluster. Επιλέγουμε το "*Configure Cluster*" και προχωράμε στη δημιουργία του.



Εικόνα 21: Ρύθμιση του Clustertου Data Center

2.4.3.3. Δημιουργία Cluster

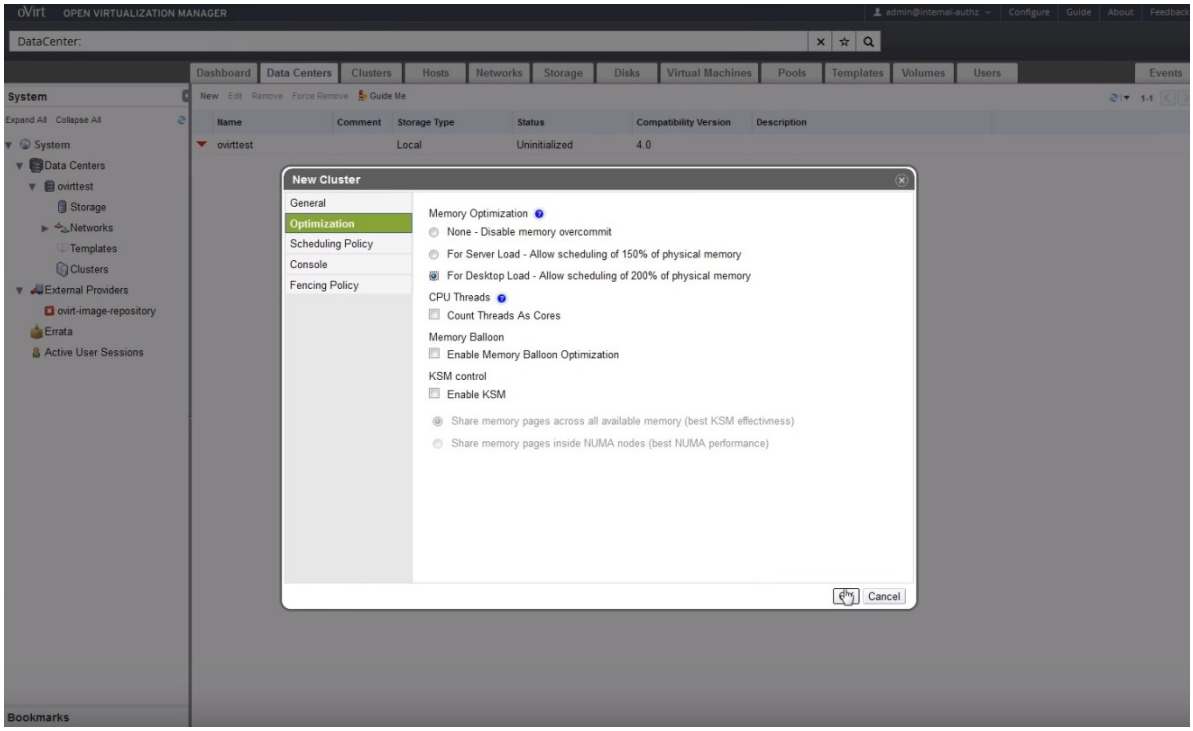
Ξεκινάμε τη δημιουργία του νέου Cluster. Ορίζουμε ένα όνομα στο πεδίο "*Name*" και την αρχιτεκτονική της CPU σε x86_64 αφού το σύστημά μας είναι 64-bit.



Εικόνα 22: Δημιουργία νέου Cluster

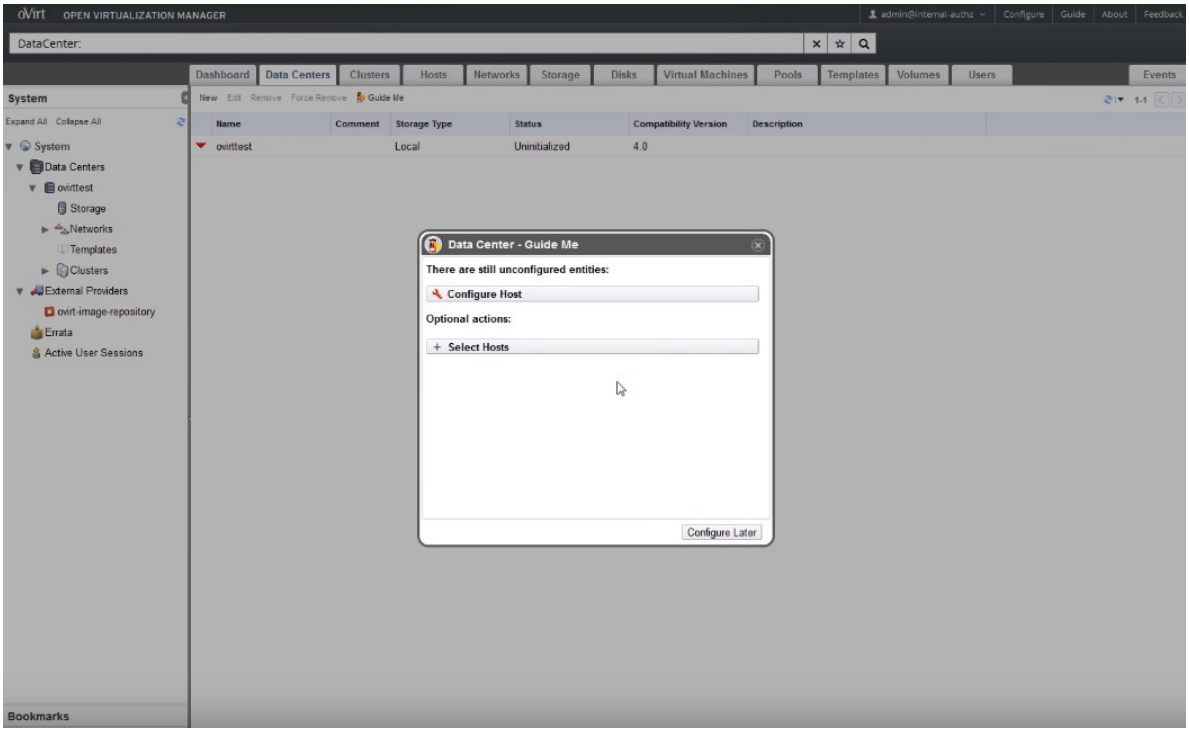
Στο πεδίο "*CPU Type*" είναι προεπιλεγμένη ο τύπος *Intel Conroe* και είναι αυτός που θέλουμε για το σύστημά μας. Σε νέα λειτουργικά συστήματα χρειάζεται εδώ να επιλεχθεί ο τύπος *Intel Nehalem.*

Στη συνέχεια, πάμε στο αριστερό μενού και επιλέγουμε την κατηγορία *Optimization* όπου μπορούμε να ορίσουμε τη βελτιστοποίηση μνήμης που θέλουμε για τον φόρτο εργασίας που θέλουμε την μνήμη. Επιλέγουμε την επιλογή *Desktop Load – 200% physical memory,* πατάμε το ok και το Cluster μας είναι έτοιμο.



Εικόνα 23: Βελτιστοποίηση για Desktop Load.

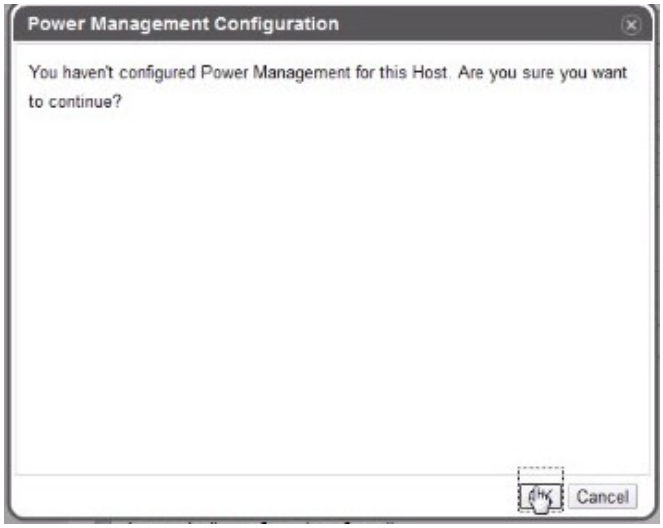
Με το τέλος της δημιουργίας του Cluster, συνεχίζουμε ορίζοντας τον host που θα ανήκει σε αυτό το Cluster.



Εικόνα 24: Επιλογή για δημιουργία νέου host ή σύνδεση με ήδη υπάρχων.

Επιλέγουμε το "*Configure Host*" για δημιουργία νέου host και συνεχίζουμε τις ρυθμίσεις μας.

Στην καινούρια έκδοση του oVirt (4.2.3.1), υπάρχει κατηγορία *Fencing Policy* όπου μπορούμε να ορίσουμε το fencing το οποίο αφορά στο τι θα γίνει με τους hosts που υπάρχουν στο ίδιο Cluster σε περίπτωση που κάποιου host η λειτουργία διακοπεί απότομα (π.χ. διακοπή ρεύματος). Στην περίπτωση που θέλουμε να ορίσουμε κάποιο Fencing Policy, τότε πρέπει να μπορούμε να ορίσουμε και τους κατάλληλους Fencing Agents, αλλιώς μπορούμε να αγνοήσουμε το αναδυόμενο παράθυρο που μας προειδοποιεί ότι δεν έχουμε ορίσει Power Management για το Cluster.



Εικόνα 25: Αναδυόμενο παράθυρο που μας προειδοποιεί για το Power management

2.4.3.4. Δημιουργία Νέου Host

Στην γενική κατηγορία ρυθμίσεων μπορούμε να ορίσουμε:

1. Το όνομα του host το οποίο είναι ένα διακριτικό όνομα, ώστε να γνωρίζει ο διαχειριστής του oVirt τον host.
2. Τη διεύθυνση του host, που θα είναι της μορφής hostname.domainname.
3. Τον κωδικό του χρήστη root, ο οποίος είναι ο κωδικός που έχει ο χρήστης root στο CentOS σύστημά μας. Καθότι δημιουργούμε μία oVirt Engine πάνω στον ίδιο host, ο κωδικός του root είναι ο ίδιος με αυτόν που δώσαμε όταν κάναμε την εγκατάσταση των CentOS.
4. Την πύλη στην οποία επιτρέπεται η SSH υπηρεσία.
5. Και εάν επιθυμούμε να δώσουμε το δημόσιο κλειδί που έχει ο χρήστης root.

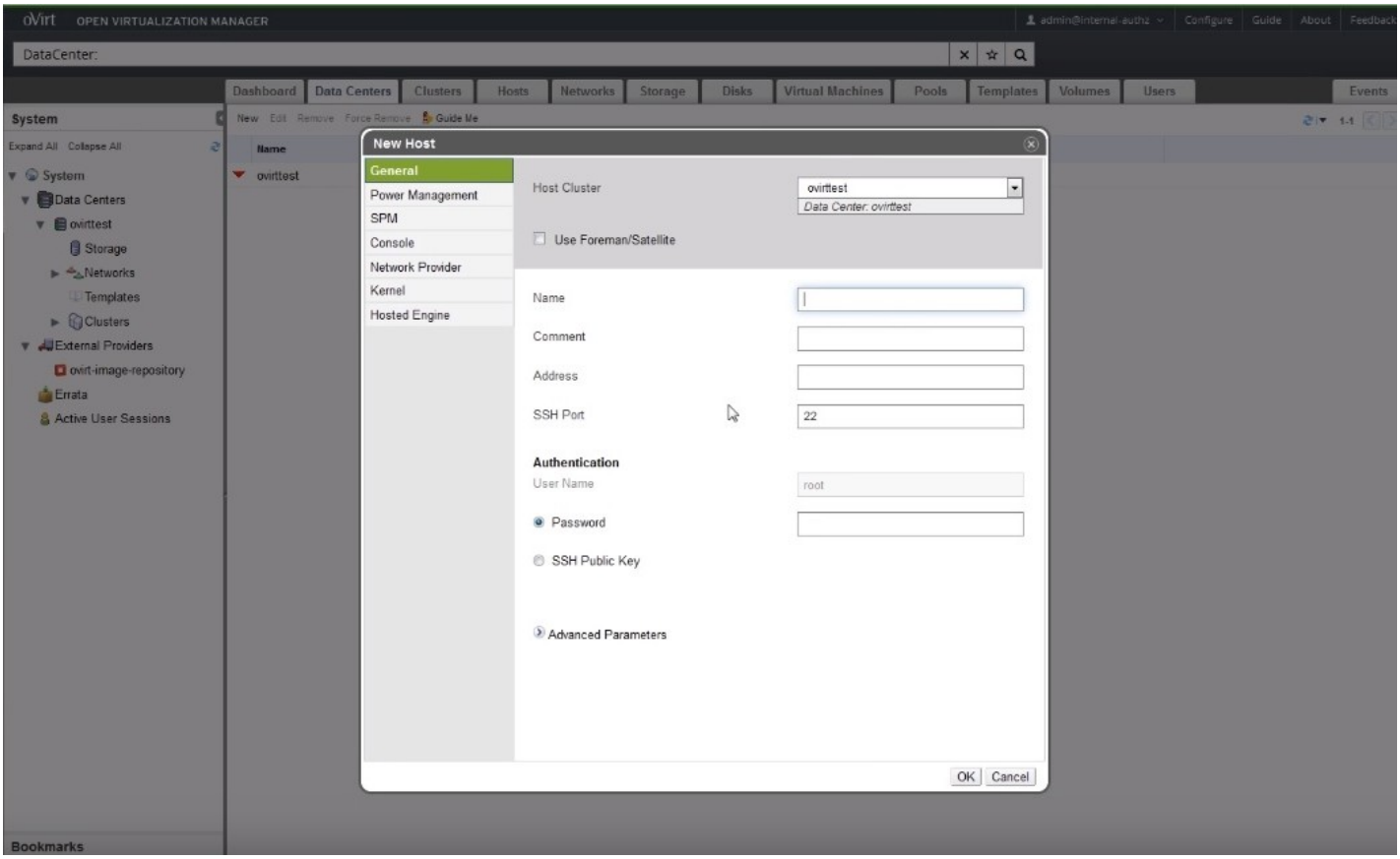
Στην περίπτωσή μας, συμπληρώσαμε τα πεδία ως εξής:

```
Name: thch
Address: thch.rhel7domain
SSH port: 22
Password: qwerty
```

Στην ίδια κατηγορία υπάρχει επιλογή για *Advanced Parameters,* την οποία και επιλέγουμε. Εδώ υπάρχει checkbox για το *Automatically Configure Firewall*. Αυτή η επιλογή πρέπει να **μην** είναι τσεκαρισμένη, καθώς στην αντίθετη περίπτωση θα αντιμετωπίσουμε προβλήματα με το firewall του host μας, αφού θα ρυθμίζεται και από τα CentOS και θα προσπαθεί και το oVirt να το ρυθμίσει όπως αυτό νομίζει.

Δεν χρειάζεται να αλλάξουμε κάποια άλλη από τις προεπιλογές, οπότε μπορούμε να πατήσουμε το ok και να ολοκληρώσουμε τη δημιουργία του host μας.

2.4.3.5. Δημιουργία Επιπλέον Data Center



Εικόνα 26: Δημιουργία νέου host.

Ακολουθούμε τα ίδια βήματα που κάναμε προηγουμένως και προχωράμε στη δημιουργία δύο νέων domains, ένα για να χρησιμοποιούμε ως δεδομένα και ένα ακόμα για τα αρχεία iso.

1. Domain για Δεδομένα:

```
Data Center: ovirttest (Local)
Domain function: data
Storage type: local on host
Host to use live cd: thch
Name: data
Path: /ovirt/data
```

2. Domain για iso:

```
Data Center: ovirttest (Local)
```

```
Domain function: iso
Name: iso
Path: /ovirt/iso
```

Τα δύο προαναφερόμενα μονοπάτια τα έχουμε ήδη δημιουργήσει με την εντολή `mkdir` και έχουμε αλλάξει τα permissions σε αυτά με την εντολή `chmod –R 777:`

```
#  mkdir /ovirt/data
#  mkdir /ovirt/iso
#  chmod –R 777 /ovirt/data
#  chmod –R 777 /ovirt/iso
```

Από ότι παρατηρήσαμε, μερικές φορές αλλάζε ο πίνακας δρομολόγησης αυτόματα και προστίθενται routes τα οποία δεν χρειαζόντουσαν αλλά ταυτόχρονα επηρεάζαν τη σύνδεσή μας με το διαδίκτυο. Συγκεκριμένα τα:

```
10.0.0.0      0.0.0.0
169.254.0.0   0.0.0.0
```

Αφαιρούμε αυτά τα routes με την εντολή:

```
# route del –net 10.0.0.0/24
# route del –net 169.254.0.0/24
```

και προσθέτουμε το αντίστοιχο route που λείπει:

```
# ip route add 0.0.0.0/0 via 192.168.10.1 dev em1
```

Χρησιμοποιούμε την εντολή:

```
# netstat -pnr
```

για να δούμε τον πίνακα δρομολόγησης ο οποίος πλέον έχει την εξής μορφή:

```
0.0.0.0       170.22.10.1   0.0.0.0
170.22.10.0   0.0.0.0       255.255.255.0
```

Εφόσον εκτελούμε την υπηρεσία samba και έχουμε συνδεθεί με τον samba χρήστη μας, εκτελούμε την εξής εντολή, ώστε να αντιγράψουμε το iso αρχείο στο CD-ROM των Windows το οποίο υπάρχει στον host που τρέχει Windows, στο directory των CentOS `/ovirt/iso`:

```
# isoinfo -d -i /dev/cdrom/grep -i -E 'block size|volume size'

# dd if =/dev/cdrom of=test.iso bs=2048 count=304937
```

Όπου, `dd`: Εντολή που μετατρέπει και αντιγράφει αρχεία,
`if`: input file, το αρχείο που θέλουμε να αντιγράψουμε,
`of`: output file, το αρχείο προορισμού, αυτό που στο τέλος θα περιέχει τα περιεχόμενα του input file,
`bs`: το block size με το οποίο θα γίνεται η αντιγραφή του ενός αρχείου στο αρχείο προορισμού,
`count`: το volume size,
Τα `bs` και `count` μεγέθη τα μάθαμε με την πρώτη εντολή `isoinfo`.

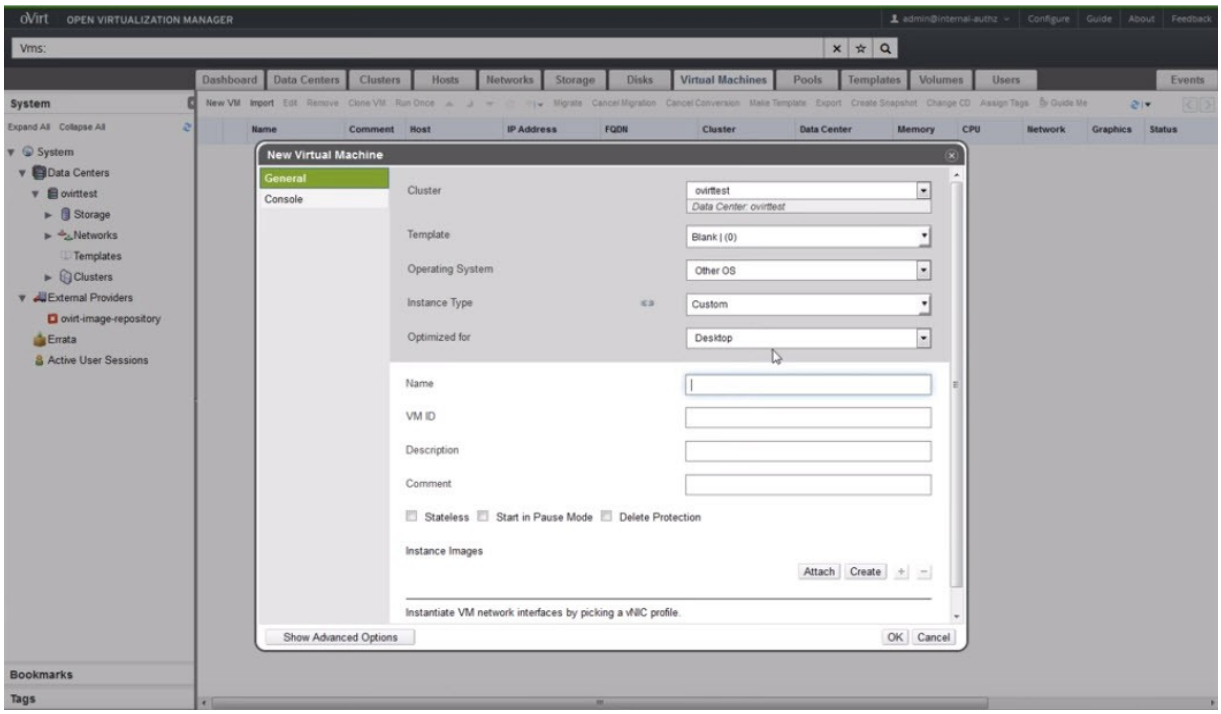
Τα iso αρχεία τα τοποθετούμε στο directory του samba (δηλώνεται και αυτό με τις εντολές `chcon -t samba_share_t` και `chown –R nobody:nobody`):

```
/ovirt/iso/0793.../images/11111111-1111-1111-1111-11111111/
```

Για να κάνουμε εγκατάσταση των Windows σε εικονική μηχανή από Linux πρέπει να κατεβάσουμε τα drivers για τα Windows από την ιστοσελίδα της Fedora http://www.linux-kvm.org/page/Windows-guest-drivers/download_drivers.

2.4.3.6. Δημιουργία Νέας Εικονικής Μηχανής

Πλέον έχουμε προετοιμάσει το έδαφος και μπορούμε να προχωρήσουμε στην δημιουργία μίας νέας εικονικής μηχανής. Πάμε στην καρτέλα



Εικόνα 27: Παράθυρο δημιουργίας νέας εικονικής μηχανής

Στα πεδία που αναγράφονται στην παράπανω εικόνα δίνουμε τις εξής τιμές:

```
Name: Linux_ Mint
VM ID: [*] eb5f8f0e-60c6-4252-8b0a-bc8ae0772974
```

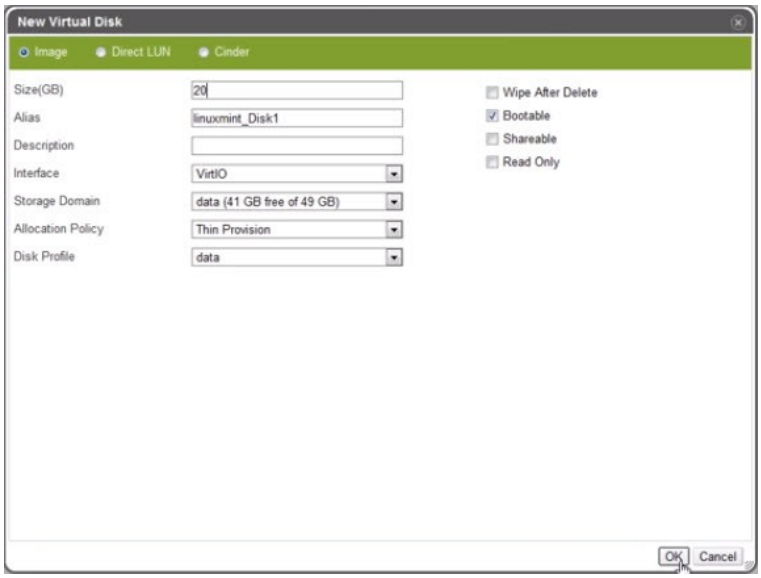
[*] Για το VM ID πρέπει να πάμε στην ιστοσελίδα https://www.guidgen.com, όπου μπορούμε να δημιουργήσουμε έναν GUID αριθμό τον οποίο και θα αντιγράψουμε και μετά θα έρθουμε να επικολλήσουμε σε αυτό το πεδίο.

Εικόνα 28: Ιστοσελίδα για δημιουργία GUIDs.

Στη συνέχεια στα *Instance images* επιλέγουμε το *Create*:

```
Instance images: Create
```

Και εμφανίζεται το παρακάτω παράθυρο:



Εικόνα 29: Το παράθυρο που εμφανίζεται όταν διαλέγουμε στο Instance να κάνουμε Create. Δημιουργία του αποθηκευτικού χώρου της VM.

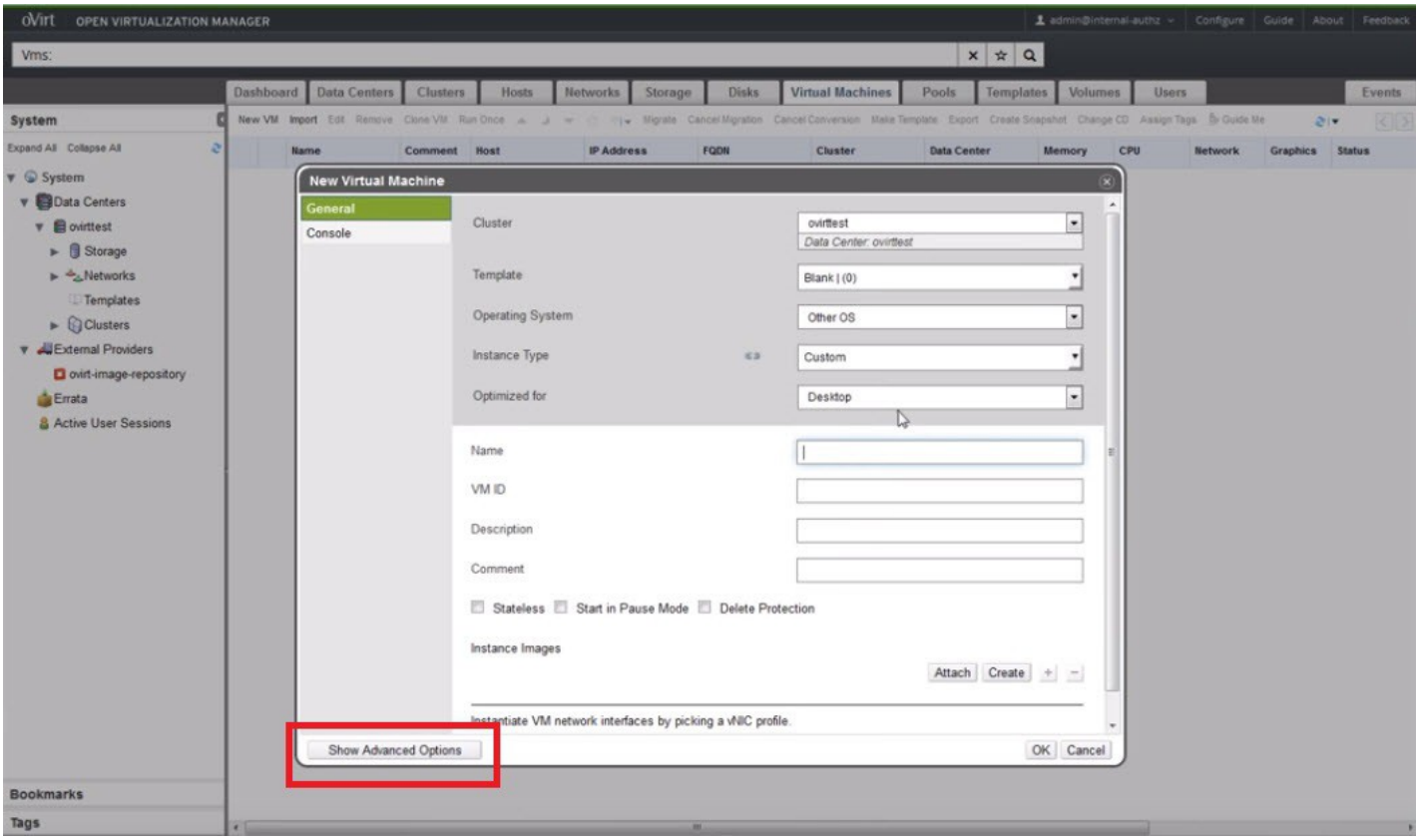
Τα πεδία που αναγράφονται παραπάνω τα συμπληρώσαμε ως εξής:

```
Size(GB): 20
Big virtual NIC profile: oVirtmgmt /oVirtmgmt
System: Memory Size: 2048
Total Virtual CPUs: 4
```

Στην κατηγορία *Console* επιλέγουμε το *USB support enabled* και πατάμε το ok, και το oVirt δημιουργεί το VM μας.

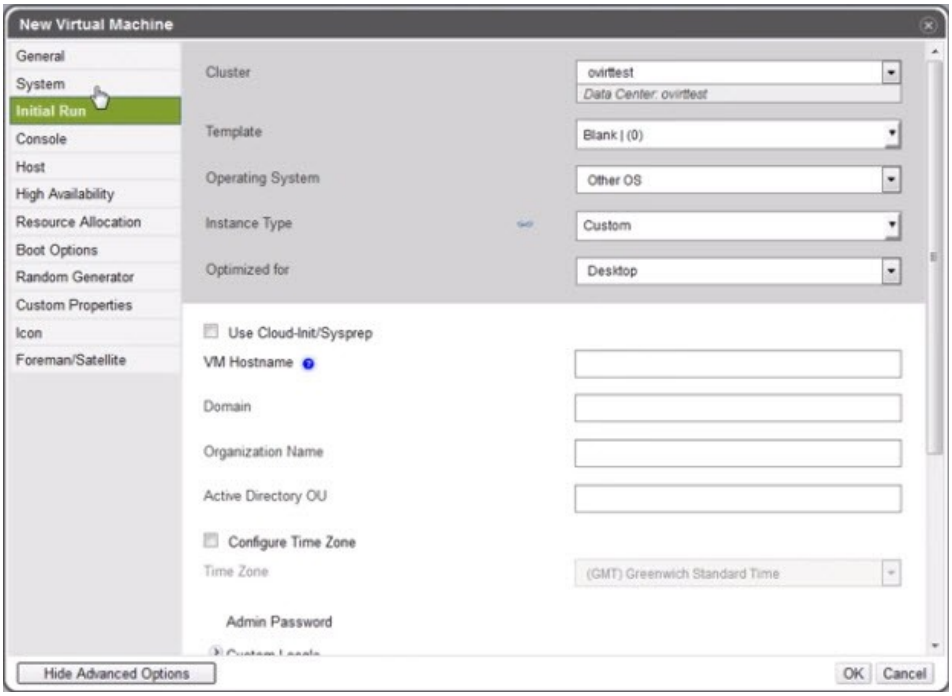
```
Console:
        USB support enabled
Ok
```

Αν πατήσουμε το *Show Advanced Options* στο κάτω αριστερό μέρος του παραθύρου:



Εικόνα 30: Δημιουργία VM, επιλογή Show Advanced Options

Με αυτή την επιλογή μπορούμε να επιλέξουμε τώρα τον τρόπο με τον οποίο θέλουμε να ξεκινήσει την πρώτη φορά το VM μας (Initial Run), το Power Management του, τα Fencing Policies κ.ά.



Εικόνα 31: Περιεχόμενα κατηγορίας Initial Run στη δημιουργία του νέου VM.

2.4.3.7. Αρχικό BOOT VM μέσω των iso

SAMBA από Linux σε Linux :

και στα δύο μηχανήματα: `/usr/bin/smbclient -L <username>`
Στο μηχάνημα με τα Windows: `/usr/bin/smbclient \\\\192.168.10.242\\iso <password>`

SAMBA από Windows σε Linux :

Ανοίγουμε ένα παράθυρο και στην γραμμή εντολών επιλέγουμε το *Map Network Drive.*
\\192.168.10.242\iso

Τελικά, εκτελούμε τις παρακάτω εντολές στο σύστημα oVirt:

```
# cd /home/oVirt/iso
# ls -lah c1a83312-6b91-45ea-b0e1-f2544537349/
# chmod -R 777 c1a83312-6b91-45ea-b0e1-f2544537349/
# chown -R nobody:nobody c1a83312-6b91-45ea-b0e1-f2544537349/
# chcon -t samba_share_t c1a83312-6b91-45ea-b0e1-f2544537349/
```

και τώρα όλα τα .iso αρχεία που έχουμε στον host μας εμφανίζονται στο παραθυρικό μενού που εμφανίζεται όταν τρέχουμε το Run Once στην Virtual Machine και μπορούμε πλέον να κάνουμε boot από κάποιο από τα .iso που υπάρχουν στην λίστα και να εγκαταστήσουμε τα Linux στην εικονική μηχανή.

Μια άλλη περίπτωση είναι να μην διαλέξουμε τίποτα στο *Initial Boot* αλλά να επιλέξουμε κατευθείαν να γίνει η δημιουργία της εικονικής μηχανής. Σε μία τέτοια περίπτωση, μόλις δημιουργηθεί η εικονική μηχανή

πάμε στην καρτέλα με τις εικονικές μηχανές, κάνουμε κλικ στο Linux Mint και επιλέγουμε *Run Once* για να επιλέξουμε τον τρόπο με τον οποίο επιθυμούμε να γίνει η εγκατάσταση του λειτουργικού συστήματος της εικονικής μηχανής.

```
Boot options: Attach CD
```

Κάνουμε *refresh* και επιλέγουμε το *Linux_Mint.iso* και στη συνέχεια πατάμε το *ok*. Στο *Boot Order,* μετακινούμε το CD-ROM πιο ψηλά στη στοίβα. Πατάμε *ok* και η εικονική μηχανή κάνει το πρώτο της boot up.

Η διαδικασία της εγκατάστασης του λειτουργικού συστήματος μέσα στην εικονική μηχανή είναι η συνηθισμένη όπου επιλέγουμε ρυθμίσεις για ημερομηνία και ώρα, ορίζουμε τα partitions, ορίζουμε χρήστες, ρυθμίζουμε το δίκτυο και τα λοιπά. Καθώς μπορούμε να επιλέξουμε σε αυτό το σημείο να μην ορίσουμε από τώρα το δίκτυο, επιλέγουμε να συνεχίσουμε, χωρίς το δίκτυο να είναι ρυθμισμένο, με την εγκατάσταση του λειτουργικού και θα κάνουμε αργότερα τις ρυθμίσεις του δικτύου που χρειάζονται είτε μέσα από την εικονική μηχανή είτε μέσα από τον manager των εικονικών μηχανών.

Παρατηρήσεις:

Αυτό που έχουμε φτιάξει είναι μία self-hosted Engine. Οπότε μπορούμε στο web interface του oVirt, να πάμε στους hosts και να επιλέξουμε τον host μας (π.χ. test_host1, στην περίπτωσή μας). Ουσιαστικά επιλέγουμε το Dell μηχάνημά μας στο οποίο έχουμε εγκαταστήσει το oVirt. Οπότε τώρα αν επιλέξουμε από το drop down menu του *Management* το *Restart*, ουσιαστικά αυτό που θα κάνουμε είναι να δώσουμε εντολή στον υπολογιστή μας να κάνει reboot το φυσικό μηχάνημά μας. (το Dell).

Αν μετά από shutdown του φυσικού μηχανήματος, ο host στο web interface του oVirt είναι σε κατάσταση *Non-Operational*, πάμε στις τελείες και επιλέγουμε το "*Confirm 'Host Has Been Rebooted'"*. Αυτό το κάνουμε αφού πρώτα έχουμε κάνει *reboot* ή έπειτα από χειροκίνητο τερματισμό του μηχανήματος μας.

2.4.3.7.1. Set-up το δίκτυο στην εικονική μηχανή

Αρχικά δοκιμάσαμε να κάνουμε Bonding VLAN Bridge, αλλά δεν είχε το επιθυμητό αποτέλεσμα.

Η διαδικασία που ακολουθήσαμε θα δοθεί παρακάτω απλά για αναφορικούς λόγους. Ουσιαστικά είχαν εκδοθεί VNICs των εικονικών μηχανών ως Slaves με Master το Bond0, το οποίο ήταν και Master της φυσικής NIC του host μας (συσκευή em1).

Ενώ η διαδικασία για τη δημιουργία σύνδεσης μεταξύ της εικονικής κάρτας δικτύου και της φυσικής κάρτας δικτύου ήταν σωστή, δεν έγινε εφικτή η σύνδεση της εικονικής μηχανής στο διαδίκτυο.

Σε περίπτωση που υπάρχει πρόβλημα σύνδεσης στο δίκτυο μπορούμε να δώσουμε τις εξής δύο εντολές ώστε να δούμε τα logs και αν μπορέσουμε να βρούμε τον λόγο που δημιουργήθηκε το πρόβλημα ώστε να προχωρήσουμε στην επίλυσή του.

```
# systemctl status network. Service
# journalctl -xe
```

Το πρόβλημα που αναφέρθηκε από την εντολή `journalctl` ήταν το εξής:

```
failed to start lsb: bring up/down networking
```

2.4.3.7.1.1. Bonding VLAN Bridge

Για να κάνουμε το Bonding VLAN Bridge ακολουθούμε την εξής διαδικασία:

1. Απενεργοποιούμε τον Network Manager:

   ```
   #  systemctl stop NetworkManager
   #  systemctl disable NetworkManager
   ```

   ή

   ```
   #  service NetworkManager stop
   #  chkconfig --level 2345 NetworkManager off
   ```

2. Ενεργοποιούμε και εκκινούμε την υπηρεσία network:

   ```
   #  systemctl enable network
   #  systemctl start network
   ```

   ή

   ```
   #  chkconfig --level 2345 network on
   #  service network start
   ```

3. Ορίζουμε ότι θέλουμε το bonding να ξεκινάει μαζί με την εκκίνηση του συστήματος:

   ```
   #  cat > /etc/modprobe.d/bonding.conf << EOF
   #  > alias bond0 bonding
   #  > EOF
   ```

4. Ορίζουμε το bond ως εξής:

   ```
   #  cat > /etc/sysconfig/network-scripts/ifcfg-bond0 << EOF
   ```

```
#  >DEFINE=bond0
#  >NM_CONTROLLED=no
#  >USERCTL=no
#  >BOOTPROTO=none
#  >BONDING_OPTS="mode=4 minmon=100"
#  >TYPE=Ethernet
#  >MTU=900
#  >EOF
```

5. Τώρα μπορούμε να κάνουμε enslave τα φυσικά NICs του συστήματός μας στο bond ως εξής:

```
#  cat > /etc/sysconfig/network-scripts/ifcfg-em1 << EOF
#  >NM_CONTROLLED="no"
#  >BOOTPROTO="none"
#  >DEVICE= "em1"
#  >ONBOOT="yes"
#  >USERCTL=no
#  >MASTER=bond0
#  >SLAVE=yes
#  >EOF
```

6. Επαναλαμβάνουμε το βήμα 5 για τη συσκευή δικτύου em2:

```
#  cat > /etc/sysconfig/network-scripts/ifcfg-em2 << EOF
#  >NM_CONTROLLED="no"
#  >BOOTPROTO="none"
#  >DEVICE= "em2"
#  >ONBOOT="yes"
#  >USERCTL=no
#  >MASTER=bond0
#  >SLAVE=yes
#  >EOF
```

7. Τώρα μπορούμε να δημιουργήσουμε τα VLAN interfaces πάνω στο bond. Εδώ θα χρησιμοποιήσουμε το VLAN ID 1 και VLAN ID 2. Η διαδικασία έχει ως εξής:

```
#  cat > /etc/sysconfig/network-scripts/ifcfg-bond0.1 << EOF
#  >DEVICE= bond0.1
#  >VLAN=yes
#  >BOOTPROTO=none
#  >NM_CONTROLLED=no
#  >BRIDGE=br0
#  >MTU=1500
#  >EOF
```

8. Επαναλαμβάνουμε για το VLAN ID 2:

```
#  cat > /etc/sysconfig/network-scripts/ifcfg-bond0.2 << EOF
#  >DEVICE= bond0.2
#  >VLAN=yes
#  >BOOTPROTO=none
#  >NM_CONTROLLED=no
#  >BRIDGE=ovirtmgmt
#  >MTU=9000
#  >EOF
```

9. Τώρα δημιουργήσουμε τις γέφυρες (bridges) πάνω στις οποίες βρισκονται τα VLAN interfaces. Μπορούμε να τα ονομάσουμε όπως θέλουμε, αλλά το ένα από τα δύο bridges πρέπει υποχρεωτικά να ονομάζεται **ovirtmgmt**. Η διαδικασία έχει ως εξής:

```
#  cat > /etc/sysconfig/network-scripts/ifcfg-ovirtmgmt << EOF
#  >TYPE=Bridge
#  >NM_CONTROLLED= “no”
#  >BOOTPROTO= “none”
#  >DEVICE= “ovirtmgmt”
#  >ONBOOT= “yes”
```

```
#  >IPADDR=192.21.10.14
#  >NETMASK=255.255.255.0
#  >GATEWAY=192.21.10.1  #gateway address does into management network
#  >EOF

#  cat > /etc/sysconfig/network-scripts/ifcfg-br0 << EOF
#  >TYPE=Bridge
#  >NM_CONTROLLED= “no”
#  >BOOTPROTO= “none”
#  >DEVICE= “br0”
#  >ONBOOT= “yes”
#  >IPADDR=192.21.10.14
#  >NETMASK=255.255.255.0
#  >DEFROUTE=no
#  >EOF
```

10. Επανεκκινούμε το δίκτυο για να τεθούν σε ισχύ οι αλλαγές που κάναμε:

```
#  systemctl restart network
```

ή

```
#  service network restart
```

Σε περίπτωση που υπάρχει κάποιο πρόβλημα με το δίκτυο, μπορούμε πάντα να ελέγχουμε τα logs του συστήματός μας με τις εντολές:

```
# systemctl status network.service
# journalctl -xe
```

2.4.3.7.2. Παρατηρήσεις

1. Με yum update που έγινε αργότερα στο σύστημά μας, έγινε αναβάθμιση και το oVirt στην καινούρια του έκδοση (4.3). Στην καινούρια έκδοση η ρύθμιση του δικτύου στις εικονικές

μηχανές έχει απλοποιηθεί πάρα πολύ και μπορούμε με «δύο κλικ» να έχουμε έτοιμο και σωστά ρυθμισμένο το δίκτυο για την εικονική μας μηχανή.

2. Ούτως ή άλλως θα δοκιμάσουμε να βάλουμε ένα switch (συγκεκριμένα το Edge RouterX της εταιρείας Ubiquiti το οποίο είναι router και firewall) και να συνδέσουμε πάνω του τις εικονικές μηχανές ώστε να είμαστε πιο κοντά και στις πραγματικές περιπτώσεις που συμβαίνουν καθημερινά, παραδείγματος χάριν σε εργαστήρια σχολών, σε σχολεία, σε εταιρείες και επιχειρήσεις. Καθώς οι εικονικές μηχανές δεν θα τρέχουν πάνω σε ένα φυσικό μηχάνημα, αλλά θα εκτελούνται πάνω σε ξεχωριστά μηχανήματα μέσα σε ένα περιβάλλον εργαστηρίου και ο κάθε υπολογιστής ξεχωριστά θα είναι συνδεδεμένος πάνω σε μία δικτυακή γραμμή. Αν οι εικονικές μηχανές ήταν όλες πάνω στον έναν, αρχικό host με τα CentOS, τότε θα μπορούσαμε απλά να έχουμε στον αρχικό host μία γέφυρα και να δίνουμε με αυτόν τον τρόπο δίκτυο στις εικονικές μηχανές. Εφόσον όμως οι εικονικές μηχανές θα βρίσκονται σε ξεχωριστά μηχανήματα, τότε θέλουμε όλες να ανήκουν στο ίδιο VLAN ή να περιορίζεται κάπως η πρόσβαση τους στο διαδίκτυο ή αν επιθυμούμε να επικοινωνούν μόνο μεταξύ τους. Το Edge RouterX θα αναφέρεται σε επόμενη ενότητα, όπου και θα ασχοληθούμε με την αρχική του εγκατάσταση, το NAT, το switching, το management του, τα routing tables και πολλά άλλα.

3. Επίσης, στο νέο σύστημα που δημιουργήσαμε με τα CentOS 7, οι εικόνες που δημιουργούμε θα πηγαίνουν στο directory `/home/vm/images`, καθώς αλλάξαμε το αρχικό path μετέπειτα, αφού ο αποθηκευτικός χώρος για τις εικονικές μηχανές δεν ήταν αρκετός στο root directory (/). Δοκιμάσαμε την εντολή `resize2fs`, αρχικά για να μικρύνουμε το partition που ήταν το `/home` και έπειτα για να αυξήσουμε το partition του root directory /. Αλλιώς μπορούμε απλά να βγάλουμε όλο το `/home` και να το βάλουμε στο root directory. Μερικές από τις εντολές που μπορεί να μας φανούν χρήσιμες για αυτή τη διαδικασία (αυξομείωση μεγέθους των partitions) είναι οι εξής:

```
# partprobe
# frisk
# swapoff
# mkswap
# swapon
# resize2fs
```

4. Μέσω ενός άλλου host με Windows (σε περίπτωση που δεν μπορούμε να βρισκόμαστε στον αρχικό host όπου είναι τα CentOS και το oVirt Engine) μπορούμε να κάνουμε απομακρυσμένο έλεγχο του με το λογισμικό PUtty και δίνουμε τη διεύθυνση hostname και την θύρα για SSH σύνδεση, στο παράδειγμά μας είναι 192.21.10.14:22 όπου και μπορούμε να συμπληρώσουμε τη διεύθυνση προορισμού με την οποία θέλουμε να να συνδεθούμε. Πατάμε το Ok και κάνουμε Login as root και μπαίνουμε κανονικά στην κονσόλα του CentOS συστήματός μας.

5. Μετά από επανεκκίνηση του CentOS συστήματός μας δεν λειτουργούσε πλέον το oVirt Engine σωστά. Προσπαθήσαμε μέσω των logs και των καταγεγραμμένων αρχείων και σημειώσεων μας να επιλύσουμε το πρόβλημα που παρουσιάστηκε, αλλά στο τέλος χρειάστηκε να αναμορφώσουμε ολόκληρο το σύστημα και να ξανά ξεκινήσουμε από την αρχή, με την

εγκατάσταση των CentOS. Σε αυτή τη φάση, επιλέξαμε να προχωρήσουμε από το βασικό single-hosted περιβάλλον και να πάμε στην εγκατάσταση των oVirt Nodes σε τρεις (3) servers, ώστε να μπορέσουμε πια να δούμε το πλήρες φάσμα των δυνατοτήτων του oVirt, συμπεριλαμβανομένου του Gluster.

6. Με την καινούργια έκδοση του oVirt έχουν αλλάξει μερικά πράγματα οπως:
    1. Αλλαγή του GUI και του web interface
    2. Δεν μας ρωτάει πια αν θέλουμε NFS κατά τη διάρκεια του configuration του Engine.
    3. Όταν φτιάχνουμε host θέλει να ξέρει για το Power Management. Εφόσον ενεργοποιούμε το Power Management στον host, τότε δεν μπορεί να προχωρήσει η διαδικασία εάν δεν δώσουμε fence agent.

7. Fence agents

    Η oVirt Engine μπορεί αυτόματα να παρέχει fence agents. Για να εκτελεστεί σωστά το fencing, υπάρχουν τρεις προϋποθέσεις :
    1. Το fencing πρέπει να έχει ρυθμιστεί και ενεργοποιηθεί στον host.
    2. Να υπάρχει ένας έγκυρος proxy host (ενός ακόμα host δηλαδή στο ίδιο Data Center με το status του να είναι up)
    3. Η σύνδεση με τον host να έχει γίνει timed out.

    Την πρώτη φορά που το network πέφτει, το status αλλάζει σε connecting. Τότε η μηχανή θα προσπαθήσει τρεις φορές ακόμα να ζητήσει το status του VDSM ή θα περιμένει για ένα χρονικό διάστημα το οποίο εξαρτάται από το φορτίο εργασίας που θα έχει ο host. Για παράδειγμα εάν το VDSM δεν αποκρίνεται, η μηχανή πρέπει να τον ρωτήσει τρεις φορές για το status του, οπότε μπορεί να περιμένουμε μέχρι και 9 λεπτά. Εάν όμως δεν αποκριθεί μέσα σε αυτόν τον χρόνο, το status του θα αλλάξει σε non responsive και στη συνέχειεα θα γίνει fenced. (Δηλαδή, σε αθλητική ορολογία, ο host πού σταματάει να αποκρίνεται και δεν μπορεί πια "να παίξει μπάλα, θα πρέπει να κάτσει στον πάγκο")

    Προσοχή:
    Η διαδικασία SSH Soft Fencing εκτελείται επίσης σε hosts στους οποίους δεν έχει ρυθμιστεί το Power Management σε αντίθεση με το κανονικό Fencing το οποίο εκτελείται μόνο στους hosts που έχει ρυθμιστεί το Power Management.

8. Όταν θέλουμε να κάνουμε επεξεργασία, συντήρηση ή να καταστρέψουμε ένα domain (storage) του συστήματος, πρέπει να βάλουμε τον host για maintenance και μετά από αυτή την πράξη γίνονται διαθέσιμες οι παραπάνω επιλογές.

9. Όταν σβήνουμε κάποιο λάθος storage domain, πολλές φορές, θα χρειαστεί να πάμε στο path (πχ /oVirt/iso) και να εκτελέσουμε την εντολή

```
#  # rm -rf /oVirt/iso/*
```

με την οποία μπορούμε να διαγράψουμε όλα τα αρχεία που υπάρχουν μέσα στο συγκεκριμένο directory.

Περισσότερες περιπτώσεις:

- Περίπτωση που το fencing αποτυγχάνει (για παράδειγμα δεν μπορούμε να επανεκκινήσουμε τον host), δεν μπορούμε να ξαναπροσπαθήσουμε και η κατάστασή του έχει αλλάξει σε non responsive status.
- Κατά τη διάρκεια που το Engine startup fencing γίνεται disabled, μπορούμε να θέσουμε τον χρόνο από την εκκίνηση στον οποίο θα απενεργοποιηθεί το fencing. Η default τιμή εδώ είναι 300 δευτερόλεπτα.
- Μόλις ο host επανεκκινηθεί, το Status του αλλάζει σε reboot για συγκεκριμένο χρόνο, η default τιμή του οποίου είναι 300 δευτερόλεπτα.
- Η διαδικασία fencing στο oVirt 4.3.1 έχει πλέον επεκταθεί στο SSH Soft Fencing, λειτουργία του οποίου είναι επανεκκίνηση του VDSM με χρήση σύνδεσης SSH.
- Η διαδικασία του SSH Soft Fencing είναι σχεδόν η ίδια με την από πάνω με την προσθήκη ότι εδώ πρέπει, σε περίπτωση που ο host δεν αποκρίνεται (μετά και τις τρεις επιπλέον φορές που θα προσπαθήσει η μηχανή να ρωτήσει το VDSM για το status του), να εκτελεστεί επανεκκίνηση του VDSM με χρήση σύνδεσης SSH. Αν η εκτέλεση της εντολής δεν είναι επιτυχής, τότε ο host πάει απευθείας στον “πάγκο”.

Τελικά:

Αφού κάναμε αναμόρφωση (format) στο μηχάνημά μας, ξανά κάνουμε εγκατάσταση των CentOS 7 και δίνουμε passphrase για το root directory, ώστε να μπορούμε να κάνουμε boot, και κωδικό για τον χρήστη root. Επίσης, δημιουργούμε έναν χρήστη, που τον ονομάζουμε mar1, με δικαιώματα διαχειριστή, και δίνουμε ένα αντίστοιχο κωδικό.

Αν για κάποιο λόγο όταν κάνουμε το partitioning, δώσουμε λάθος τιμές και θέλουμε π.χ. να αλλάξουμε τα μεγέθη, μπορούμε αργότερα με χρήση της εντολής `resize2fs` να αυξομειώσουμε τα partitions. Γενικά για τέτοια περίπτωση οι εντολές που θα μπορούσαμε να χρησιμοποιήσουμε είναι (με τη σειρά): `partprobe, fdisk, swapoff, mkswap, swapon, resize2fs`.

Κατά την διάρκεια της αρχικής εγκατάστασης, μπορούμε να δώσουμε απευθείας ποια σύνολα πακέτων μας ενδιαφέρουν και να εγκατασταθούν αυτόματα. Στη δικιά μας περίπτωση μας ενδιαφέρει το Virtualization οπότε και εγκαθιστούμε το virtualization package. Επίσης, επιλέγουμε το GUI (Graphical User Interface) πακέτο για να έχουμε γραφικό περιβάλλον ως χρήστες, αφού χρειάζεται να μπορούμε να ανοίγουμε το παράθυρο του προγράμματος περιήγησης και να διαχειριζόμαστε το oVirt σύστημά μας μέσω ενός προγράμματος περιήγησης.

Κατά τη διάρκεια της εγκατάστασης μας ζητήθηκε επίσης να δώσουμε hostname και domainname. Στην περίπτωσή μας δώσαμε `thch` για hostname και `rhel7domain` για domainname, έτσι ώστε να μπορούμε μέσω του προγράμματος περιήγησης να πληκτρολογήσουμε τη διεύθυνση http://thch.rhel7domain και να μπούμε στο web interface που παρέχεται για τη διαχείριση του oVirt.

Μετά το πέρας της εγκαταστάσεως, μπορούμε να εγκαταστήσουμε το oVirt ακολουθώντας τις οδηγίες που αναφέρουμε προηγουμένως στο αντίστοιχο κεφάλαιο.

Ανοίγουμε κονσόλα και με χρήση της εντολής yum προχωράμε στην ενημέρωση των εγκατεστημένων πακέτων, ώστε να είναι ενημερωμένο το σύστημά μας και να είμαστε σίγουροι ότι τρέχουμε τα current versions.

```
# yum update
```

Για να κατεβάσει το rpm πακέτο oVirt και να το κάνει εγκατάσταση στο σύστημά μας.

```
# yum install http://resources.oVirt.org /pub/yum-repo/oVirt-release42.rpm
```

Για να είμαστε σίγουροι ότι έχουμε την τελευταία έκδοση oVirt στο σύστημά μας.

```
# yum update
```

Για να κάνουμε εγκατάσταση του oVirt Engine στο σύστημά μας και να μπορούμε μετά να κάνουμε εικονικές μηχανές διαχείρισή τους και πολλά άλλα που θα τα δούμε αργότερα.

```
# yum install ovirt-engine
```

Σε αυτό το σημείο, επειδή είχε εγκατασταθεί με την αρχική εγκατάσταση του λειτουργικού συστήματος CentOS, και το πακέτο προστασίας `ipa-server` και `freeipa-server`, δεν μπορούσαμε να προχωρήσουμε την εγκατάσταση της μηχανής του oVirt καθώς τα δύο αυτά πακέτα δεν υποστηρίζουν το ένα τη χρήση του άλλου. Οπότε και χρειάστηκε να διαγράψουμε τα πακέτα `ipa-server` και `freeipa-server` ώστε να μπορέσουμε να συνεχίσουμε με την εγκατάσταση του engine.

Διαγραφή πακέτων ipa-server και freeipa-server:

```
# yum erase *ipa-server
```

Εγκατάσταση του oVirt Engine:

```
# yum install oVirt-Engine
```

Με το πέρας της εγκατάστασης αυτής μπορούμε να εκτελέσουμε την εξής εντολή με την οποία θα μπορέσουμε να ρυθμίσουμε το configuration του oVirt Engine και να ξεκινήσουμε το virtualization που επιθυμούμε.

```
# engine-setup
```

Στο configuration που παρέχεται όταν εκτελούμε την εντολή `engine-setup`, αφήνουμε τα προεπιλεγμένα ως έχουν απλά πατώντας Enter. Μόλις τελειώσει να εκτελείται αυτή η εντολή, μπορούμε πλέον να ανοίξουμε έναν web browser (π.χ. Mozilla Firefox) και στην μπάρα του URL να δώσουμε τη διεύθυνση thch.rhel7domain/oVirt-Engine και να μπούμε στο web interface του oVirt.

Στην πρώτη σελίδα, επιλέγουμε να μπούμε στο admin portal και δίνουμε τα εξής στοιχεία:

```
User: admin
Password: R00+
Domain: internal
```

Το παραπάνω password μας ζητήθηκε κατά τη διάρκεια της εγκατάστασης του Engine και είναι κοινό με αυτό του χρήστη root στο σύστημα CentOS.

Βρισκόμαστε πλέον στο γενικό interface του oVirt έχουμε τη δυνατότητα να δημιουργήσουμε νέο domain, νεο cluster, νέα VMs, templates, pools ή ακόμα και χρήστες.

2.5. oVirt Node

Από τον υπεύθυνο προϊστάμενο του τμήματος του Κέντρου Διαχείρισης Δικτύου του Δημόκριτου, κύριο Κοροβέση Ιωάννη, μας διατέθηκαν τρία (3) μηχανήματα HP Proliant DL360 G5, με τα εξής χαρακτηριστικά:

Πόροι	**Χαρακτηριστικά**
CPU : Intel® Xeon® E5420@ 2.50GHz	• 2 CPU sockets • 4 CPU cores per socket
Φυσική Μνήμη RAM	23027 MB
HDD	• 70 GB για το λειτουργικό σύστημα • 300 GB για να χρησιμοποιηθούν στο Gluster
Network	2 NICs

Σε αυτά τα μηχανήματα θα εγκαταστήσουμε το oVirt Node, κάνοντας έτσι τους τρεις servers, nodes του oVirt, ώστε στη συνέχεια να μπορέσουμε να εγκαταστήσουμε το GlusterFS πάνω τους.

Το GlusterFS (Gluster File System) είναι ένα είδος συστήματος αρχείων με το οποίο μπορούμε να αυξήσουμε την ακεραιότητα των υπηρεσιών μας αλλά και να λύσουμε πολλά από τα προβλήματα που παρουσιάζονται όταν είμαστε σε παραγωγικό στάδιο. Για παράδειγμα, όταν προσφέρουμε κάποιες υπηρεσίες στους πελάτες μας, π.χ. livestreaming, αλλά, από τη μεριά μας, πρέπει να κλείσουμε για λίγο τον server για να του κάνουμε συντήρηση, τότε ο πελάτης μας θα μείνει χωρίς τη δυνατότητα να παρακολουθήσει τους προημιτελικούς του Μουντιάλ. Εάν κάτι τέτοιο συνέβαινε, σίγουρα θα χάναμε πολλούς πελάτες.

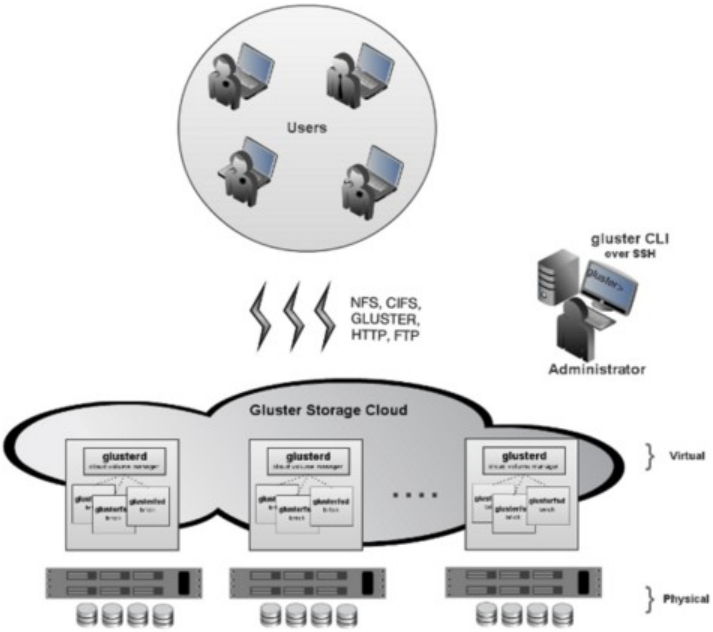
Το GlusterFS είναι ένα κλιμακωτό, κατανεμημένο σύστημα αρχείων το οποίο συναθροίζει τους αποθηκευτικούς πόρους πολλαπλών servers σε ένα μοναδικό και καθολικό namespace (global namespace).

Στα πλεονεκτήματα της χρήσης GlusterFS είναι ότι το σύνολο των αποθηκευτικών χώρων που χρησιμοποιούμε μπορεί να φτάσει μέχρι και μερικά petabytes (1024^5 PiB). Μέσω αυτού, τα συστήματά μας μπορούν να διαχειριστούν χιλιάδες πελάτες.

Επίσης στα πλεονεκτήματα της χρήσης GlusterFS συμπεριλαμβάνονται τα παρακάτω:

- Είναι συμβατό με POSIX.
- Χρησιμοποιεί κοινό hardware, που οι τιμές του δεν είναι της τάξεως των χιλιάδων.

- Μπορεί να χρησιμοποιήσει οποιοδήποτε ondisk filesystem που υποστηρίζει πρόσθετα χαρακτηριστικά.
- Είναι προσβάσιμο μέσω κοινών και συνηθισμένων πρωτοκόλλων όπως είναι το NFS και το SMB.
- Παρέχει αναπαραγωγή, ποσοστώσεις (quotas), γεωανατύπωση (geo-replication), στιγμιότυπα (snapshots) και ανίχνευση bitrot.
- Επιτρέπει βελτιστοποίηση για διαφορετικά φορτία εργασίας.
- Είναι Open Source.



Εικόνα 32: Gluster Αρχιτεκτονική

Οι επιχειρήσεις μπορούν να προσαρμόζουν την παραγωγική ικανότητα, τις επιδόσεις και τη διαθεσιμότητα, χωρίς να κλειδώνουν τον πωλητή, σε κοινόχρηστο χώρο, δημόσιο σύννεφο και υβριδικά περιβάλλοντα. Το GlusterFS χρησιμοποιείται στην παραγωγή σε χιλιάδες επιχειρήσεις που αφορούν τα μέσα ενημέρωσης, την υγειονομική περίθαλψη, την κυβέρνηση, την εκπαίδευση, το web 2.0 και τις χρηματοπιστωτικές υπηρεσίες.

Το GlusterFS μπορεί να εφαρμοστεί και χειροκίνητα σε συνδυασμό με το Vargant. Στην περίπτωσή μας το GlusterFS παρέχεται μαζί με το .iso εγκατάστασης των nodes του oVirt και η δημιουργία, εγκατάσταση και εφαρμογή του γίνεται μέσω παραθυρικού περιβάλλοντος και όχι μέσω της κονσόλας.

Με το GlusterFS, αυτό που καταφέρνουμε είναι να μπορούμε να συνεχίσουμε να παρέχουμε τις υπηρεσίες για τις οποίες μας έχει προτιμήσει ο πελάτης, αλλά ταυτόχρονα να έχουμε κατεβάσει τον server για συντήρηση. Αυτό επιτυγχάνεται καθώς με τον GlusterFS, έχουμε όχι έναν αλλά τρεις server να παρέχουν τις υπηρεσίες μας. Όταν ο κεντρικός server είναι υγιής και ενεργός, είναι και αυτός που παρέχει τις υπηρεσίες μας. Οι άλλοι δύο servers στέκονται εφεδρικοί , στην περίπτωση που ή χρειαστεί ο main server να “κατέβει” για συντήρηση ή υπάρξει οποιοδήποτε άλλο πρόβλημα με το οποίο καθίσταται ανίκανος να παρέχει τις υπηρεσίες του (π.χ. κάποιο βραχυκύκλωμα). Σε αυτή τη περίπτωση θα αναλάβει την θέση του Hypervisor και της oVirt Engine ένας από τους δύο εφεδρικούς servers μέσω κατάλληλου αλγόριθμου.

Από τη σελίδα https://oVirt.org/Node κατεβάζουμε το .iso αρχείο (το οποίο και θα κάνοουμε burn σε ένα DVD-R) για να μπορέσουμε να κάνουμε την εγκατάσταση του oVirt-node απευθείας στους servers.

2.5.1. Προϋποθέσεις

Οι βασικές προϋποθέσεις για την εγκατάσταση του oVirt-node είναι:

Hardware: Τρεις (3) μηχανές με 16GB μνήμη RAM τουλάχιστον και επεξεργαστές που έχουν την δυνατότητα παροχής επεκτάσεων εικονιοποίησης υλικού (hardware virtualization extensions). Γενικά, μπορούμε να κάνουμε τεστ και σε virtualized περιβάλλοντα μέσω του nested KVM, αλλά το καλύτερο είναι να έχουμε πρόσβαση και χρήση πραγματικών, φυσικών μηχανών.

Software: oVirt Node 4.2 (https://www.oVirt.org/node/). Το oVirt Node είναι ένα streamlined λειτουργικό σύστημα το οποίο βασίζεται στο CentOS 7. Το oVirt Node θα χρειαστεί για τους τρεις servers και ένα CentOS-based appliance image για το Engine VM.

Δίκτυο: Το όνομα του host υπολογιστή που χρησιμοποιούμε πρέπει να επιλύεται σωστά, είτε μέσω του DNS του δικτύου μας, είτε μέσω του αρχείου / etc / hosts στον virt host σας, στο VM που θα φιλοξενήσει την oVirt Engine και σε οποιουσδήποτε πελάτες μέσω των οποίων θα διαχειριζόμαστε το oVirt. Δεν είναι απολύτως απαραίτητο, αλλά είναι καλή ιδέα να αφήσουμε είναι χωριστά τα δίκτυα, ένα δίκτυο αποθήκευσης για την Gluster κίνηση και ένα διαφορετικό δίκτυο για το VM migration. *Στο εργαστήριό μου, χρησιμοποιώ ένα ξεχωριστό 10G nic σε κάθε έναν από τους κεντρικούς υπολογιστές για το δίκτυο αποθήκευσης μου.*

Storage: Η hosted Engine λειτουργία απαιτεί να έχουμε NFS, iSCSI, FibreChannel ή Gluster **καθαρούς** αποθηκευτικούς χώρους για να σταθεί η VM η οποία θα φιλοξενήσει την Engine. Εμείς, χρησιμοποιούμε ένα Gluster arbiter volume, το οποίο περιλαμβάνει την δημιουργία τριών (3) αντιγράφων Gluster volume. Το κάθε ένα από αυτά θα είναι επιπλέον χωρισμένο σε τρία bricks, με τα δύο να είναι τυπικά data bricks και το τρίτο arbiter brick να είναι αυτό στο οποίο θα αποθηκεύονται μόνο τα file names και metadata, παρέχοντας με αυτόν τον τρόπο σε ένα oVirt hosted Engine setup το data consistency που απαιτείται, καθώς υπάρχει σημαντική μείωση στα αντεγραμμένα δεδομένα και στην κίνηση του δικτύου.

2.5.2. Πώς κάνουμε το .iso bootable

Η διαδικασία για να κάνουμε το oVirt-node.iso αρχείο, το οποίο βρίσκουμε στην ιστοσελίδα του oVirt[11] [12], bootable είναι η εξής:

```
#  mkdir /mnt/disc

#  cd   /home/oVirt/iso/c1a83312-6b91-45ea-b0e1-f2544537349/images/11111111-
   1111-1111-1111-1111-11111111

#  ls -lah

#  mount -o loop -t iso9660 CentOS7-x86_64.iso /mnt/disc

#  ls -lah /mnt/disc

#  ls -lah /mnt/disc/isolinux

#  mkdir /home/temp

#  mkdir /home/temp/isolinux

#  cd /home/temp
```

```
# cp /mnt/disc/isolinux/* isolinux/

# chmod u+w isolinux/*

# mkisofs -o /home/oVirt/iso/c1a83312-6b91-45ea-b0e1-
f2544537349/images/11111111-1111-1111-1111-1111-11111111/oVirt-node-
installer.iso -b isolinux.bin -c boot.cat -no-emul-boot boot-load-size 4
-boot-info-table -R -J -v -T isolinux/

# cd /home/oVirt/iso/c1a83312-6b91-45ea-b0e1-f2544537349/images/11111111-
1111-1111-1111-1111-11111111

# sha1sum oVirt-node.iso

# sudo growisofs -speed=1 -dvd-compat -Z /dev/sr0=oVirt-node.iso
```

Την τελευταία εντολή, η οποία γράφει το dvd, θα μπορούσαμε να την αντικαταστήσουμε χρησιμοποιώντας το παραθυρικό περιβάλλον της εφαρμογής K3b.

2.5.3. Εγκατάσταση Node

Θα εγκαταστήσουμε την έκδοση του OVirt Node 4.2.3.1 σύμφωνα με το έκτο κεφάλαιο του documentation του oVirt Node που βρίσκεται στην ιστοσελίδα https://www.oVirt.org/documentation/install-guide/chap-oVirt_Nodes καθώς και τον οδηγό “Up and Running with oVirt 4.2 and Gluster Storage” που βρίσκεται στην ιστοσελίδα https://www.oVirt.org/blog/2018/02/up-and-running-with-oVirt-4-2-and-gluster-storage.

1. Κατεβάζουμε το .iso αρχείο του oVirt Node (ουσιαστικά είναι minimal CentOS λειτουργικό με τις επεκτάσεις και τα πακέτα που χρειάζονται για να κάνουμε το μηχάνημά μας να τρέχει το oVirt).

2. Κάνουμε το .iso burn σε κάποιο media (usb, cd, dvd κλπ).

```
# dd if=/oVirt-node-version.iso of=/dev/sdX
```

3. Κάνουμε boot από το media με το oVirt Node OS.

4. Εγκατάσταση του λειτουργικού συστήματος του oVirt Node. Κατά τη διάρκεια της εγκατάστασης, δηλώνουμε μόνο password για τον χρήστη root και δεν δημιουργούμε απλούς χρήστες στο node. **ΠΡΟΣΟΧΗ**: Για να γίνει η εγκατάσταση του node πρέπει ο σκληρός δίσκος στον οποίο θα εγκατασταθεί το λειτουργικό σύστημα να είναι εντελώς καθαρός. Αυτό το καταφέρνουμε είτε με χρήση του gparted live cd ή μέσω κονσόλας και της εντολής `fdisk /dev/sda` ή μέσω της εντολής `dd` με την οποία μπορούμε να αντιγράψουμε το μηδενικό αρχείο που βρίσκεται στο μονοπάτι `/dev/zero`.

5. Μετά την εγκατάσταση κάνουμε εγκατάσταση το oVirt appliance image στον hypervisor.

```
# yum install ovirt-engine-appliance-4.2-20180617.1.el7.noarch.rpm
```

το οποίο υπάρχει στην ιστοσελίδα https://resources.oVirt.org/pub/oVirt-4.2-snapshot/rpm/el7/noarch/oVirt-Engine-appliance-4.2-20180617.1.el7.noarch.rpm.

6. Ανοίγουμε έναν web browser (π.χ. Firefox Mozilla) και συνδεόμαστε με τον hypervisor μέσω του Cockpit και της θύρας 9090, δίνοντας στο πεδίο του URL είτε την IP διεύθυνση του μηχανήματος ή με το resolved name του: https://ov-node1.islab.local:9090/

7. Κάνουμε login στο Cockpit με τον root λογαριασμό μας

2.5.4. Πρώτη Εκτέλεση του Node

Στους servers όπου εγκαταστήσαμε το oVirt Node, πάμε και επαληθεύουμε ότι το δίκτυό μας είναι σωστά ορισμένο:

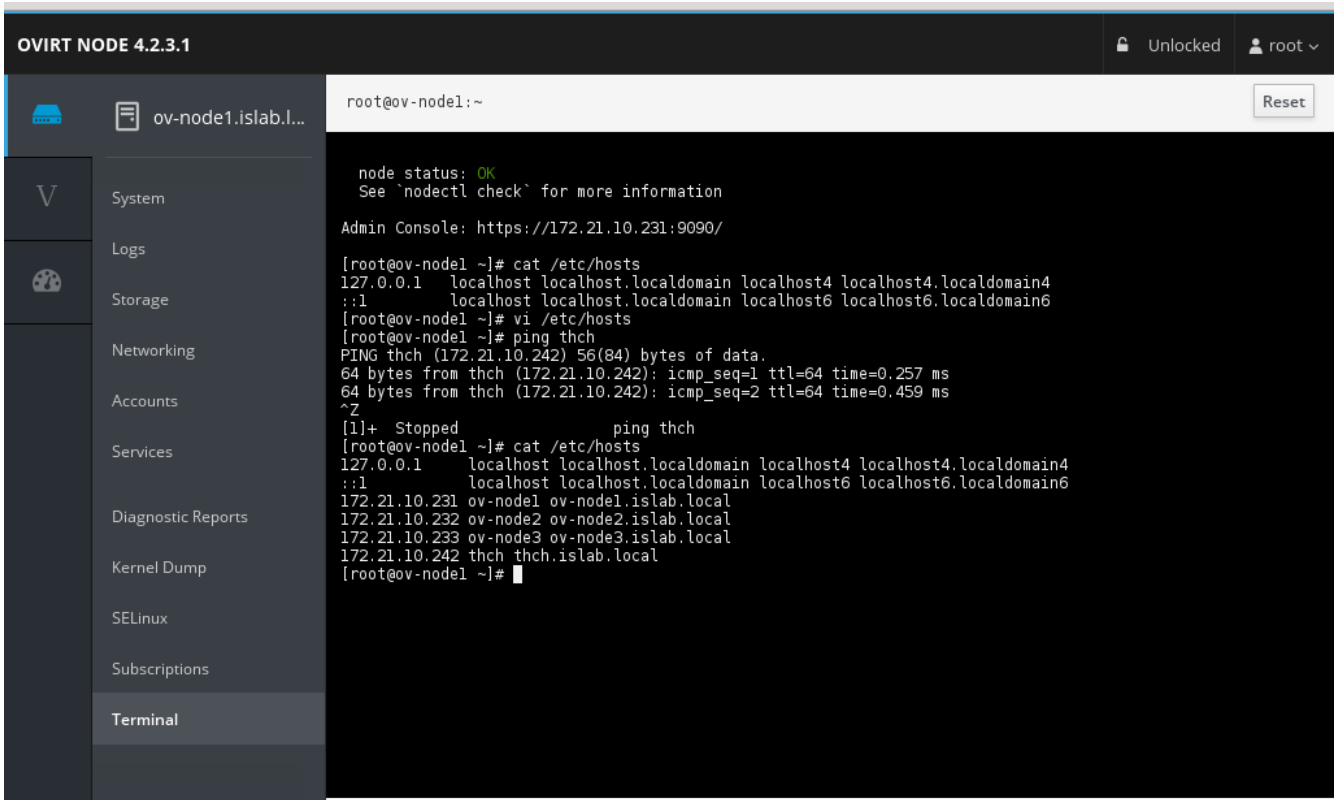
```
#  ifconfig ή # ip a
#  nmtui -> [x] Automaticallly Connect
#  systemctl restart network
#  ip a
#  exit
```

Τώρα μπορούμε να συνδεθούμε στο Cockpit interface με την διεύθυνση https://ip:9090 με τον root χρήστη του node, επιλέγουμε την επιλογή *"Reuse my password for privileged"* που υπάρχει κάτω από τα πεδία εισαγωγής των πληροφοριών του χρήστης μας, μπαίνουμε στο Administrative interface και από εδώ μπορούμε να ξεκινήσουμε την oVirt Engine.

Πάμε σε κάθε ένα από τα nodes (ov-node1.islab.local, ov-node2.islab.local, ov-node3.islab.local) και φτιάχνουμε τα αρχεία `/etc/hosts`, εάν δεν έχουμε ρυθμίσει τον DNS server, και προσθέτουμε τα εξής :

```
192.168.10.231 ov-node1 ov-node1.islab.local
192.168.10.232 ov-node2 ov-node2.islab.local
192.168.10.233 ov-node3 ov-node3.islab.local
192.168.10.234 ov-eng ov-eng.islab.local
192.168.10.242 thch thch.islab.local
```

Επίσης, έχουμε προσθέσει την διεύθυνση IP του oVirt Engine (ov-eng), καθώς παρότι δεν έχει δημιουργηθεί ακόμα, χρειάζεται να είναι αφιερωμένη η διεύθυνση σε αυτό καθώς θα χρειαστεί αργότερα όταν θα κάνουμε την εγκατάστασή του στο Node.



Εικόνα 33: Περιεχόμενο αρχείου /etc/hosts

Καθώς ο host (ov-node1) πρέπει να έχει πρόσβαση στον εαυτό του και στους άλλους δύο nodes, αλλά και στην Engine που θα εγκατασταθεί στον ov-node1 (και θα έχει διαφορετική IP διεύθυνση). Τα nodes πρέπει να έχουν πρόσβαση μέσω passwordless SSH, οπότε δίνουμε στην κονσόλα (του ov-node1) τις εξής εντολές:

```
#  ssh-keygen
#  ssh-copy-id root@ov-node1.islab.local
#  ssh-copy-id root@ov-node2.islab.local
#  ssh-copy-id root@ov-node3.islab.local
```

Αυτό χρειάζεται καθώς όταν θα πάμε μέσω του web interface του oVirt να δημιουργήσουμε το Gluster, το gdeploy εργαλείο που χρησιμοποιείται από το oVirt για αυτόν τον λόγο (δημιουργία Gluster) θα προσπαθήσει να κάνει login ως root στην κάθε μηχανή μας (σε κάθε node). Αν έχουμε δώσει password ή διαφορετικό μονοπάτι για το πού βρίσκεται το αρχείο με το κλειδί του SSH (`/root/.ssh/id_rsa.key`) όταν εκτελούμε την εντολή `ssh-keygen`, τότε η δημιουργία του Gluster με χρήση του gdeploy θα αποτύχει.

Θα αναφερθούμε περαιτέρω στο εργαλείο gdeploy σε μετέπειτα κομμάτι.

Από τον ov-node1 εκτελούμε τις παρακάτω εντολές :

```
#  ssh root@ov-node1.islab.local
```

```
#  exit (έξοδος από το ssh)
```

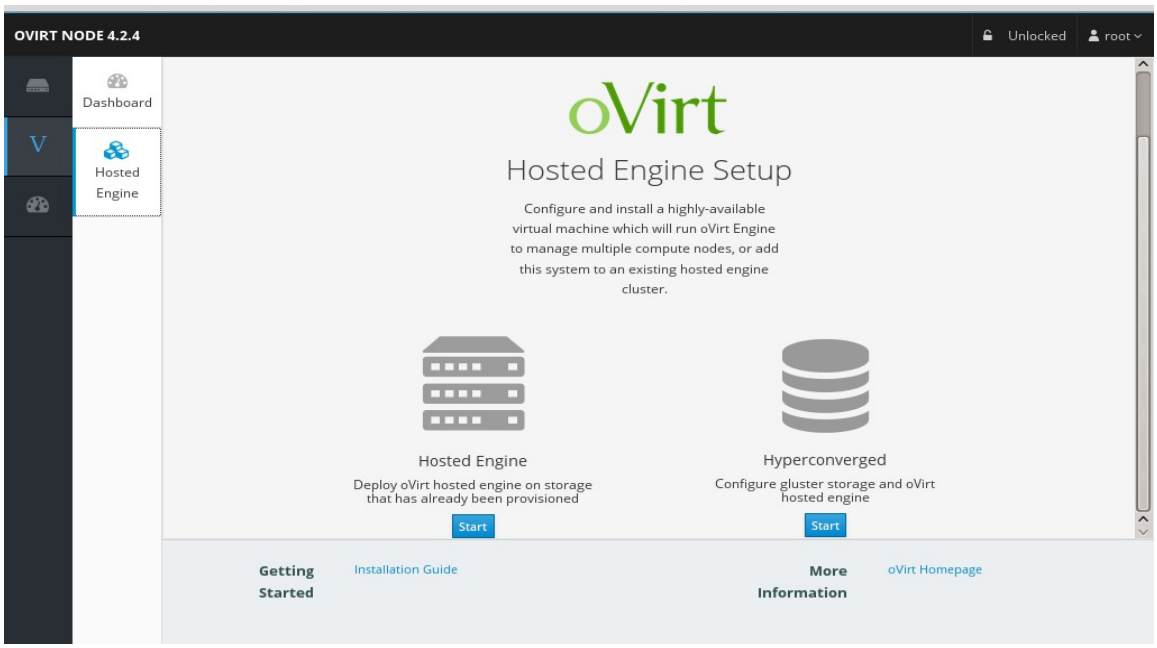
Εκτελούμε τις αντίστοιχες εντολές και για τους άλλους δύο nodes (`ssh root@ov-node2.islab.local`, `ssh root@ov-node3.islab.local`), ώστε να αποθηκευτούν τα κλειδιά τους στο αρχείο `~/.ssh/known_hosts`, όπου και αποθηκεύονται οι γνωστοί στον υπολογιστή μας hosts με τους οποίους θα μπορούμε να επικοινωνούμε μέσω passwordless ssh.

2.5.5. Gluster

Αφού έχει γίνει και αυτή η διαδικασία, πάμε μέσω κάποιου web browser στο URL του πρώτου Node, ov-node1.islab.local:9090, και βρισκόμαστε πλέον στο web interface του Cockpit, κάνουμε login με τον root account του ov-node1 και ακολουθούμε την εξής διαδικασία:

Βήμα 1

Επιλέγουμε την καρτέλα *Virtualization* και από εκεί επιλέγουμε το *Hyperconverged – Configure a gluster storage and oVirt hosted Engine.*

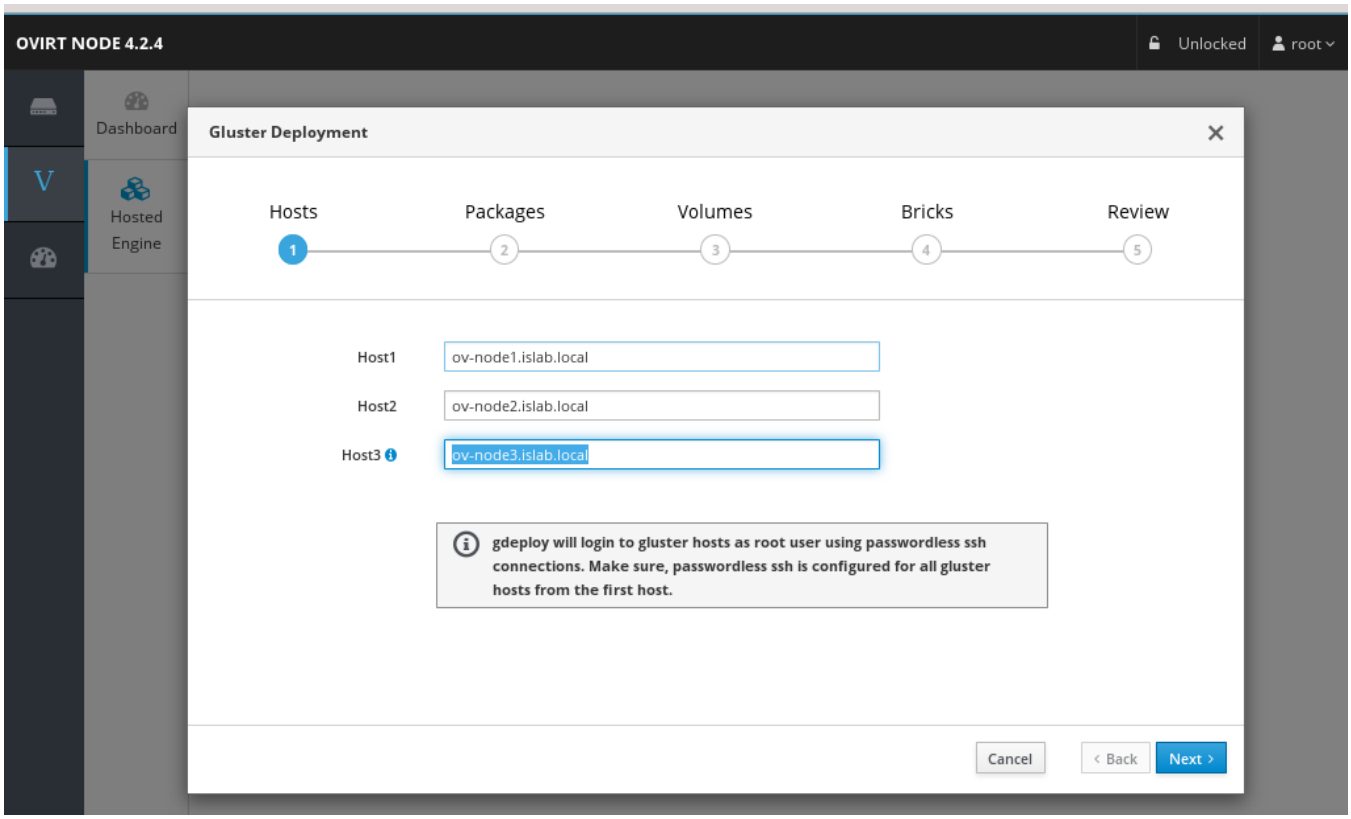


Εικόνα 34: Cockpit - Οθόνη για επιλογή δημιουργίας hosted engine στον node ή δημιουργία gluster και δημιουργία hosted engine.

Στο πρώτο βήμα για τη δημιουργία του GlusterFS αποθηκευτικού χώρου, πρέπει να δηλώσουμε τα ονόματα των nodes τα οποία θα φιλοξενήσουν το Gluster.

```
Host 1: ov-node1.islab.local
Host 2: ov-node2.islab.local
Host 3: ov-node3.islab.local
```

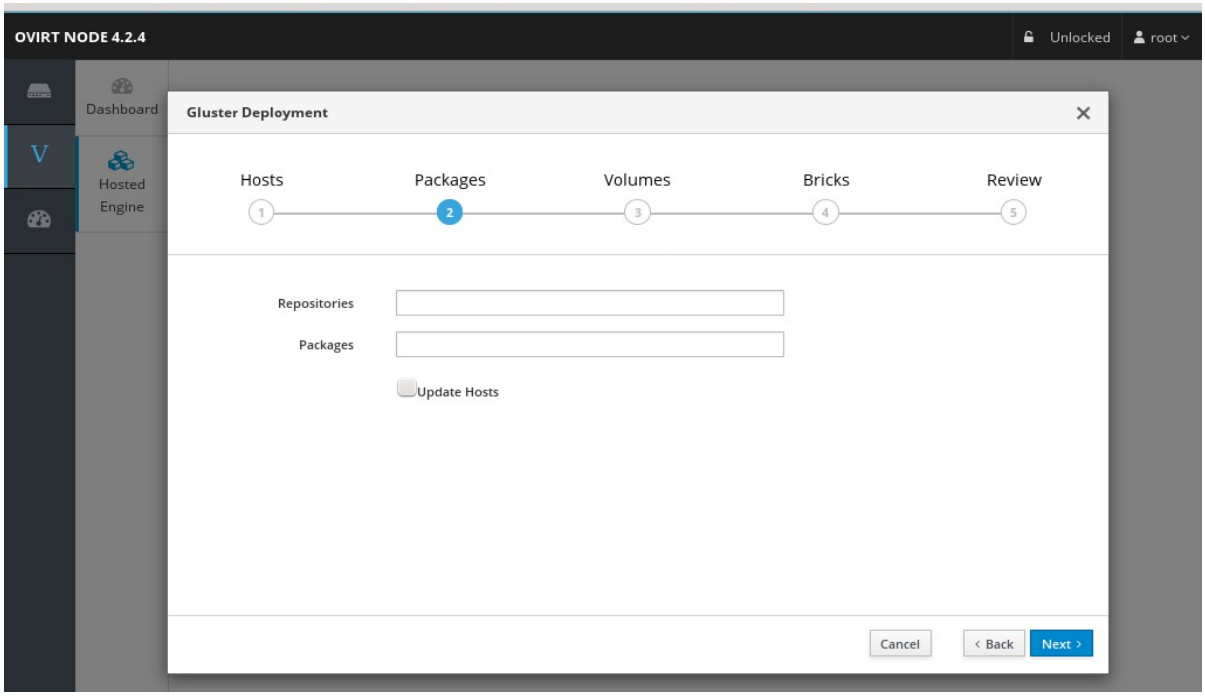
,όπου ο τρίτος host χρησιμοποιείται ως arbiter node ενώ παράλληλα δημιουργούνται arbiter volumes.



Εικόνα 35: Δήλωση των ονομάτων των nodes τα οποία θα φιλοξενήσουν το gluster.

Βήμα 2

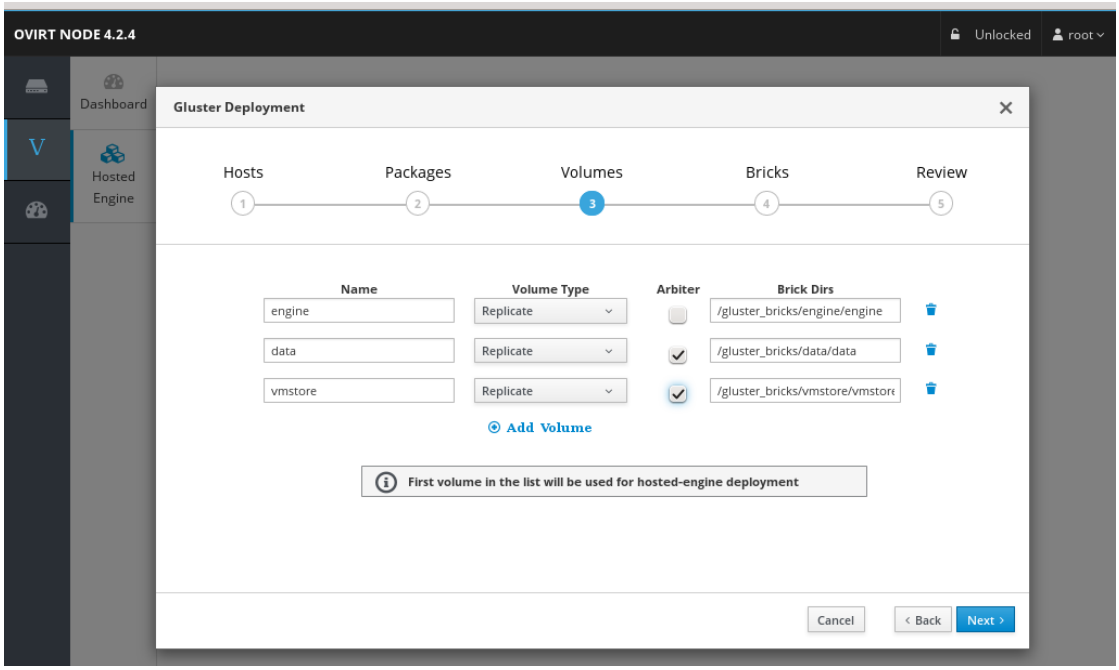
Τώρα βρισκόμαστε στην καρτέλα με τα πακέτα και τα repositories που θέλουμε να κατεβάσουμε για τη δημιουργία του Gluster. Σε αυτή την καρτέλα δεν αλλάζουμε καμία από τις επιλογές και την αφήνουμε ως έχει.



Εικόνα 36: Καρτέλα επιλογής πακέτων για εγκατάσταση στο gluster

Βήμα 3

Σε αυτό το βήμα διαλέγουμε ποια θα είναι τα volumes του Gluster. Σε αυτή τη φάση είναι που κρίνεται απαραίτητη η χρήση του gdeploy, καθώς αυτό θα είναι το εργαλείο με το οποίο θα δημιουργηθούν τα volumes που θα δηλώσουμε όπως και τα μεγέθη τους.

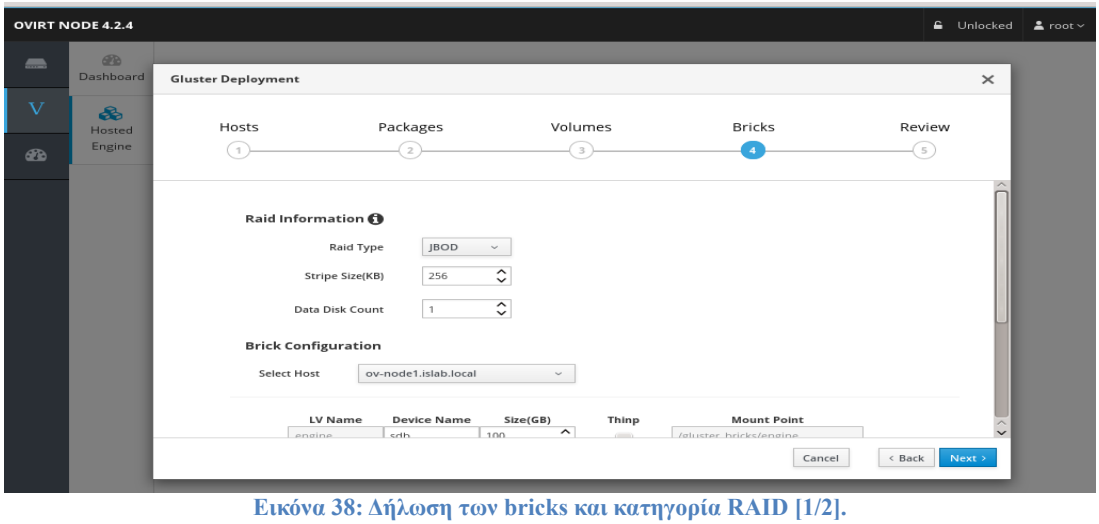


Εικόνα 37: Δήλωση των volumes για το Gluster.

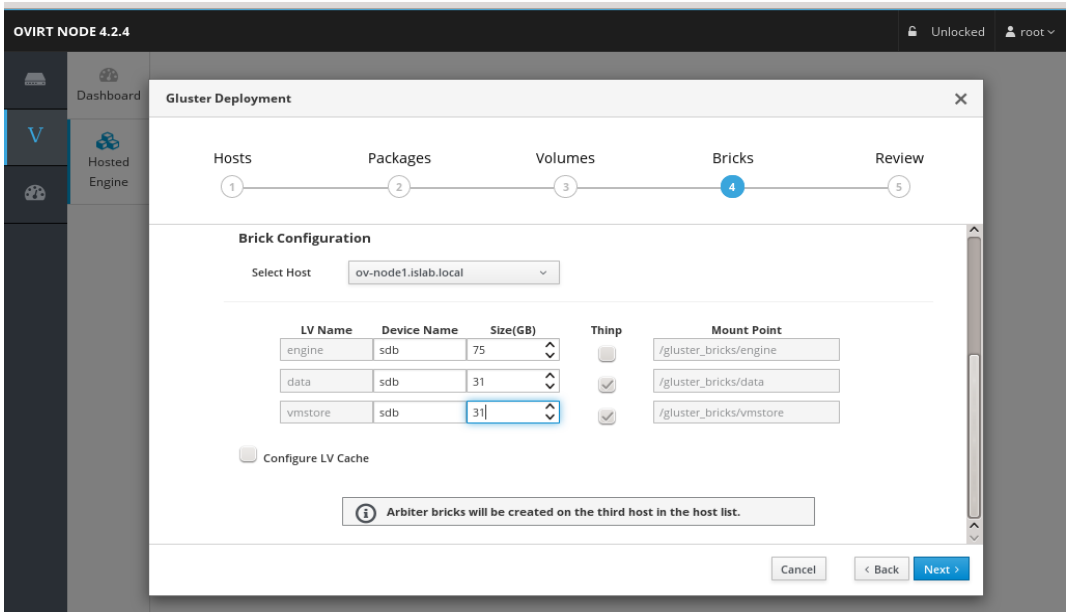
Βήμα 4

Σε αυτό το βήμα πρέπει να ορίσουμε τα bricks που θα αποτελούν το κάθε volume του gluster, αλλά και επιπλέον επιλογές για τους δίσκους. Εδώ μπορούμε να επιλέξουμε ποιας κατηγορίας RAID είναι οι δίσκοι μας αν θέλουμε η εγκατάστασή μας να είναι ακόμα πιο ασφαλής και σταθερή. Για παράδειγμα, αν επιλέξουμε τον τύπο του RAID να είναι **RAID 1,** τότε θα θέλαμε από δύο δίσκους σε κάθε Node οι οποίοι θα ήταν ακριβή αντίγραφα ο ενός του άλλου. Σε συνδυασμό, με το gluster που δημιουργούμε αυτό σημαίνει ότι τα ίδια δεδομένα θα ήταν αποθηκευμένα από δύο φορές σε κάθε server επί τρεις φορές σε κάθε node. Με μία τέτοια επιλογή, τα δεδομένα μας θα ήταν ακόμα πιο ασφαλισμένα και η εμπειρία των χρηστών μας ακόμα πιο ομαλή.

Στα bricks ισχύει η λογική του «ένα brick - για κάθε έναν host - για κάθε ένα volume». Δηλαδή, στον κάθε host έχουμε και από ένα volume το οποίο αποτελείται από τρία ίδια bricks.



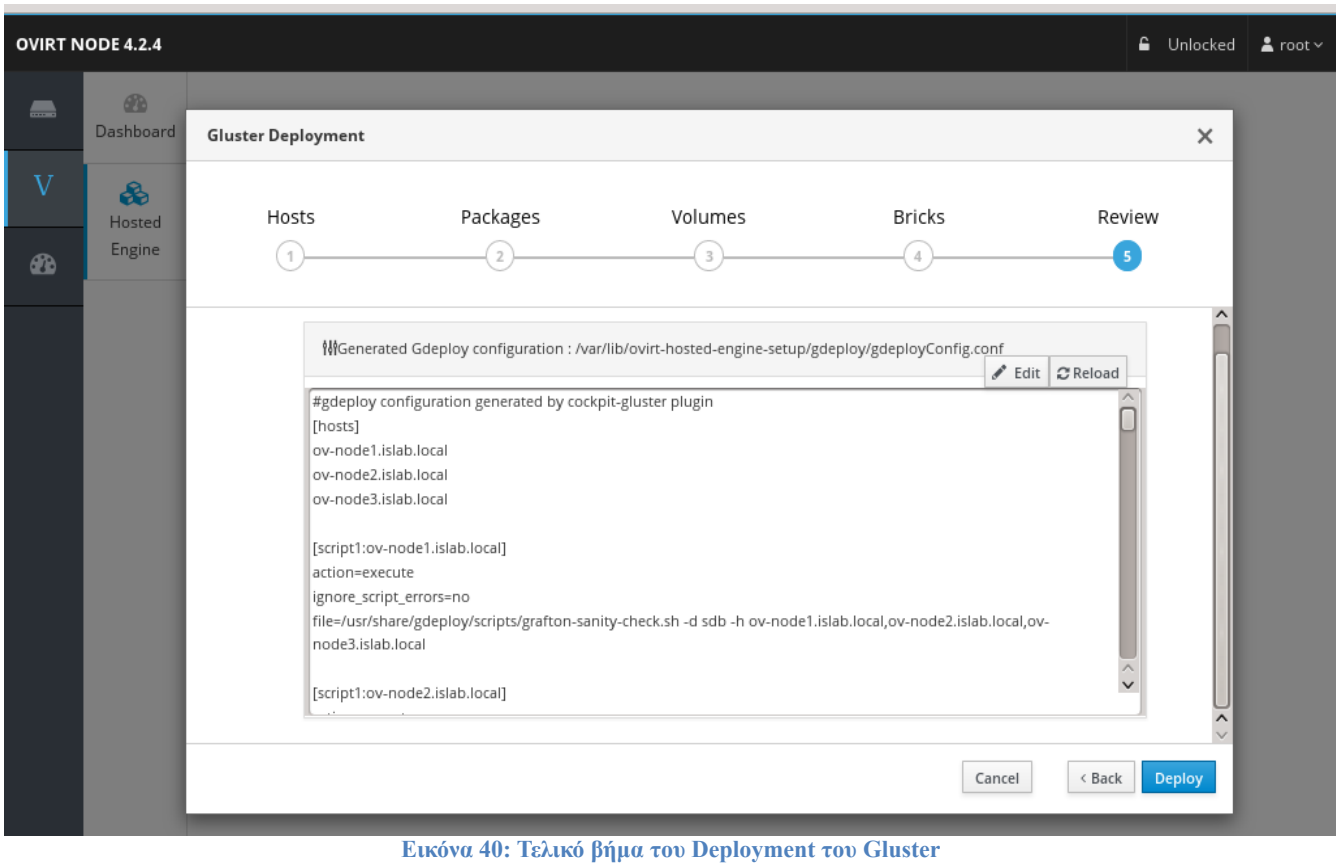
Εικόνα 38: Δήλωση των bricks και κατηγορία RAID [1/2].



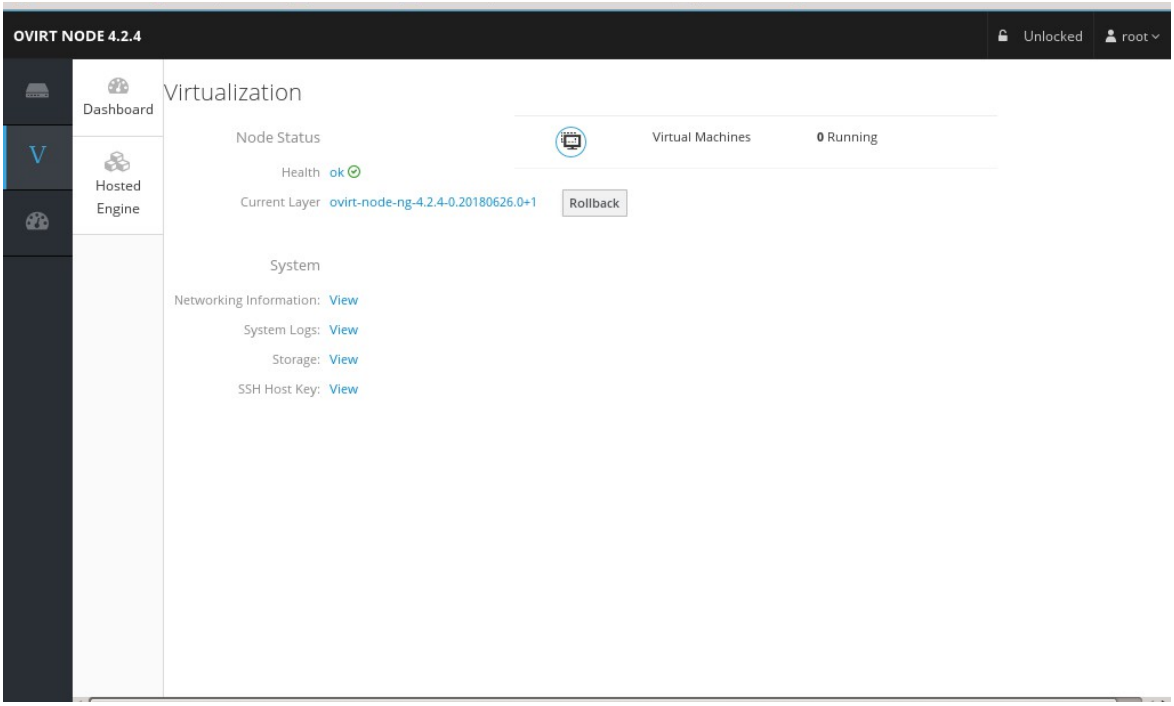
Εικόνα 39: Δήλωση των bricks και κατηγορία RAID [2/2].

Βήμα 5

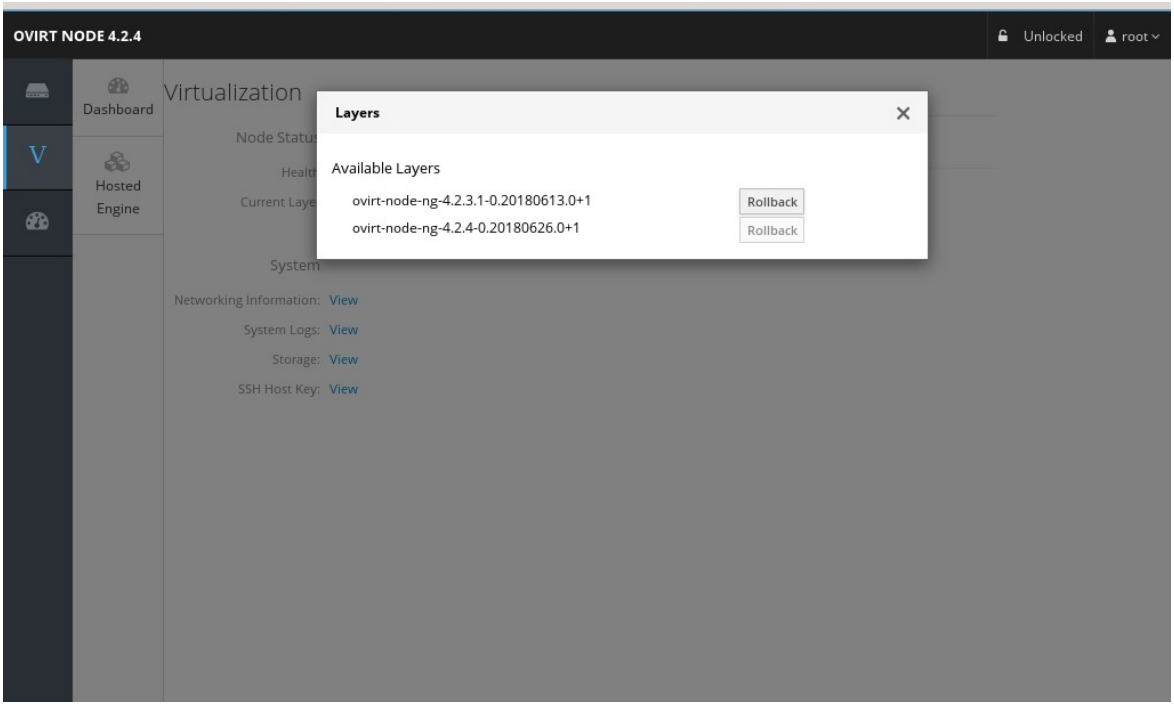
Σύνοψη και παρουσίαση του configuration αρχείου του gdeploy, όπου μπορούμε να επιβλέψουμε τις διαδικασίες οι οποίες θα πραγματοποιηθούν μόλις πατήσουμε το πλήκτρο *Deploy* και ξεκινήσει η διαδικασία δημιουργίας του Gluster στα nodes. Τα scripts που υπάρχουν μέσα στο configuration αρχείο του gdeploy θα εκτελέσουν όλες τις διαδικασίες partition και δημιουργίας των volumes και bricks σε κάθε έναν από τα nodes. [Παράρτημα 1]



Εικόνα 40: Τελικό βήμα του Deployment του Gluster



Εικόνα 41: Αρχική οθόνη του ov-node1



Εικόνα 42: Εγκατεστημένες εκδόσεις ovirt-node στο ov-node1.

Στην παραπάνω εικόνα φαίνονται οι εγκατεστημένες εκδόσεις του συστήματος στο node. Εάν για οποιονδήποτε λόγο επιθυμούμε να επιστρέψουμε σε μία προηγούμενη έκδοση του oVirt-Node, τότε μπορούμε να επιλέξουμε το πλήκτρο *Rollback* από το παραπάνω παράθυρο.

Σημείωση:

Οι εκδόσεις στους nodes μας ΠΡΕΠΕΙ να είναι ομοιογενείς, αλλιώς δεν θα επικοινωνούν σωστά μεταξύ τους και μπορεί να οδηγηθούμε σε αρκετά προβλήματα στις λειτουργίες τους.

2.5.6. oVirt Engine στο oVirt Node

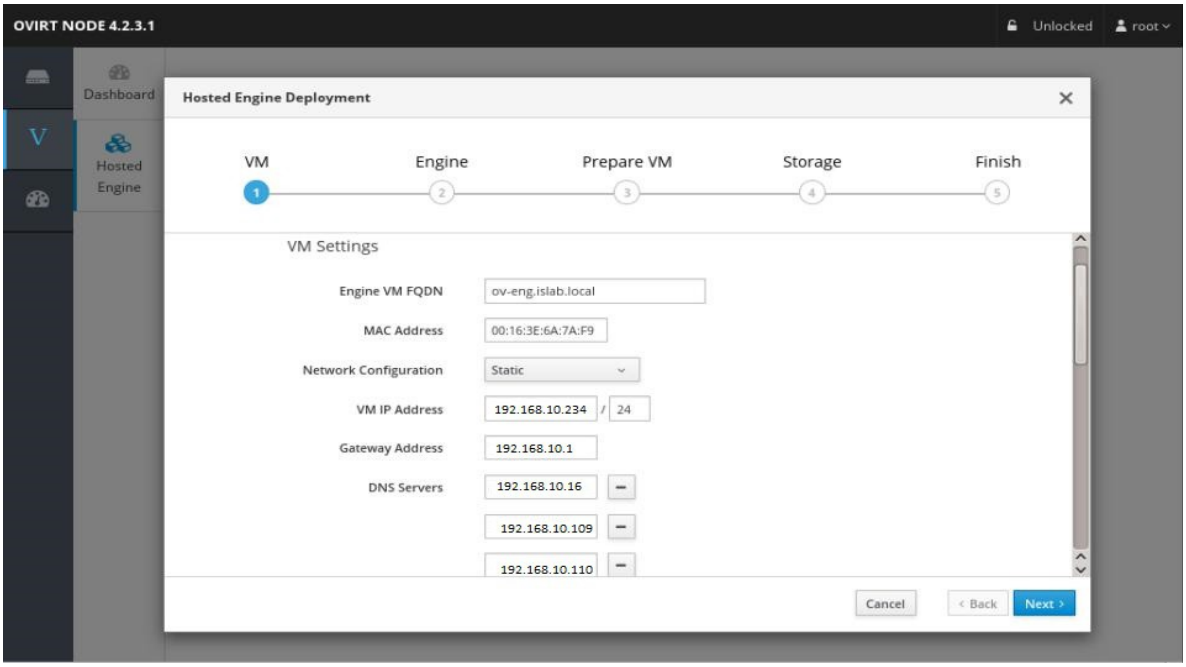
Το oVirt Engine στην περίπτωση των Nodes έχει διαφορετική μορφή από αυτήν που είχε στην προηγούμενη περίπτωση που το εγκαταστήσαμε σε ένα και μόνο μηχάνημα. Εδώ, το oVirt Engine θα δημιουργηθεί ως μία εικονική μηχανή που θα εκτελείται πάνω στο Node, ακόμα και αν ιεραρχικά το oVirt Engine είναι ανώτερο του Node και αυτό που ουσιαστικά καθορίζει τις λειτουργίες του Node. Εφόσον το oVirt Engine δημιουργηθεί ως εικονική μηχανή, σε περίπτωση που ο Node πάνω στον οποίο εκτελείται βγει εκτός λειτουργίας, τότε το Engine μπορεί να μετακινηθεί σε κάποιον άλλον από τους Nodes χωρίς οι χρήστες να καταλάβουν ότι υπήρξε κάποιο πρόβλημα και χωρίς να βιώσουν μία διακοπή στις εργασίες τους.

Η διαδικασία της δημιουργίας του Engine φαίνεται στις παρακάτω εικόνες αναλυτικά.

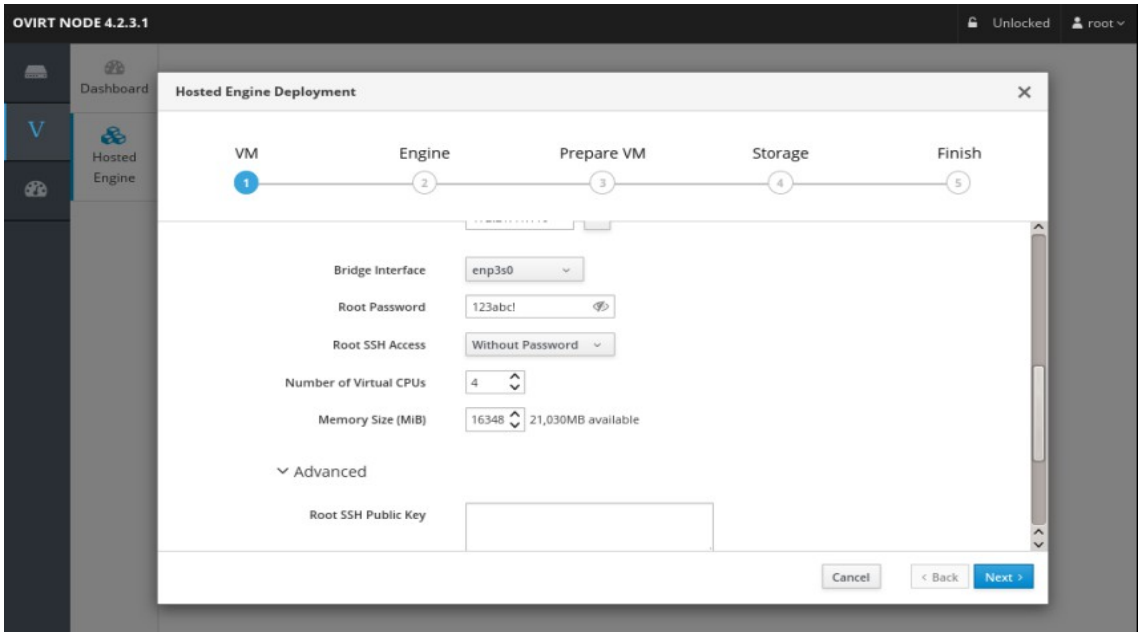
Βήμα 1

Σε αυτό το βήμα ρυθμίζουμε τις παραμέτρους που θα έχει η εικονική μηχανή που θα φιλοξενεί το oVirt Engine μας. Πρέπει να δηλώσουμε ένα όνομα για το domain (Fully Qualified Domain Name - FQDN) της εικονικής μηχανής το οποίο είναι το όνομα που είχαμε φροντίσει από την αρχή της δημιουργίας των Nodes να έχει το oVirt Engine μας (`ov-eng.islab.local` με IP διεύθυνση την 192.168.10.234), ορίζοντάς το στα αρχεία `/etc/hosts` των Nodes ή ορίζοντάς το στον DNS server μας.

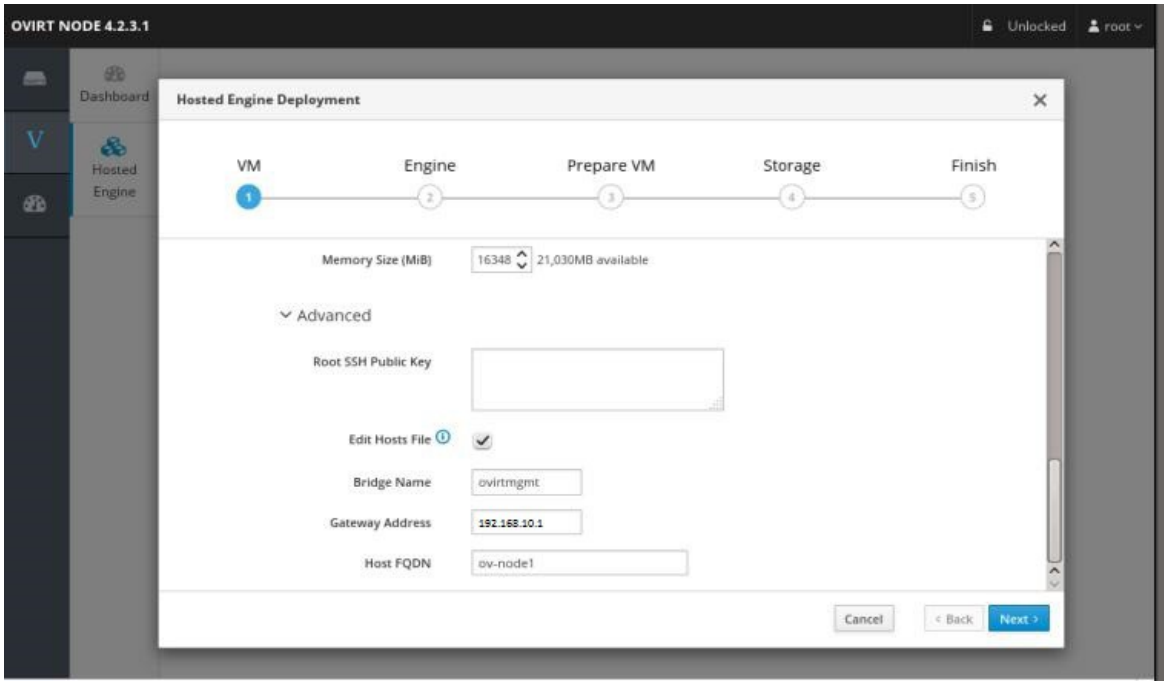
Επίσης, προσφέρουμε όλες τις λεπτομέρειες του δικτύου μας, αφού πρέπει να ορίσουμε τη διεύθυνση IP στατικά, οπότε παραθέτουμε τις διευθύνσεις gateway καθώς και τους DNS servers μας.



Εικόνα 43: Δημιουργία της εικονικής μηχανής που θα φιλοξενεί το Engine [1/3]



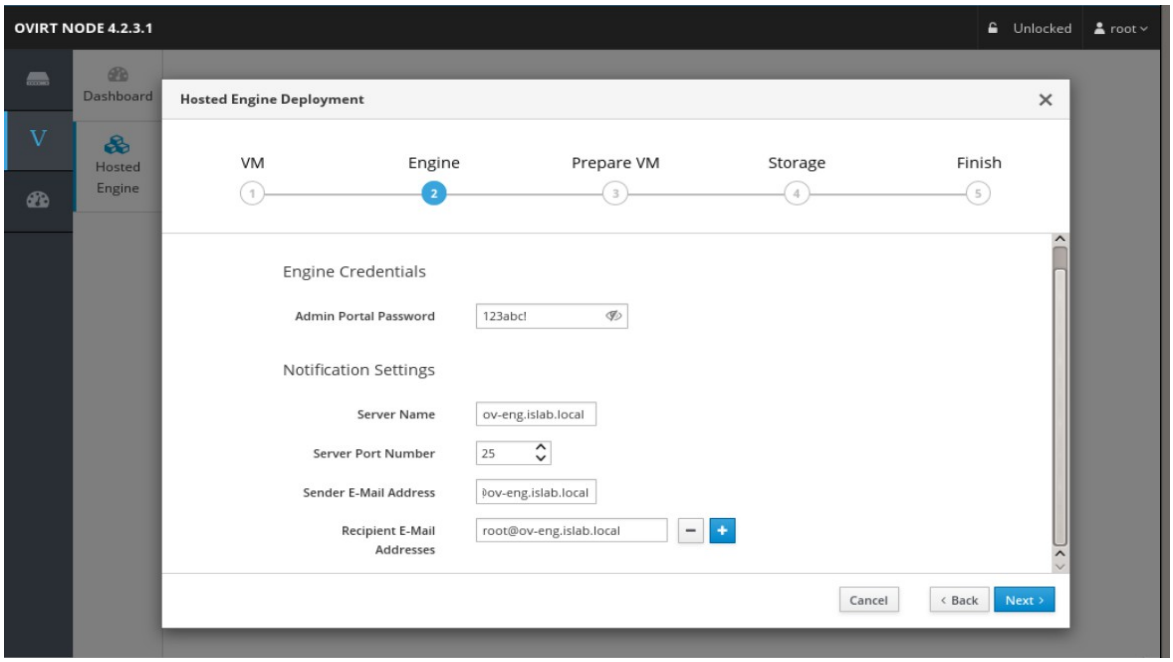
Εικόνα 44: Δημιουργία της εικονικής μηχανής που θα φιλοξενεί το Engine [2/3]



Εικόνα 45: Δημιουργία της εικονικής μηχανής που θα φιλοξενεί το Engine [3/3]

Βήμα 2

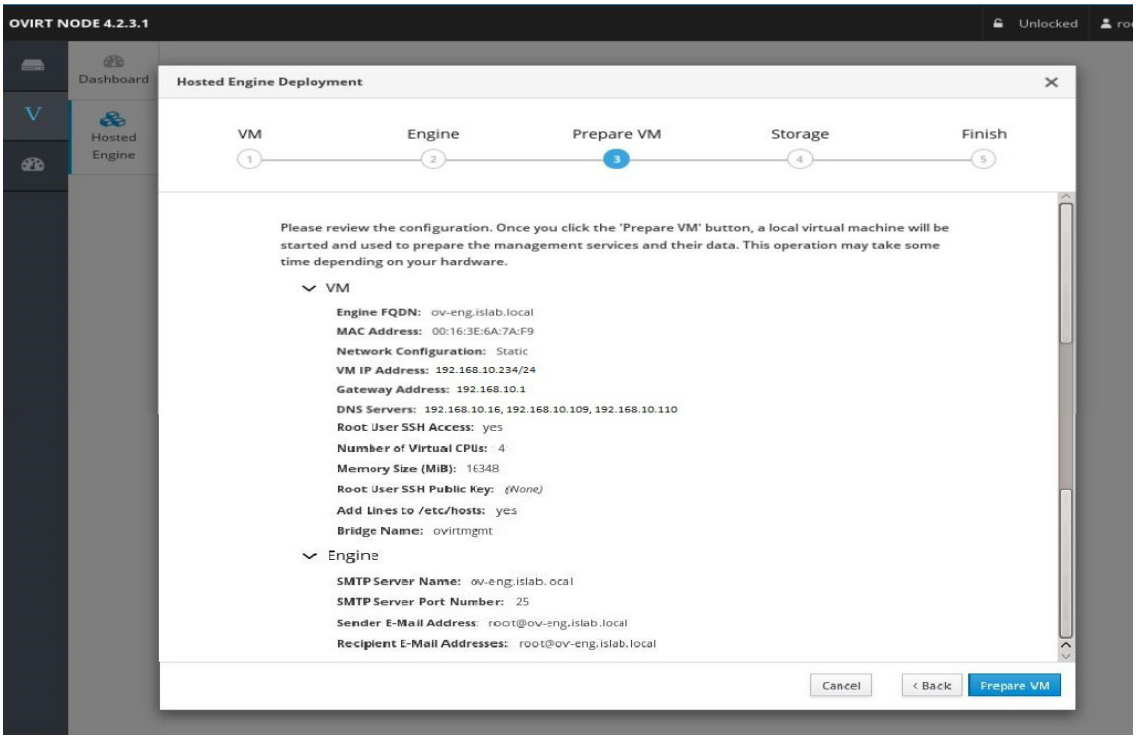
Σε αυτό το βήμα ρυθμίζουμε επιπλέον λεπτομέρειες για το Engine. Εδώ δηλώνουμε ποιο θα είναι το password του root χρήστη. Αυτό το password **πρέπει** να είναι το ίδιο που έχει ο χρήστης root στο φυσικό μηχάνημα. Επίσης δίνουμε το όνομα του Node και ορίζουμε σε ποιο email θέλουμε να αποστέλλονται τα logs και άλλα σημαντικά γεγονότα.



Εικόνα 46: Ρυθμίσεις του Engine

Βήμα 3

Σε αυτό το σημείο έχει δημιουργηθεί μία σύνοψη των επιλογών μας και μπορούμε να την εξετάσουμε συνολικά και στην περίπτωση που κάτι θέλουμε να αλλάξουμε μπορούμε να επιλέξουμε να πάμε πίσω και να αλλάξουμε κάποια παράμετρο. Αλλιώς, επιλέγουμε το *Prepare VM* και αρχίζει η δημιουργία της εικονικής μηχανής του Engine.



Εικόνα 47: Σύνοψη των ρυθμίσεων που κάναμε για το Engine.

2.6. Παρατηρήσεις και Συμπεράσματα από την Εμπειρία μας με το oVirt

- Οι εκδόσεις του oVirt ανανεώνονται σε μηνιαία βάση, γεγονός που μας διευκόλυνε γιατί με την αναβάθμιση σε καινούρια έκδοση λύθηκαν μερικά από τα προβλήματα που αντιμετωπίσαμε.
- Προσφέρει πολλές δυνατότητες, και το γεγονός ότι προσφέρει δυνατότητα να κάνουμε Gluster τους servers μας είναι πολύ σημαντικό και μπορούμε να κάνουμε το σύστημά μας ακόμα πιο ασφαλές και σταθερό, αφού ακόμα και στην περίπτωση που ο ένας από τους τρεις servers βγει εκτός λειτουργίας, υπάρχουν οι υπολειπόμενοι δύο να πάρουν τη θέση του, μην αφήνοντας έτσι τους χρήστες να καταλάβουν οποιαδήποτε αλλαγή στην εμπειρία τους με τα συστήματά μας.
- Είναι πολύ σταθερό σύστημα, εξελίσσεται ταχύτατα και έχει την υποστήριξη της Red Hat, και σκοπό έχει να γίνει ένα προϊόν άξιο να ανταγωνίζεται την Vmware, η οποία είναι και η “βασίλισσα” στα περιβάλλοντα δημιουργίας και διαχείρισης εικονικών συστημάτων.

Κεφάλαιο 3 – KVM

3.1. Περίληψη Κεφαλαίου

Σε αυτό το κεφάλαιο θα ασχοληθούμε με το Linux module KVM το οποίο, παράλληλα με χρήση του QEMU, επιτρέπει στον διαχειριστή να διαχειριστεί επιπλέον εικονικά συστήματα πάνω στο αρχικό του σύστημα. Το QEMU είναι αυτό που προσφέρει τις δυνατότητες που χρειαζόμαστε για τη δημιουργία εικονικών μηχανών και τη διαχείρισή τους σε περιβάλλον Linux.

Θα εγκαταστήσουμε περιβάλλον Debian στο σύστημά μας και έπειτα θα προχωρήσουμε σε εγκατάσταση των πακέτων QEMU και KVM, συμπεριλαμβανομένων των προαπαιτούμενων τους, στο περιβάλλον αυτό. Συνήθως δεν προσφέρονται δυνατότητες γραφικού περιβάλλοντος για τη διαχείριση μέσω της κονσόλας των Linux, αλλά με την εγκατάσταση αυτών των δύο πακέτων, μπορεί να γίνει εφικτή η διαχείρισή τους.

Σχετικά με το πακέτο libvirt το οποίο είναι και να κάνουμε εγκατάσταση καθώς είναι από τα προαπαιτούμενα για να λειτουργήσει σωστά το kvm:

Το libvirt παρέχει ένα ανεξάρτητο API hypervisor με πλήθος δυνατοτήτων εικονικοποίησης για πληθώρα λειτουργικών συστημάτων. Περιέχει ένα virtualization επίπεδο για ασφάλεια και διαχείριση των εικονικών μηχανών σε έναν host. Μπορεί να διαχειρίζεται τοπικούς και απομακρυσμένους hosts. Απαιτούνται ορισμένα API για τη δημιουργία, μετατροπή, παρακολούθηση, έλεγχο, μετανάστευση, εκκίνηση και τερματισμό των εικονικών μηχανών. Το libvirt παρέχει την πρόσβαση σε πολλαπλούς hosts, ενώ τα API περιορίζονται σε απλές εργασίες. Μόνο εργασίες οι οποίες υποστηρίζονται από τον hypervisor μπορούν να εκτελεστούν με τη χρήση του libvirt.

3.2. KVM – Kernel Virtual Manager

Το Kernel-based Virtual Machine (KVM) είναι μια υποδομή virtualization για τον πυρήνα των Linux που έχει την δυνατότητα να το μετατρέπει σε έναν hypervisor. Ήταν συγχωνευμένη με τον πυρήνα των Linux στον έκδοση 2.6.20, η οποία κυκλοφόρησε στις 5 Φεβρουαρίου 2007. Το KVM απαιτεί επεξεργαστή με επεκτάσεις εικονικοποίησης υλικού. Το KVM έχει επίσης μεταφερθεί σε FreeBSD και illumos με τη μορφή φορητών ενοτήτων πυρήνα (kernel modules).

Το KVM (Kernel-based Virtual Machine) αναπτύχθηκε από την εταιρεία Qumranet την οποία και αργότερα αγόρασε η Red Hat, το 2008, οπότε και έκαναν relicense το KVM στο GPL και το έκαναν μέρος του πυρήνα των Linux.

Το KVM είναι hypervisor τύπου-2, δηλαδή εκτελείται σε host λειτουργικό, όπως και το VirtualBox και το Hyper-V τα οποία είναι και αυτά hypervisor τύπου-2. Σε αντίθεση με τους hypervisor Τύπου-1 οι οποίοι εκτελούνται σε bare metal και δεν απαιτούν host λειτουργικά συστήματα (Xen, Vmware ESX).

Το KVM αρχικά υποστήριζε επεξεργαστές x86 και έχει μεταφερθεί σε S/390, PowerPC, και IA-64. Μια θύρα ARM συγχωνεύθηκε κατά τη διάρκεια του παραθύρου συγχώνευσης πυρήνα 3.9.

Μια μεγάλη ποικιλία από λειτουργικά συστήματα guest λειτουργούν με KVM, συμπεριλαμβανομένων πολλών εκδόσεων Linux, BSD, Solaris, Windows, Haiku, ReactOS, Plan 9, AROS (A Research Operating System)

και το OS X. Επιπλέον, τα Android 2.2, GNU / Hurd (Debian K16), Minix 3.1.2a, Solaris 10 U3 και Darwin 8.0.1 μαζί με άλλα λειτουργικά συστήματα και ορισμένες νεότερες εκδόσεις αυτών λειτουργούν συνήθως μαζί με κάποιους περιορισμούς.

Η υποστήριξη Paravirtualization για ορισμένες συσκευές είναι διαθέσιμη για Linux, OpenBSD, FreeBSD, NetBSD, Plan9 αλλά και για χρήστες των Windows που χρησιμοποιούν το API VirtIO. Αυτό το API υποστηρίζει μια paravirtual κάρτα Ethernet, έναν paravirtual ελεγκτή εισόδου / εξόδου δίσκου, μια «συσκευή μπαλονιών» (balloon device) για τη ρύθμιση της χρήσης της μνήμης επισκεπτών και μια διασύνδεση γραφικών VGA που χρησιμοποιεί προγράμματα οδήγησης SPICE ή VMware drivers.

3.2.1. Προαπαιτούμενα

Στα προαπαιτούμενα για την εγκατάσταση του KVM είναι τα εξής:

Υλικό :	• Επεξεργαστής με Virtualization support
Λογισμικό :	• CentOS ή Ubuntu • RHEL 7.+ x86_64 • `Python 2.7.x` • οι τρέχοντες device drivers των εξής πακέτων: ◦ `kmod_be2iscsi_4.6.267.4-1.x86_64` ◦ `kmod_tg3_3.129d-1.x86_64` ◦ `kmod_elx_lpfc_8.3.7.29-1.x86_64` ◦ `kmod_be2net_4.6.267.4-1.x86_64` ◦ `kmod_brocade-bfa_3.2.1.1-0.x86_64` ◦ `kmod_qlgc_qla2xxx_8.04.00.12.06.0_k3-1.x86_64`
Δίκτυο :	Υποστηρίζει μέχρι και 25 interfaces, 24 intefaces για την κίνηση του δικτύου και 1 interface για διαχείριση

3.3. QEMU

Το KVM δουλεύει σε συνεργασία με το QEMU πακέτο.

Το QEMU είναι ένα γενικό και ανοιχτού τύπου (open source) εξομοιωτής μηχανών και virtualizer. Επιτυγχάνει γρήγορες ταχύτητες εξομοίωσης χάρη στη χρήση της δυναμικής μετάφρασης (dynamic translation).

Το QEMU λειτουργεί με δύο τρόπους:

- **Εξομοίωση Ολοκληρωμένου Συστήματος**

όπου το QEMU εξομοιώνει ένα πλήρες υπολογιστικό σύστημα (π.χ. έναν ηλεκτρονικό υπολογιστή), συμπεριλαμβανομένου ενός ή περισσοτέρων επεξεργαστών και των διαφόρων περιφερειακών που έχει ένα ολοκληρωμένο υπολογιστικό σύστημα. Μπορεί να χρησιμοποιηθεί για την εκτέλεση διαφόρων λειτουργικών συστημάτων χωρίς να χρειάζεται η επανεκκίνηση του συστήματος του υπολογιστή ή να γίνει debug ο κώδικας του συστήματος.

- **Εξομοίωση Περιβάλλοντος Χρήστη**

Το QEMU μπορεί να εκτελέσει διεργασίες οι οποίες έχουν αναπτυχθεί πάνω σε έναν συγκεκριμένου τύπου επεξεργαστή, σε κάποιον άλλον και η εκτέλεση της διεργασίας να είναι η πρέπουσα. Για παράδειγμα, μπορεί να χρησιμοποιηθεί για να εκτελεστεί το API Wine Windows ή για να διευκολυνθεί η δια-σύνταξη και δια-μεταγλώττιση καθώς και η επιδιόρθωση του κώδικα πάνω σε διαφορετικές πλατφόρμες.

3.3.1. Χαρακτηριστικά

Το QEMU μπορεί να εκτελείται χωρίς την υποστήριξη κάποιου driver του πυρήνα στο σύστημα και παρόλα αυτά να έχει αποδεκτή απόδοση. Χρησιμοποιεί δυναμική μετάφραση στον ιθαγενή κώδικα του συστήματος για να έχει καλή ταχύτητα, μαζί με την υποστήριξη για αυτό-διαμορφωμένο κώδικα και ακριβείς εξαιρέσεις.

Υποστηρίζει φορητότητα σε πλειάδα λειτουργικών συστημάτων (GNU/Linux, *BSD, MacOSX, Windows) και αρχιτεκτονικών.

Αποδίδει ακριβή λογισμική εξομοίωση της FPU.

Συγκεκριμένα, ανάλογα με τον τρόπο εκτέλεσης του QEMU, τα χαρακτηριστικά του QEMU διαμορφώνονται ως εξής:

- **Εξομοίωση Ολοκληρωμένου Συστήματος**
    - Χρήση του πλήρες λογισμικού MMU για επίτευξη μέγιστης φορητότητας.
    - Προαιρετική χρήση ενός συγχωνευμένου με τον πυρήνα των Linux επιταχυντή, όπως είναι το KVM. Οι επιταχυντές αυτοί εκτελούν τον περισσότερο από τον κώδικα του guest συστήματος τοπικά, ενώ ταυτόχρονα συνεχίζει η εξομοίωση του υπόλοιπου συστήματος.
    - Συσκευές υλικού καθώς και συσκευές του host συστήματος (π.χ. σειριακές και παράλληλες θύρες) μπορούν να εξομοιωθούν και να χρησιμοποιηθούν από τα λειτουργικά συστήματα των guest συστημάτων. Η μεταβίβαση των συσκευών μπορεί να χρησιμοποιηθεί για επικοινωνία με εξωτερικές φυσικές συσκευές (π.χ. εκτυπωτής, webcam κ.λπ.)
    - Υποστήριξη συμμετρικής πολύ-επεξεργασίας (Symmetric Multi-Processing – SMP). Μέχρι και σήμερα, χρειάζεται η χρήση ενός συγχωνευμένου με τον πυρήνα των Linux επιταχυντή, όπως είναι το KVM, για να μπορεί να χρησιμοποιεί περισσότερες από μία CPU των host συστημάτων για εξομοίωση.
- **Εξομοίωση Περιβάλλοντος Χρήστη**
    - Γενικές κλήσεις του converter του συστήματος των Linux συμπεριλαμβανομένων των περισσοτέρων ioctls.
    - Μπορεί να κάνει εξομοίωση της συνάρτησης clone() του συστήματος δημιουργώντας μία δική του clone() η οποία θα έχει τη δυνατότητα να χρησιμοποιεί τον προγραμματιστή των χρονοδιαγραμμάτων των νημάτων του συστήματος.
    - Ακριβής μεταχείριση των σημάτων μέσω της μετάφρασης των σημάτων του host συστήματος σε σήματα συστήματος στόχου.

3.4. KVM & QEMU

Το KVM είναι ένα module του πυρήνα των Linux, ενώ το QEMU είναι ένα λογισμικό εξομοίωσης συστημάτων.

Μέσω του KVM , το QEMU μπορεί να εξομοιώνει τα εικονικοποιημένα περιβάλλοντα.

Το QEMU σε σχέση με το KVM, είναι πιο αργό, πιθανόν επειδή το KVM είναι συγχωνευμένο με τον πυρήνα των Linux και μπορεί να κάνει καλύτερη διαχείριση των πόρων του συστήματος. Το qemu_kvm είναι ένα QEMU project που αναπτύχθηκε και με την έκδοση του QEMU 1.3, όλα του τα χαρακτηριστικά και λειτουργίες έχουν συγχωνευτεί και διατίθενται πλέον μαζί με την κεντρική έκδοση του QEMU.

Το KVM είναι μία πλήρης λύση εικονικοποίησης για τα Linux σε υλικό x86 που παρέχει επεκτάσεις virtualization. Το KVM αποτελείται από τα:

- kvm.ko : το οποίο είναι ένα loadable module του πυρήνα που παρέχει την κεντρική υποδομή για την εικονικοποίηση του συστήματος.
- kvm-inte.ko ή kvm-amd.ko: τα οποία είναι modules για τους συγκεκριμένους επεξεργαστές.

Με την έκδοση του QEMU 1.3 και έπειτα, το userspace του KVM περιλαμβάνεται πλέον στο πακέτο του QEMU.

3.4.1. Προαπαιτούμενα

- `qemu-kvm_release.tar.gz`
- `kvm-kmod_release.tar.gz`, εάν χρειάζεται να κάνουμε compile κάποια modules του πυρήνα.
- Intel επεξεργαστή με δυνατότητες VT ή AMD επεξεργαστή με δυνατότητες SVM (δυνατότητες virtualization)

Πιο συγκεκριμένα για το QEMU, χρειαζόμαστε τις εξής βιβλιοθήκες και headers:

- `zib`
- SDL
- `alsa` (αρχικά η υποστήριξη για το alsa είναι απενεργοποιημένη, αν επιθυμούμε όμως μπορούμε να την ενεργοποιήσουμε με χρήση της παραμέτρου `--enable-alsa`)
- `gnutils`
- kernel headers (συγκεκριμένα για τη fedora τα πακέτα kernel-devel)

Στα Debian συστήματα μπορούμε να εγκαταστήσουμε όλα τα προαπαιτούμενα με την εντολή:

```
# apt-get install gcc libsdl1.2-dev zlib1g-dev libasound2-dev linux-
  kernel-headers pkg-config libgnutils-dev libpci-dev
```

Εάν δουλεύουμε από το git, θα χρειαστεί να προσθέσουμε και το πακέτο gawk.

3.5. Debian

3.5.1. Εγκατάσταση

Εγκαθιστούμε το λειτουργικό περιβάλλον Debian 9 στο σύστημά μας, το οποίο βρίσκουμε την πρόσφατη και σταθερή έκδοσή του στην ιστοσελίδα https://www.debian.org/releases/stable/amd64/ch03S01.html ή στην σελίδα https://www.debian.org/CD/http-ftp/#stable.

Ακολουθούμε παρόμοια βήματα με αυτά της εγκατάστασης των CentOS όπως αναφέρονται στο προηγούμενο κεφάλαιο και στη συνέχεια ρυθμίζουμε το σύστημά μας ώστε να είναι έτοιμο να δεχθεί και να εκτελέσει καθώς πρέπει τις λειτουργίες virtualization που επιθυμούμε.

Μετά το πέρας της εγκατάστασης του λειτουργικού, φροντίζουμε λοιπόν να προσθέσουμε τα repositories contrlib και non-free, καθώς και να ενεργοποιήσουμε τις πιο πρόσφατες ενημερώσεις ασφαλείας του συστήματος. Εν συνεχεία, εγκαθιστούμε τα προαπαιτούμενα για τις λειτουργίες virtualization των KVM και QEMU με την εντολή:

```
# apt -y install qemu-kvm libvirt-daemon libvirt-daemon-system virtinst
  bridge-utils
```

3.5.2. Λίγα Λόγια για το Σύστημα Debian

Μπορούμε να βρούμε όποιες πληροφορίες μας ενδιαφέρουν για το σύστημα Debian στα κανάλια IRC με χρήση των hashtags #debian και #debian-boot στο OFTC δίκτυο.

Ειδικοί φάκελοι εγκαθίστανται στον φάκελο /ust/share/doc.

Αρχεία με οδηγίες για τα Linux βρίσκονται στο μονοπάτι /usr/share/doc/HOWTO/en-txt σε μορφή .gz.

Με την εγκατάσταση και του dhelp μπορούμε να βρούμε κατάλογο με σελιδοποιημένο υλικό για το documentation των Debian στο μονοπάτι /usr/share/doc/HTML/index.html. Ένας εύκολος τρόπος για να μπορέσουμε να δούμε αυτά τα αρχεία, είτε έχουμε εγκαταστήσει το Debian με GUI (Graphical User Interface) ή χωρίς, είναι με χρήση των εντολών:

```
#  cd /usr/share/doc
#  w3m .
```

Όπου η εντολή w3m αφορά σε έναν text based πλοηγητή και η τελεία "." αφορά στο μονοπάτι στο οποίο βρισκόμαστε, δηλαδή στο /usr/share/doc.

Ή εάν το Debian που έχουμε εγκαταστήσει έχει GUI, μπορούμε μέσω ενός προγράμματος πλοήγησης στο Διαδίκτυο, να τοποθετήσουμε στο πεδίο URL το μονοπάτι /usr/share/doc και να πλοηγηθούμε στα περιεχόμενα του φακέλου.

Για να έχουμε τα αρχεία με τις οδηγίες του συστήματος μπορούμε να εγκαταστήσουμε τα πακέτα με τα documentations εγκαθιστώντας τα πακέτα doc-linux-html με το documentation σε μορφή HTML και το doc-linux-ascii με το documentation σε μορφή ASCII.

Επίσης με χρήση μίας των κάτωθι εντολών, μπορούμε να έχουμε άμεση πρόσβαση στο documentation οποιασδήποτε εντολής μας ενδιαφέρει:

```
#  info <command>
#  man <command>
#  <command> --help | more
```

3.5.2.1. Το Σύστημα Πακέτων στο Debian

Μεγάλα μέρη του συστήματός μας βρίσκονται υπό τον έλεγχο του συστήματος πακέτων (packaging system) του Debian:

- /usr, εκτός του /usr/local
- /var, όπου μπορούμε να δημιουργήσουμε τον φάκελο local και να είναι ασφαλή τα αρχεία του φακέλου αυτού (/var/local)
- /bin
- /sbin
- /lib

Για παράδειγμα, μπορούμε να αντικαταστήσουμε την έκδοση της perl που έχουμε και βρίσκεται στο μονοπάτι /usr/bin/perl. Εάν όμως αντικαταστήσουμε την έκδοση κάποιου πακέτου με κάποια πρωτύτερη έκδοση, τότε της επόμενη φορά που θα κάνουμε αναβάθμιση στα πακέτα μας, τότε και αυτό το πακέτο θα αναβαθμιστεί και δεν θα έχουμε πλέον την επιλεγμένη μας έκδοση. Για να αποφύγουμε κάτι τέτοιο,

μπορούμε να τοποθετήσουμε το συγκεκριμένο πακέτο ή πακέτα που επιθυμούμε να παραμείνουν σε κάποια συγκεκριμένη έκδοση υπό αναμονή στο “hold” στο aptitude, με το οποίο και διαχειριζόμαστε όλα μας τα πακέτα.

Το apt είναι μία από τις καλύτερες μεθόδους εγκατάστασης πακέτων στο σύστημά μας. Υπάρχει έκδοση του apt για την γραμμή εντολών των Linux (`apt-get install`), καθώς και το aptitude που είναι μία έκδοση πλήρους οθόνης. Με το apt μπορούμε να συνδυάσουμε τα repositories των πακέτων `main`, `contrib` και `non-free` ώστε να έχουμε περιορισμένες (`export-restricted`) αλλά και τις `standard` εκδόσεις.

Επιπλέον λογισμικό για το Debian μπορούμε να βρούμε στην ιστοσελίδα http://wiki.debian.org/DebianSoftware. Επιπλέον πληροφορίες για τη διαχείριση των εκδόσεων των πακέτων μπορούμε να βρούμε με την εντολή `man update-alternatives.`

3.5.2.2. Διαχείριση Εργασιών μέσω του CRON

Οποιεσδήποτε εργασίες του διαχειριστή του συστήματος πρέπει να βρίσκονται στον φάκελο `/etc`, αφού είναι αρχεία configuration.

Εάν έχουμε εργασίες cron του root χρήστη που πρέπει να επαναλαμβάνονται σε καθημερινή, εβδομαδιαία ή μηνιαία βάση, τις τοποθετούμε στο κατάλληλο αρχείο στο `/etc/cron.{daily,weekly,monthly}`. Τα αρχεία αυτά καλούνται από το `/etc/crontab` και εκτελούνται με αλφαβητική σειρά, σειριακά.

Εάν έχουμε εργασίες cron οι οποίες πρέπει να εκτελούνται υπό κάποιον άλλον λογαριασμό χρήστη και όχι αυτόν του root ή πρέπει να εκτελούνται κάποια συγκεκριμένη στιγμή ή με κάποια άλλη συχνότητα, χρησιμοποιούμε το `/etc/crontab` ή το `/etc/cron.d/whatever`. Αυτά τα συγκεκριμένα αρχεία περιέχουν ένα επιπλέον πεδίο το οποίο μας επιτρέπει να ορίσουμε τον λογαριασμό του χρήστη υπό τον οποίο πρέπει να εκτελεστεί η εργασία cron.

Όποια και αν είναι η περίπτωση που μας απασχολεί, μπορούμε απλά να επιμεληθούμε το κατάλληλο αρχείο και από εκεί και έπειτα, το `cron` θα προσέξει τις αλλαγές που έχουμε κάνει και θα εκτελέσει τις εργασίες αναλόγως, χωρίς να χρειαστεί εμείς να δώσουμε κάποια επιπλέον εντολή.

Περισσότερες πληροφορίες σχετικά με το `cron` μπορούμε να βρούμε στις σελίδες `man` για το `cron(8)` και το `crontab(5)` και στο αρχείο `/usr/share/doc/cron/README.Debian`.

Περισσότερες πληροφορίες σχετικά με το Debian μπορούμε να βρούμε να βρούμε στις παρακάτω ιστοσελίδες:

https://www.debian.org/

https://www.debian.org/doc/FAQ

https://www.debian.org/doc/user-manuals#quick-reference

https://www.debian.org/doc/ddp

https://lists.debian.org

https://www.tldp.org

3.6. KVM

Κάνουμε την εγκατάσταση του KVM στο σύστημά μας μέσω της γραμμής εντολών, την κονσόλα. Ωστόσο, τις εικονικές μηχανές που θέλουμε να δημιουργήσουμε, μπορούμε να τις φτιάξουμε με έναν από τους δύο παρακάτω τρόπους:

1. Μέσω της γραμμής εντολών, ή
2. Μέσω του γραφικού περιβάλλοντος virt-manager.

3.6.1. 1ος τρόπος – Κονσόλα, text-based configuration

3.6.1.1. Αποσυμπίεση και Ρύθμιση του KVM

Κάνουμε αποσυμπίεση του πακέτου qemu-kvm που έχουμε κατεβάσει και προχωράμε στην εγκατάστασή του ως εξής:

```
#  tar xzf qemu-kvm-release.tar.gz
#  cd qemu-kvm-release
#  ./configure –prefix=/usr/local/kvm
#  make
#  sudo make install
#  sudo /sbin/modprobe kvm-intel
```

ή

```
#  sudo /sbin/modprobe kvm-amd
```

αναλόγως με την αρχιτεκτονική του επεξεργαστή μας.

3.6.1.2. Δημιουργία Αρχείου .image για τον Guest Χρήστη

Μπορούμε να δημιουργήσουμε ένα αρχείο .image για τον guest χρήστη μέσω της κονσόλας με την εντολή:

```
#  /usr/local/kvm/bin/qemu -img create -f qcow2 vdisk.img 10G
```

με την οποία δημιουργούμε ένα αρχείο .img το οποίο έχει το ρόλο του σκληρού δίσκου του εικονικού συστήματος το οποίο και θα δημιουργήσουμε στην συνέχεια. Στην περίπτωσή μας δημιουργήσαμε έναν "σκληρό δίσκο" μεγέθους 10G στον οποίο και θα εγκατασταθεί το λειτουργικό σύστημα της εικονικής μηχανής και θα παραμείνει και μερικός χώρος για αποθήκευση δεδομένων και προγραμμάτων. Με 10G χώρο, μπορούμε να εγκαταστήσουμε αρκετά από τα Linux λειτουργικά, αλλά όχι λειτουργικό Windows το οποίο απαιτεί περισσότερο αποθηκευτικό χώρο.

3.6.1.3. Εγκατάσταση Λειτουργικού Συστήματος στο Guest Σύστημα

Μέσω της κονσόλας, συνεχίζουμε και προχωράμε στην εγκατάσταση του λειτουργικού συστήματος που έχουμε επιλέξει για την εικονική μας μηχανή με την παρακάτω εντολή:

```
#  sudo   /usr/local/kvm/bin/qemu-system-x86_64   -hda   vdisk.img   -cdrom
   /path/to/boot-media.iso -boot d -m 384
```

με την οποία δηλώνουμε ότι θέλουμε να εγκαταστήσουμε 64-bit λειτουργικό σύστημα με μνήμη 384MB στον σκληρό δίσκο hda ο οποίος είναι ο "δίσκος" που δημιουργήσαμε με την προηγούμενη εντολή. Δίνουμε το μονοπάτι του συστήματος στο οποίο βρίσκεται αποθηκευμένο το bootable .iso αρχείο το οποίο και θα χρησιμοποιήσουμε για να γίνει η εγκατάσταση.

3.6.1.4. Εκτέλεση του Guest Συστήματος

Μέσω της κονσόλας, μπορούμε να εκτελέσουμε το guest σύστημα με δύο τρόπους. Ο ένας είναι και ο πιο απλός με απλή δήλωση του disk image όπου βρίσκεται το εικονικό σύστημα. Ο δεύτερος τρόπος περιέχει αρκετές παραμέτρους με τις οποίες μπορούμε να συγκεκριμενοποιήσουμε λεπτομέρειες του συστήματος όπως το δίκτυο, αν θέλουμε να πάρουμε στιγμιότυπο του εικονικού συστήματος, ενεργή υποστήριξη ή όχι για usb κ.λπ.

Ο απλός τρόπος είναι:

```
#  sudo /usr/local/kvm/bin/qemu-system-x86_64 vdisk.img -m 384
```

Και ο πιο περίπλοκος τρόπος είναι:

```
# sudo /usr/local/kvm/bin/qemu-system-x86_64 -hda vdisk.img -m 384
  -soundhw es1370 -no-acpi -snapshot -localtime -boot c -usb -usbdevice
  tablet -net nic,vlan=0,macaddr=00:00:10:52:37:48 -net
  tap,vlan=0,ifname=tap0,script=no
```

3.6.2. 2ος τρόπος – virt-manager

3.6.2.1. Εγκατάσταση Προαπαιτούμενων

Εγκαθιστούμε τα προαπαιτούμενα για τα KVM και QEMU με την εντολή:

```
# apt -y install qemu-kvm libvirt-daemon libvirt-daemon-system virtinst
  bridge-utils virt-manager
```

Το virt-manager είναι ένα πρόγραμμα για εύκολη διαχείριση των εικονικών μηχανών με γραφικό περιβάλλον.

3.6.2.2. Ρύθμιση Δικτύου

Εκτελούμε την παρακάτω εντολή και μόλις μας ζητηθεί δίνουμε τον κωδικό του root χρήστη, ώστε να μπορέσουμε να συνεχίσουμε με την εργασία μας, χωρίς να χρειάζεται να προσθέτουμε μπροστά από κάθε εντολή το sudo.

```
# su -
# /Password:
```

Εκτελούμε τις παρακάτω εντολές για να βρούμε ποιο είναι το εικονικό δίκτυό μας, το όνομα του interface δικτύου μας (enp0s25 στην περίπτωσή μας), ώστε να μπορέσουμε αργότερα να αλλάξουμε τις ρυθμίσεις που είναι ορισμένες στο αρχείο `/etc/network/interfaces` και να μπορέσουμε να δημιουργήσουμε ένα bridged δίκτυο το οποίο θα μπορούν να χρησιμοποιούν και οι εικονικές μηχανές μας.

```
# modprobe vhost_net
# lsmod | grep vhost
# ip addr show
# pico /etc/network/interfaces
```

Τα περιεχόμενα του αρχείου `/etc/network/interfaces` διαμορφώνεται ως εξής:

```
#This file contains the network interfaces
#For more information, see interfaces(5)
source /etc/network/interfaces.d/*

#The loopback network interface
auto lo
iface lo inet loopback

#The primary network interface
auto enp0s25
#iface enp0s25 inet static
#address 192.168.10.145
#network 192.168.10.0
#netmask 255.255.255.0
#broadcast 192.168.10.255
#gateway 192.168.10.1
#dns-nameservers 192.168.10.16

#add bridge interface
iface br0 inet static
address 192.168.10.145
network 192.168.10.0
netmask 255.255.255.0
broadcast 192.168.10.255
gateway 192.168.10.1
dns-nameservers 192.168.10.16
bridge_ports enp0s25
bridge_stp off
auto br0
```

Αποθηκεύουμε το αρχείο και προχωράμε σε επανεκκίνηση του συστήματος ώστε να ενημερωθεί για τις αλλαγές που κάναμε.

3.6.2.3. Δημιουργία Εικονικής Μηχανής μέσω CLI

Αρχικά εγκαθιστούμε τα παρακάτω πακέτα και προχωράμε σε update του νέου μας συστήματος Debian.

```
#  apt -y install libosinfo-bin libguestfs-tools virt-top
#  apt update ή aptitude update ή aptitude full-upgrade
```

Μπορούμε να προσθέσουμε κάποιο από τα repositories πακέτων με την εντολή:

```
#  add-apt-repository
```

Δημιουργούμε “πισίνα” αποθηκευτικού χώρου. Αρχικά, δημιουργούμε το μονοπάτι `/var/kvm/images` με την εντολή:

```
#  mkdir -p /var/kvm/images
```

Εγκαθιστούμε την εικονική μηχανή που ονομάζουμε ”vdebian”. Δηλώνουμε ότι η εικονική μηχανή θα έχει 2 επεξεργαστές, 4GB μνήμη RAM, τύπο λειτουργικού συστήματος Linux και έκδοση Debian 9. Επίσης, ότι θα δεν έχει γραφικό περιβάλλον, αλλά θα είναι text-based. Τέλος, δηλώνουμε την ιστοσελίδα μέσω της οποίας θα κατεβάσουμε το λειτουργικό σύστημα προς εγκατάσταση στην εικονική μηχανή.

```
#  virt-install \
#  --name vdebian \
#  --ram 4096 \
#  --disk path=/var/kvm/images/template.img,size=30 \
#  --vcpus 2 \
#  --os-type linux \
#  --os-variant debian9 \
```

```
#  --network bridge=br0 \
#  --graphics none \
#  --console pty,target_type=serial \
#  --location
   'http://ftp.ntua.gr/pub/linux/debian/dists/stretch/main/installer-
   amd64/' \
#  --exrta-args 'console=ttyS0,115200n8 serial'
```

Με το πέρας της εγκατάστασης της εικονικής μηχανής, τερματίζουμε την λειτουργία του guest συστήματος που φτιάξαμε με την εντολή:

```
#  virsh shutdown vdebian
```

Κάνουμε mount τον δίσκο του guest και ενεργοποιούμε υπηρεσίες με τον κάτωθι τρόπο:

```
#  guestmount -d vdebian -i /mnt
#  ln -s /mnt/lib/systemd/system/getty@.service
   /mnt/etc/systemd/system/getty.target.wants/getty@ttyS0.service
#  umount /mnt
```

Εκκινούμε το guest σύστημα ως εξής:

```
#  virsh start vdebian –-console
```

και πλέον συνδεόμαστε στην κονσόλα του guest συστήματος. Για να μετακινηθούμε από την κονσόλα του guest στην κονσόλα του φυσικού μας συστήματος, πατάμε ταυτόχρονα Ctrl +]. Για να μετακινηθούμε από την κονσόλα του φυσικού μας συστήματος στην κονσόλα του guest συστήματος, δίνουμε στην κονσόλα μας την παρακάτω εντολή:

```
#  virsh console vdebian
```

Για να κλωνοποιήσουμε μία εικονική μηχανή, μπορούμε να δώσουμε την εξής εντολή στην κονσόλα μας:

```
# virt-clone --original vdebian --name debian9 --file
  /var/kvm/images/debian9.img

# ll /var/kvm/images/debian9.img

# ll /etc/libvirt/qemu/debian9.xml
```

Με τις εντολές “ll” μπορούμε να επαληθεύσουμε τη δημιουργία των παραπάνω αρχείων.

3.6.2.3.1. Βασικές Λειτουργείες με την εντολή virsh

Παρακάτω παρατίθενται μερικές από τις πιο σημαντικές λειτουργίες που μπορούμε να εκτελέσουμε με την εντολή virsh:

1. Εκκίνηση εικονικής μηχανής που ονομάζεται test1:

```
# virsh start test1
```

2. Εκκίνηση εικονικής μηχανής που ονομάζεται test1 και είσοδος στο text-based περιβάλλον της:

```
# virsh start test1 --console
```

3. Τερματισμός λειτουργίας εικονικής μηχανής:

```
# virsh shutdown test1
```

4. Εξαναγκαστικός τερματισμός λειτουργίας εικονικής μηχανής:

```
# virsh destroy test1
```

5. Εμφάνιση όλων των ενεργών εικονικών μηχανών:

```
# virsh list
```

6. Εμφάνιση όλων των εικονικών μηχανών, ενεργών και μη:

```
#  virsh list --all
```

7. Είσοδος στην κονσόλα της εικονικής μηχανής με όνομα test1:

```
#  virsh console test1
```

8. Έξοδος από την κονσόλα της εικονικής μηχανής με όνομα test1:

```
#  Ctrl + ]
```

9. Περισσότερες πληροφορίες και βοήθεια σχετικά με την εντολή `virsh`:

```
#  virsh --help
```

3.6.2.4. Δημιουργία Εικονικής Μηχανής μέσω του virt-manager

Αρχικά πρέπει να εγκαταστήσουμε μερικά επιπλέον πακέτα:

```
#  apt -y install libguestfs-tools virt-top
```

Η εντολή `virt-top` μας επιτρέπει να δούμε όλες τις εικονικές μηχανές που έχουμε δημιουργήσει, αλλά και να επιβλέψουμε την κατάσταση στην οποία βρίσκεται η κάθε εικονική μηχανή.

Με αυτά τα πακέτα μπορούμε, ενώ θα βρισκόμαστε στην κονσόλα του φυσικού μας συστήματος, να εκτελούμε εντολές οι οποίες μας παρέχουν πληροφορίες για το σύστημα των εικονικών μηχανών. Μια μικρή λίστα τέτοιου τύπου εντολών είναι η παρακάτω:

- Εμφάνιση των περιεχομένων του directory /root της εικονικής μηχανής test1 με χρήση της εντολής `ls -l`:

```
#  virt-ls -l -d test1 /root
```

- Εμφάνιση στο τερματικό του περιεχομένου του αρχείου `passwd` με την εντολή `cat`:

```
#  virt-cat -d test1 /etc/passwd
```

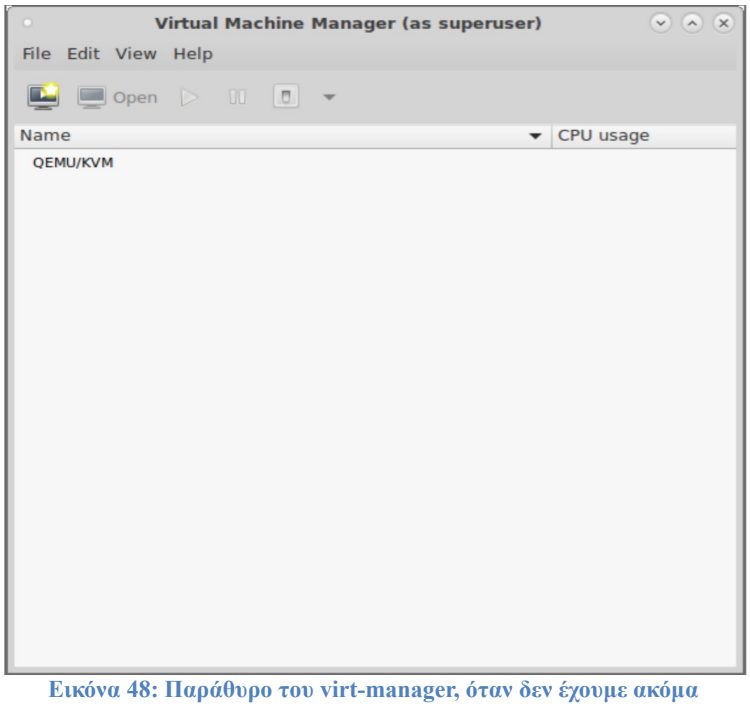
- Επεξεργασία του αρχείου `fstab` του συστήματος της εικονικής μηχανής test1:

```
#  virt-edit -d test1 /etc/fstab
```

- Εμφάνιση του partitioning που έχει γίνει στον σκληρό δίσκο της εικονικής μηχανής test1 με χρήση της εντολής `df`:

```
#  virt-df -d test1
```

Με το πέρας της εγκατάστασης αυτών των πακέτων, αν τώρα εκτελέσουμε την εντολή `virt-top` θα μας εμφανιστεί μόνο η εικονική μηχανή που δημιουργήσαμε μέσω της κονσόλας πρωτύτερα, η `vdebian`.



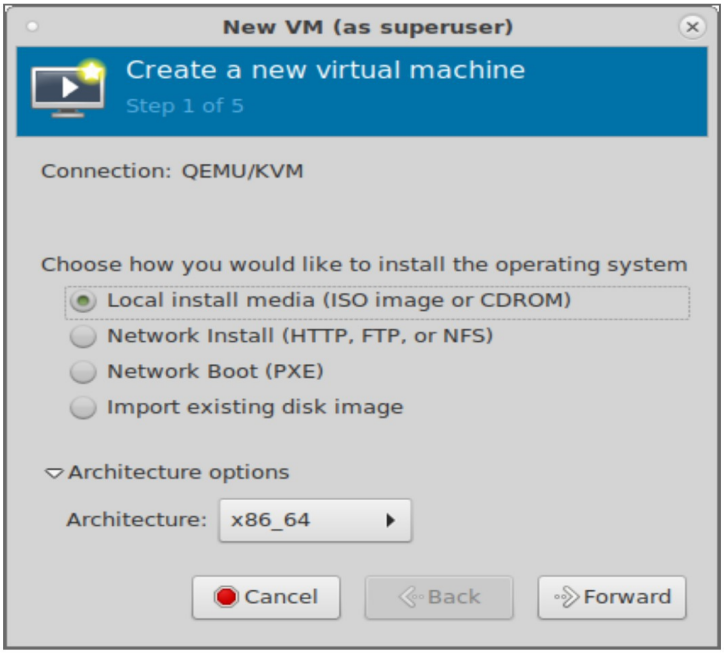
Εικόνα 48: Παράθυρο του virt-manager, όταν δεν έχουμε ακόμα δημιουργήσει καμία εικονική μηχανή.

Τώρα μπορούμε να ανοίξουμε τον `virt-manager` και μέσω αυτού να δημιουργήσουμε εικονικές μηχανές μέσω του γραφικού περιβάλλοντος που μας παρέχει.

Πατώντας το πρώτο εικονίδιο (και το μοναδικό που είναι ενεργό σε αυτό το σημείο), μπορούμε να ξεκινήσουμε την δημιουργία της εικονικής μας μηχανής.

Βήμα 1

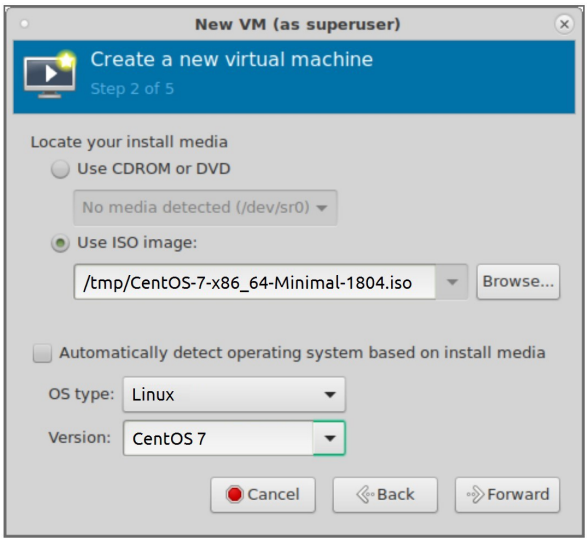
Στο πρώτο βήμα επιλέγουμε την πηγή στην οποία βρίσκεται το μέσο εγκατάστασης του λειτουργικού συστήματος που θέλουμε να εγκαταστήσουμε στην εικονική μηχανή. Μπορούμε να διαλέξουμε από τον τοπικό αποθηκευτικό χώρο ή αν θα κάνουμε εγκατάσταση δικτυακά. Επίσης, επιλέγουμε την αρχιτεκτονική του λειτουργικού που θα εγκαταστήσουμε και αν αυτό είναι 32-bit ή 64-bit. Στην περίπτωσή μας, το αρχείο .iso με την εγκατάστασή του λειτουργικού βρισκόταν αποθηκευμένο στον τοπικό μας δίσκο, οπότε επιλέξαμε την πρώτη επιλογή στην παρακάτω λίστα.



Εικόνα 49: Δηλώνουμε πού υπάρχει το μέσο με το λειτουργικό σύστημα που θέλουμε να εγκαταστήσουμε και την αρχιτεκτονική του.

Βήμα 2

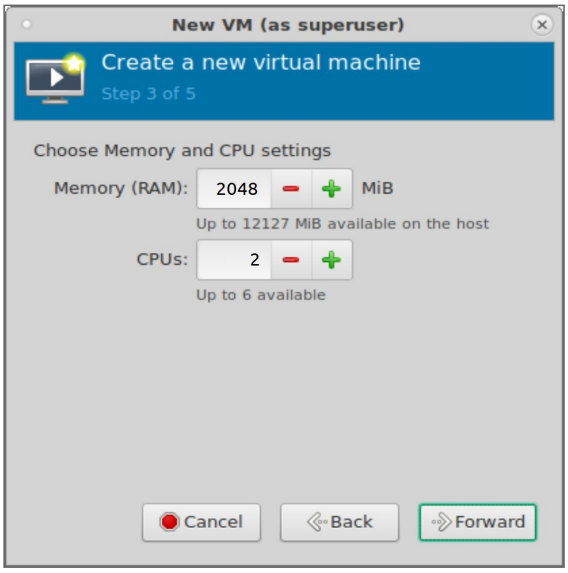
Σε αυτό το βήμα, και αφού έχουμε δηλώσει ότι το αρχείο .iso βρίσκεται στο τοπικό αποθηκευτικό μας χώρο, δηλώνουμε την ακριβή τοποθεσία του αρχείου. Το KVM σε αυτό το σημείο συμπληρώνει αυτόματα τα άλλα δύο πεδία με τον τύπο του λειτουργικού *(OS Type)* και την έκδοσή του *(Version)*.



Εικόνα 50: Εισαγωγή της τοποθεσίας του .iso αρχείου στο σύστημά μας.

Βήμα 3

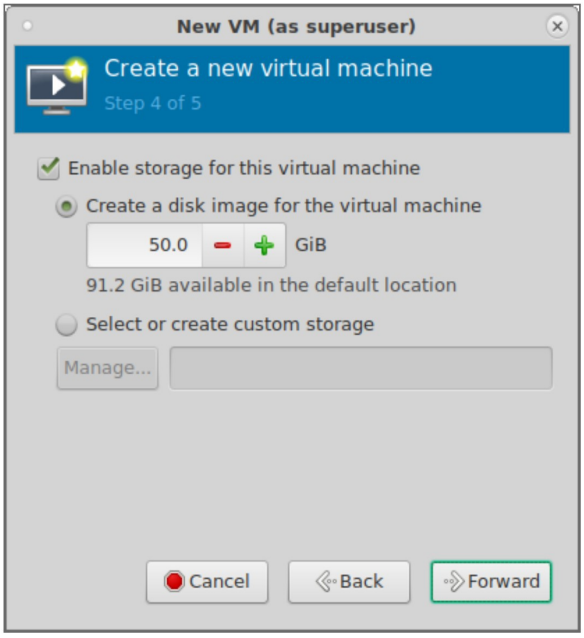
Στο τρίτο βήμα, δηλώνουμε πόση μνήμη RAM θέλουμε να έχει η εικονική μας μηχανή και πόσους πυρήνες ο επεξεργαστής της (*CPU*).



Εικόνα 51: Εισαγωγή μνήμης RAM και CPU.

Βήμα 4

Τώρα δηλώνουμε πόσο αποθηκευτικό χώρο θα δεσμεύσουμε για αυτή την εικονική μηχανή.

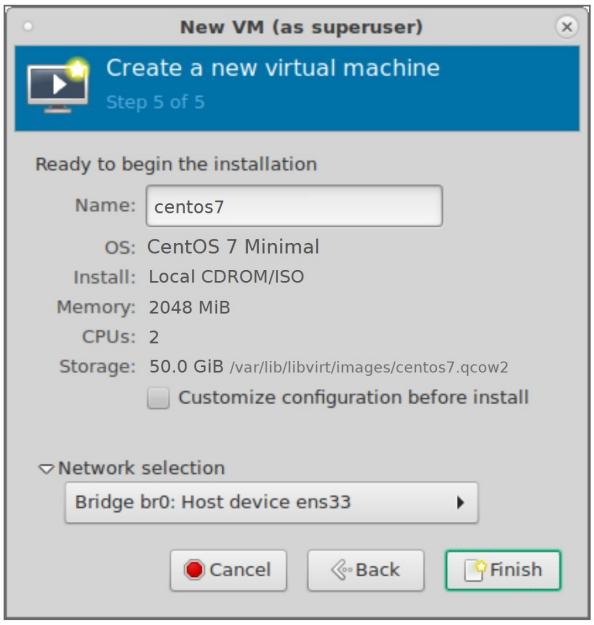


Εικόνα 52: Δημιουργία του σκληρού δίσκου της εικονικής μηχανής, μεγέθους 50GB.

Βήμα 5

Σε αυτό το βήμα δίνουμε το όνομα της εικονικής μας μηχανής και μας παρουσιάζονται συνοπτικά οι ρυθμίσεις που κάναμε στα προηγούμενα βήματα. Σε αυτό το σημείο ακόμα μπορούμε να αλλάξουμε γνώμη για κάποιες ή και για όλες τις ρυθμίσεις που έχουμε δώσει και μπορούμε να πατήσουμε το κουμπί *Back* και να πάμε στο αντίστοιχο βήμα όπου θέλουμε να κάνουμε τις αλλαγές μας.

Επίσης, σε αυτό το βήμα δηλώνουμε τον τρόπο με τον οποίο θα συνδέεται η εικονική μηχανή στο δίκτυο. Οι επιλογές είναι το default που παρέχει σύνδεση από το δικό μας σύστημα, *bridged* το οποίο κάνει χρήση της γέφυρα που έχουμε δημιουργήσει στο σύστημά μας και συνδέεται πάνω στο ενεργό network interface, κ.ά.



Εικόνα 53: Δήλωση ονόματος εικονικής μηχανής.

Πατώντας το *Finish,* ολοκληρώνουμε τις ρυθμίσεις, δηλαδή το configuration, της εικονικής μηχανής και ξεκινάει η εγκατάσταση του λειτουργικού στην εικονική μηχανή. Με την ολοκλήρωση της εγκατάστασης, στο παράθυρο του virt-manager φαίνεται πλέον η νέα εικονική μηχανή που δημιουργήσαμε και ονομάσαμε centos7.

3.7. CloudStack στο KVM

3.7.1. Τι είναι το CloudStack και τι Προσφέρει

Το Cloudstack της Apache είναι μία open source πλατφόρμα που προσφέρει Infrastructure as a Service. Διαχειρίζεται το σύνολο των αποθηκευτικών χώρων, το δίκτυο και πόρους των υπολογιστών ώστε να μπορεί να χτίσει ένα ιδιωτικό ή δημόσιο IaaS υπολογιστικό Cloud.

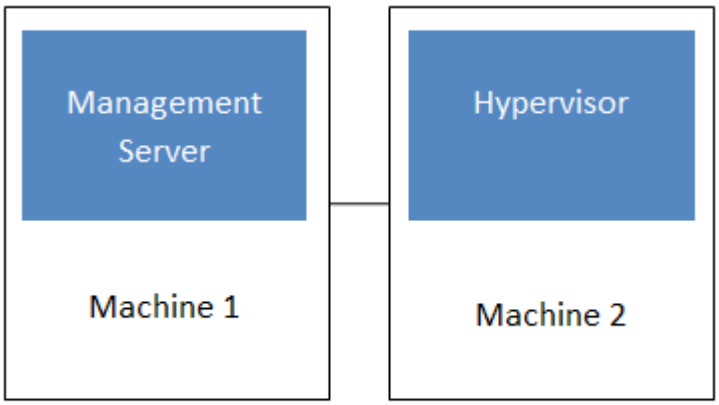
Υποστηρίζει πλήθος Hypervisors και τις τεχνολογίες τους. Ένα απλό Cloud του CloudStack μπορεί να περιέχει clusters που αντιστοιχούν σε διαφορετικές τεχνολογίες. Για παράδειγμα, ένα Cloud που έχει δημιουργηθεί με το CloudStack μπορεί να περιέχει, να διαχειρίζεται και να διαμοιράζει πόρους μεταξύ KVM , VMware και Xen χωρίς να αντιμετωπίζει κανένα πρόβλημα στη διαχείρισή τους.

Το CloudStack μπορεί να διαχειρίζεται δεκάδες χιλιάδες servers που είναι σε διαφορετικά γεωγραφικά Data Centers. Όπως είναι δομημένο το CloudStack, με έναν server ο οποίος έχει και εκτελεί το CloudStack η ανάγκη για άλλους Management Servers σχεδόν εκμηδενίζεται. Επιπλέον, για παράδειγμα, στην περίπτωση που ο server με το CloudStack πρέπει να σταματήσει να λειτουργεί για να του γίνει συντήρηση ή εάν γίνει μία διακοπής ρεύματος, το cloud δεν επηρεάζεται καθόλου και η λειτουργία του μπορεί να συνεχίζεται κανονικά.

Με τη δημιουργία κάθε καινούριας εικονικής μηχανής, το CloudStack ρυθμίζει αυτόματα το δίκτυό της και τον αποθηκευτικό της χώρο. Εσωτερικά, υπάρχουν αρκετά εικονικά στοιχεία (virtual appliances) τα οποία υποστηρίζουν τη λειτουργία του cloud. Στα στοιχεία αυτά συμπεριλαμβάνονται υπηρεσίες που προσφέρουν firewall, routing, DHCP, VPN, απομακρυσμένη κονσόλα, πρόσβαση στον αποθηκευτικό χώρο κ.ά. Γενικά, η δυνατότητα του CloudStack για οριζόντια κλιμάκωση των εικονικών μηχανών που χρησιμοποιούνται, απλοποιεί πολύ την εγκατάσταση και συνεχή χρήση του cloud που φτιάχνεται.

Το CloudStack προσφέρει γραφικό περιβάλλον χρήστη (GUI) το οποίο είναι παρόμοιο με αυτό του Cockpit που είδαμε νωρίτερα στο oVirt. Είναι και αυτό web interface και χρησιμοποιείται για την δημιουργία, διαχείριση, παρακολούθηση και συντήρηση, αλλά αυτή τη φορά αφορά το cloud και όχι τις εικονικές μηχανές καθεαυτές, αν και προσφέρεται ένα ακόμα GUI για τον τελικό χρήστη που χρησιμοποιείται για τις εικονικές μηχανές. Το GUI μπορεί να προσαρμοστεί έτσι ώστε να αντικατοπτρίζει τον επιθυμητό παροχέα ή να έχει μία συγκεκριμένη εμφάνιση και αίσθηση.

Γενικά η πιο απλή μορφή της αρχιτεκτονικής του CloudStack μπορεί να φανεί στην παρακάτω εικόνα η οποία είναι και η πιο βασική εγκατάσταση του CloudStack που μπορεί να γίνει και αποτελείται από έναν Management Server ο οποίος εκτελεί το Cloudstack και άλλον έναν υπολογιστή που περιέχει τον Hypervisor. Το deployment του Cloudstack μπορεί να απλοποιηθεί ακόμα, και να εφαρμοστεί στην πιο απλή του μορφή με τον έναν server να εκτελεί και το Cloudstack και να είναι host του Hypervisor.



Εικόνα 54: Απλοποιημένη μορφή του βασικού deployment του CloudStack.

3.7.1.1. Management Server

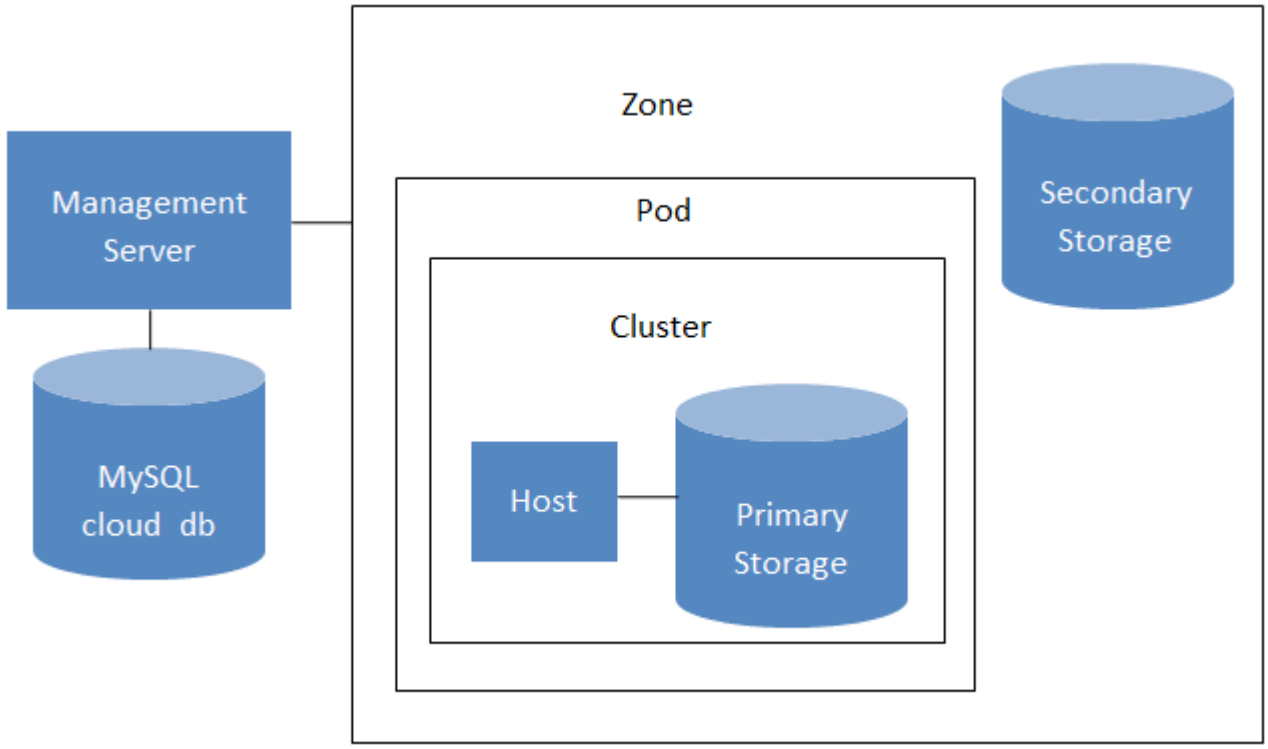
Ο Management Server οργανώνει και διαχειρίζεται τους πόρους του cloud μας. Συνήθως εκτελείται πάνω σε μία μηχανή που είναι αφιερωμένη σε αυτόν ή ως εικονική μηχανή. Ελέγχει τον διαμοιρασμό των πόρων και των εικονικών μηχανών στους hosts και αναθέτει διευθύνσεις IP και αποθηκευτικούς χώρους στις εικονικές μηχανές. Ο Management Server εκτελείται μέσα σε ένα container του Apache Tomcat και απαιτεί να υπάρχει μία βάση δεδομένων MySQL εγκατεστημένη και ρυθμισμένη για δουλεύει σωστά.

Γενικά, ο Management Server είναι υπεύθυνος για τις παρακάτω λειτουργίες:

- Παρέχει το web interface τόσο για τον διαχειριστή όσο και για τον τελικό χρήστη.
- Παρέχει τις διεπαφές API, τόσο για το CloudStack API όσο και για το EC2.
- Διαχειρίζεται τις αναθέσεις των guest VM σε συγκεκριμένους υπολογιστικούς πόρους.
- Διαχειρίζεται τις αναθέσεις των IP διευθύνσεων, τόσο των δημοσίων όσο και των ιδιωτικών.
- Κατανέμει τον αποθηκευτικό χώρο κατά τη διαδικασία δημιουργίας στιγμιότυπων εικονικών μηχανών.
- Διαχειρίζεται στιγμιότυπα, εικόνες δίσκων, πρότυπα (templates), και εικόνες ISO.
- Παρέχει ένα ενιαίο σημείο διαμόρφωσης για το Cloud μας.

3.7.1.2. Δομή του Cloud

Η δομή του Cloud που δημιουργείται με το cloudStack μπορεί να περιγραφεί στο παρακάτω σχήμα:



Εικόνα 55: Δομή του Cloud.

Οι πόροι που υπάρχουν μέσα σε ένα Cloud ομαδοποιούνται και διαχειρίζονται ως εξής:

- ***Regions***

Τα regions ή αλλιώς *περιοχές,* είναι μια ομάδα από μία ή περισσότερες ζώνες που βρίσκονται γεωγραφικά κοντά, και διαχειρίζονται από έναν ή περισσότερους Management Servers. Τα regions είναι η μεγαλύτερη μονάδα μέσα στο CloudStack και είναι μία πολύ χρήσιμη τεχνική κυρίως γιατί μπορούν να παρέχουν fault tolerance και disaster recovery.

- ***Zones***

Τα zones ή ζώνες, συνήθως αντιπροσωπεύουν ένα και μοναδικό Data Center. Η ζώνη αποτελείται από ένα ή περισσότερα pods και τον δευτερεύοντα αποθηκευτικό χώρο του Cloud. Εάν έχουμε παραπάνω από μία ζώνες, τότε ο δευτερεύοντας αποθηκευτικός χώρος που περιέχεται στη μία ζώνη, αυτόματα, μοιράζεται και στις υπόλοιπες ζώνες. Μία ζώνη μπορεί να περιέχει και έναν ή περισσότερους πρωτεύοντες αποθηκευτικούς χώρους οι οποίοι μοιράζονται ανάμεσα στα pods της ζώνης. Οι ζώνες είναι ορατές στον τελικό χρήστη.

- ***Pods***

Ένα pod είναι συνήθως ένα rack ή μία σειρά από rack που περιέχουν και ένα layer-2 switch, καθώς και ένα ή περισσότερα cluster. Οι hosts που υπάρχουν στο pod ανήκουν στο ίδιο υποδίκτυο (subnet). Τα pods δεν είναι ορατά στον τελικό χρήστη.

- ***Cluster***

Το cluster αποτελείται από έναν ή περισσότερους ομογενείς hosts καθώς και τον πρωτεύοντα αποθηκευτικό χώρο. Οι hosts στο cluster έχουν το ίδιο υλικό, εκτελούν τον ίδιο hypervisor, είναι στο ίδιο υποδίκτυο και έχουν πρόσβαση στον ίδιο πρωτεύοντα αποθηκευτικό χώρο.

- ***Host***

Ένας υπολογιστικός κόμβος, ένας Η/Υ, μέσα σε ένα cluster, συνήθως είναι και ο Hypervisor. Οι hosts στο CloudStack προσφέρουν CPU, μνήμη, αποθηκευτικούς και δικτυακούς πόρους. Συνδέονται χρησιμοποιώντας υψηλού εύρους TCP/IP network και σύνδεση στο Internet. Μπορεί, επίσης, να ανήκουν σε πολλαπλά Data Centers σε διαφορετικές γεωγραφικές περιοχές. Οι hosts μπορεί να είναι διαφορετικοί, να έχουν διαφορετική μνήμη, CPU ταχύτητες κ.λπ. αλλά οι hosts που ανήκουν στο ίδιο cluster πρέπει να είναι ομογενείς. Επιπλέον Hosts μπορούν να προστεθούν ανά πάσα στιγμή χωρίς να υπάρχει πρόβλημα, και έτσι να παρέχουν περισσότερο χώρο για τις εικονικές μηχανές.

- ***Primary Storage***

Το primary storage, ή αλλιώς ο πρωτεύων αποθηκευτικός χώρος, είναι ο αποθηκευτικός πόρος που παρέχεται σε ένα cluster και χρησιμοποιείται για την αποθήκευση των εικονικών δίσκων των εικονικών μηχανών που εκτελούνται στους hosts του cluster. Μπορούμε να έχουμε περισσότερους από έναν πρωτεύοντες αποθηκευτικούς χώρους σε ένα cluster ή σε μία ζώνη, αλλά ο ένας είναι υποχρεωτικός.

- ***Secondary Storage***

Το secondary storage, ή αλλιώς ο δευτερεύων αποθηκευτικός χώρος, αφορά ολόκληρη τη ζώνη και σε αυτόν αποθηκεύονται τα snapshots των εικονικών μηχανών, τα disk templates και οι εικόνες ISO. Πιο αναλυτικά, τα templates αφορούν σε εικόνες των λειτουργικών συστημάτων που μπορούν να χρησιμοποιηθούν για να γίνει boot στις εικονικές μηχανές και μπορεί να περιλαμβάνουν επιπλέον πληροφορίες διαμόρφωσης, όπως είναι οι εγκατεστημένες εφαρμογές στις εικονικές μηχανές. Οι εικόνες ISO είναι εικόνες δίσκων που περιέχουν δεδομένα ή bootable media για λειτουργικά συστήματα. Το snapshot των εικονικών μηχανών είναι αποθηκευμένα αντίγραφα των δεδομένων και πληροφοριών μίας εικονικής μηχανής και μπορεί να χρησιμοποιηθεί για ανάκτηση δεδομένων ή για να δημιουργηθεί νέο template.

Τα δεδομένα στον δευτερεύοντα αποθηκευτικό χώρο είναι διαθέσιμα σε όλους τους hosts που έχουν πρόσβαση σε αυτόν τον χώρο, δηλαδή ανά ζώνη/περιοχή.

3.7.1.3. Δικτύωση

Το CloudStack προσφέρει αρκετούς τύπους δικτύωσης, αλλά μπορούν να κατηγοριοποιηθούν σε δύο κατηγορίες: την **βασική** και την **προχωρημένη δικτύωση**.

Η βασική δικτύωση προσφέρει ένα επίπεδο layer-2 δίκτυο όπου στα guest συστήματα η “απομόνωση” παρέχεται από το layer-3 από τις γέφυρες του Hypervisor. Όταν χρησιμοποιείται η βασική δικτύωση, το CloudStack θα αντιστοιχήσει τις διευθύνσεις IP στο CIDR του pod, στους guests αυτού του pod. Ο διαχειριστής πρέπει να προσθέσει ένα εύρος άμεσων IP *(Direct IP range)* στο pod για τον σκοπό αυτό. Αυτές οι IP διευθύνσεις βρίσκονται στο ίδιο VLAN με του hosts.

Η προχωρημένη δικτύωση, συνήθως, χρησιμοποιεί την “απομόνωση” που παρέχεται από τα VLANs, και παράλληλα αυτή η κατηγορία προσφέρει τεχνολογίες SDN, όπως είναι το Nicira NVP. Στην προχωρημένη δικτύωση, μπορούν να υπάρχουν πολλαπλά φυσικά δίκτυα σε μία ζώνη. Το κάθε ένα από αυτά μπορεί να έχει διαφορετικούς τύπους κίνησης και πρέπει να ορίσουμε τι είδους κίνηση θέλουμε να έχει το κάθε δίκτυο. Οι τύποι κίνησης σε ένα προχωρημένο δίκτυο στο CloudStack μπορεί να είναι ένας από τους παρακάτω:

Guest

Όταν οι τελικοί χρήστες χρησιμοποιούν τις εικονικές μηχανές, τότε δημιουργείται η guest κίνηση δικτύου. Οι guest εικονικές μηχανές επικοινωνούν μεταξύ τους μέσω του δικτύου που ονομάζουμε guest δίκτυο. Αυτό το δίκτυο μπορεί να είναι απομονωμένο ή κοινό. Σε ένα απομονωμένο guest δίκτυο ο διαχειριστής χρειάζεται να εξασφαλίζει τμήματα VLAN ώστε να παρέχει την απομόνωση που χρειάζεται κάθε τέτοιο δίκτυο. Στο κοινό guest δίκτυο, όλες οι guest εικονικές μηχανές μοιράζονται το ίδιο δίκτυο.

Management

Η κίνηση management δημιουργείται όταν οι εσωτερικοί πόροι του CloudStack επικοινωνούν μεταξύ τους. Σε αυτή την κίνηση περιλαμβάνονται η επικοινωνία μεταξύ των hosts, των system VMs του CloudStack και οποιοδήποτε άλλο στοιχείο επικοινωνεί απευθείας με τον Management Server του CloudStack. Επίσης, εδώ πρέπει να ρυθμίσουμε το εύρος των IP που μπορούν να χρησιμοποιήσουν τα system VMs.

Public

Η δημόσια κίνηση δημιουργείται όταν οι εικονικές μηχανές στο cloud έχουν πρόσβαση στο Διαδίκτυο.

Storage

Πρόκειται συγκεκριμένα για την κίνηση του δευτερεύοντα αποθηκευτικού χώρου και δεν επηρεάζει την κίνηση του πρωτεύοντα. Στην storage κίνηση περιλαμβάνονται τα templates των εικονικών μηχανών και τα snapshots τα οποία στέλνονται μεταξύ των δευτερευόντων χώρων των εικονικών μηχανών και του δευτερεύοντα αποθηκευτικού server. Το CloudStack χρησιμοποιεί ξεχωριστό NIC που ονομάζεται NIC αποθήκευσης και είναι υπεύθυνο για την κυκλοφορία του δικτύου αποθήκευσης. Η χρήση ενός NIC αποθήκευσης το οποίο λειτουργεί πάντα σε υψηλού εύρους ζώνης δίκτυα, επιτρέπει την γρήγορη

αντιγραφή των templates και των snapshots. Και για αυτή την κίνηση δικτύου, πρέπει να ορίσουμε το εύρος IP διευθύνσεων που μπορούν να χρησιμοποιηθούν.

3.7.2. Δημιουργία Cloud με το CloudStack

Για να φτιάξουμε μία σωστή δομή, θα πρέπει να ακολουθήσουμε τα παρακάτω βήματα:

1. Διαλέγουμε και εγκαθιστούμε το επιθυμητό λειτουργικό σύστημα όπου θα εγκαταστήσουμε το CloudStack και όλες τις απαιτούμενες υπηρεσίες και εφαρμογές.
2. Ρυθμίζουμε το δίκτυο.
3. Ρυθμίζουμε τον αποθηκευτικό χώρο.
4. Εγκαθιστούμε τον Management Server.
5. Διαλέγουμε και εγκαθιστούμε τον Hypervisor που επιθυμούμε.
6. Ρυθμίζουμε το Cloud μας (ζώνες, pods, clusters, αποθηκευτικοί χώροι)

3.7.2.1. Λειτουργικό Σύστημα

Αποφασίσαμε να εξετάσουμε το CloudStack σε μία εικονική μηχανή ώστε να αποφασίσουμε αν είναι εφικτή η εφαρμογή του στο φυσικό σύστημα μέσα στον χρόνο που μας έχει διατεθεί. Φροντίσαμε να φτιάξουμε μία εικονική μηχανή η οποία θα ανταποκρίνεται στα προαπαιτούμενα του CloudStack.

Επειδή με το CentOS μπορούμε να ορίσουμε περισσότερες επιλογές και να δούμε το CloudStack σε μεγαλύτερο βάθος, επιλέξαμε να εγκαταστήσουμε τα CentOS 7 στην εικονική μηχανή. Η εγκατάσταση του KVM και πάλι στην εικονική μηχανή θα γίνει μέσω κονσόλας και είναι η ίδια με αυτή που προηγήθηκε σε προηγούμενο μέρος του κεφαλαίου.

Με το πέρας της εγκατάστασης των CentOS, μπορούμε να αρχίσουμε να ρυθμίζουμε την εικονική μηχανή μας ώστε να μπορεί να δεχτεί και να εκτελεί σωστά το CloudStack. Τα βήματα που ακολουθήσαμε είναι σχεδόν ίδια με αυτά του οδηγού που παρέχεται από το CloudStack στην ιστοσελίδα τους [13], αλλά έχουμε κάνει μερικές αλλαγές, καθώς επιλέξαμε διαφορετική έκδοση CentOS (CentOS 7 Minimal και όχι τα CentOS 6) και κάποιες από τις απαιτούμενες ρυθμίσεις χρειάζονταν είτε κάποια συγκεκριμένα πακέτα για να λειτουργήσουν σωστά ή κάποια επιπλέον βήματα. Επίσης, όλες οι εντολές που δίνουμε και οι ρυθμίσεις που κάνουμε συμβαίνουν μέσω του root χρήστη (`su -`).

3.7.2.2. Ρυθμίσεις Δικτύου

Το πρώτο πράγμα που πρέπει να κάνουμε σε οποιαδήποτε από τις εγκαταστάσεις μας είναι να ρυθμίσουμε σωστά το δίκτυο της μηχανής με την οποία θα δουλέψουμε. Με την εντολή `ip addr show` βλέπουμε τα network interfaces της μηχανής μας και τώρα μπορούμε να πάμε να επεξεργαστούμε το σωστό αρχείο με το συγκεκριμένο interface (`ens192`) το οποίο βρίσκεται στη θέση `/etc/sysconfig/network-scripts/ifcfg-ens192` και να προσθέσουμε τις παρακάτω γραμμές μέσα σε αυτό:

```
DEVICE=ens192
NM_CONTROLLED=no
ONBOOT=yes
IPADDR=192.168.10.245
NETMASK=255.255.255.0
GATEWAY=192.168.10.1
DNS1=192.168.10.16
DNS2=8.8.8.8
```

πατώντας το Esc και δίνοντας `:wq!` Γράφουμε τις αλλαγές που κάναμε στο έγγραφο με τον επεξεργαστή κειμένου vi και βγαίνουμε από το αρχείο.

Επειδή θέλουμε αυτές οι ρυθμίσεις να φορτώνονται άμεσα με την εκκίνηση λειτουργίας του μηχανήματος, δίνουμε την παρακάτω εντολή:

```
#  chkconfig network on
```

και τώρα εκκινούμε την υπηρεσία του network και στη συνέχεια κάνουμε επανεκκίνηση της εικονικής μηχανής, ώστε να είμαστε σίγουροι ότι εφαρμόστηκαν σωστά οι αλλαγές που κάναμε στο αρχείο του network interface.

```
#  service network start
#  reboot ή shutdown -r now
```

3.7.2.3. Hostname

Στη συνέχεια πρέπει να σιγουρευτούμε για το hostname του μηχανήματός μας. Έχοντας στο μυαλό μας ότι σε αυτό το μηχάνημα θα εκτελείται το CloudStack και οπότε θα έχουμε το Cloud μας εδώ, θα του δώσουμε ένα κατάλληλο όνομα στο αρχείο του `/etc/hosts`.

```
#  vi /etc/hosts
```

και προσθέτουμε την εξής γραμμή στο τέλος του αρχείου:

```
192.168.10.245 srvr1.cloud.priv
```

Κάνουμε επανεκκίνηση της υπηρεσίας network και ελέγχουμε ότι οι αλλαγή μας αυτή εφαρμόστηκε:

```
#  service network restart
#  hostname --fqdn
```

3.7.2.4 SELinux

Μέχρι στιγμής, για να λειτουργεί σωστά το CloudStack πρέπει το SELinux να είναι ρυθμισμένο στο permissive mode. Αυτό το επιτυγχάνουμε εκτελώντας την εντολή:

```
#  setenforce 0
```

και προσθέτοντας στο αρχείο `/etc/selinux/config` τις παρακάτω γραμμές:

```
SELINUX=permissive
SELINUX=targeted
```

3.7.2.5. NTP

Πρέπει να είναι σωστά ρυθμισμένο το ρολόι του συστήματος και πρέπει να είναι σωστά ρυθμισμένο το NTP (Network Time Protocol), ώστε όλα τα ρολόγια των συστημάτων που υπάρχουν στο cloud μας να έχουν την ίδια ώρα για να μην υπάρξουν προβλήματα συγχρονισμού μεταξύ τους ή μεταξύ λειτουργιών που επιθυμούν να εκτελέσουν. Αρχικά για να ρυθμίσουμε το NTP, πρέπει να υπάρχει στο σύστημά μας. Επειδή εμείς εγκαταστήσαμε τα CentOS 7 Minimal, πρέπει να κάνουμε εγκατάσταση του πακέτου με το NTP. Αυτό το κάνουμε ως εξής:

```
#  yum -y install ntp
```

Στη συνέχεια, θέλουμε το NTP να είναι ενεργό με την ενεργοποίηση του server μας (με το που γίνεται boot στο σύστημα), οπότε εκτελούμε την εντολή:

```
#  chkconfig ntpd on
```

Επίσης θέλουμε να ενεργοποιήσουμε την υπηρεσία του NTP, οπότε εκτελούμε:

```
#  service ntpd start
```

3.7.2.6. Προσθήκη του Repository του CloudStack

Τώρα πρέπει να ορίσουμε το repository του CloudStack. Για να το καταφέρουμε αυτό, δημιουργούμε το παρακάτω αρχείο:

```
#  vi /etc/yum.repos.d/cloudstack.repo
```

και γράφουμε τις εξής γραμμές:

```
[cloudstack]
name=cloudstack
baseurl=http://cloudstack.apt-get.eu/centos/7/4.11/
enabled=1
gpgcheck=0
```

με τις οποίες, με τη σειρά, ονομάζουμε το νέο repository που προσθέτουμε, δηλώνουμε την διαδικτυακή του τοποθεσία, το ενεργοποιούμε και δηλώνουμε ότι δεν θέλουμε να γίνεται έλεγχος των gpg κλειδιών.

3.7.2.7. Ρύθμιση του NFS

Σε αυτό το βήμα, πρέπει να ρυθμίσουμε το NFS (Network File System) για τον πρωτεύων και δευτερεύων αποθηκευτικό χώρο, οπότε πρέπει να δημιουργήσουμε δύο NFS shares για αυτόν τον λόγο. Αρχικά, εγκαθιστούμε το πακέτο με το NFS:

```
#  yum -y install nfs-utils
```

Και στη συνέχεια επεξεργαζόμαστε με τον vi editor το αρχείο `/etc/exports` ως εξής:

```
#  vi /etc/exports
```

και προσθέτουμε στο αρχείο τις παρακάτω γραμμές:

```
/secondary *(rw,async,no_root_squash,no_subtree_check)
/primary *(rw,async,no_root_squash,no_subtree_check)
```

Γράφουμε τις αλλαγές και βγαίνουμε από την επεξεργασία του αρχείου (`:wq!`) και τώρα πάμε να δημιουργήσουμε αυτά τα directories στο σύστημά μας.

```
#  mkdir -p /export/primary
#  mkdir /export/senondary
```

Πρέπει να ρυθμίσουμε το domain στο οποίο θα ανήκουν όλοι οι servers και οι χρήστες του cloud μας. Αυτό το καταφέρνουμε επεξεργάζοντας το αρχείο /etc/idmapd.conf:

```
#  vi /etc/idmapd.conf
```

και προσθέτοντας την παρακάτω γραμμή στο πεδίο *General:*

```
[General]
Domain=cloud.priv
```

Τώρα πρέπει να επεξεργαστούμε το αρχείο `/etc/sysconfig/nfs` και να βγάλουμε από τα σχόλια τις παρακάτω γραμμές:

```
LOCKD_TCPPORT=32803

LOCKD_UDPPORT=32769

MOUNTD_PORT=892

RQUOTAD_PORT=875

STATD_PORT=662

STATD_OUTGOING_PORT=2020
```

Γράφουμε τις αλλαγές στο αρχείο και βγαίνουμε από την επεξεργασία του. Εάν κάποια από τις παραπάνω γραμμές δεν συμπεριλαμβάνεται στο αρχικό αρχείο `/etc/sysconfig/nfs`, τότε μπορούμε να πάμε στο τέλος του αρχείου και να τις προσθέσουμε εκεί.

Για να εκτελεστούν σωστά τα παρακάτω βήματα πρέπει να σιγουρέψουμε ότι το πακέτο iptables-services είναι εγκατεστημένο στο σύστημά μας. Για να το εγκαταστήσουμε εκτελούμε την παρακάτω εντολή:

```
#  yum -y install iptables-services
```

Τώρα θα ανοίξουμε πάλι τον επεξεργαστή vi αυτή τη φορά για το αρχείο `/etc/sysconfig/iptables` και θα προσθέσουμε σε αυτό τις παρακάτω γραμμές:

```
-A INPUT -s 192.168.10.0/24 -m state --state NEW -p udp --dport 111 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 111 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 2049 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 32803 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p udp --dport 32769 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 892 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p udp --dport 892 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 875 -j
ACCEPT
```

```
-A INPUT -s 192.168.10.0/24 -m state --state NEW -p udp --dport 875 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p tcp --dport 662 -j
ACCEPT

-A INPUT -s 192.168.10.0/24 -m state --state NEW -p udp --dport 662 -j
ACCEPT
```

Σώζουμε το αρχείο μας, βγαίνουμε από το `vi` και κάνουμε επανεκκίνηση της υπηρεσίας των `iptables`.

```
#  service iptables restart
```

Και τώρα πρέπει να ενεργοποιήσουμε τις υπηρεσίες `rpcbind` και `nfs` και να τις ρυθμίσουμε ώστε να ξεκινάνε με την εκκίνηση του server:

```
#  service rpcbind start

#  service nfs start

#  chkconfig rpcbind on

#  chkconfig nfs on
```

3.7.2.8. Δημιουργία του Management Server – Δημιουργία και Ρυθμίσεις Βάσης Δεδομένων

Τώρα μπορούμε να προχωρήσουμε στην εγκατάσταση του Management Server του CloudStack. Για να το κάνουμε αυτό πρέπει να εκτελέσουμε τις παρακάτω εντολές:

```
#  yum -y install cloudstack-management
```

Και τώρα, αφού έχει εγκατασταθεί το λογισμικό, εγκαθιστούμε και τη βάση δεδομένων του με την εξής εντολή:

```
#  cloudstack-setup-databases cloud:password@localhost --deploy-as=root
```

Μόλις τελειώσει αυτή η διαδικασία, μας εμφανίζεται ένα μήνυμα ότι το CloudStack αρχικοποίησε επιτυχώς τη βάση δεδομένων , *"CloudStack has successfully initialized the database".*

Με την βάση δεδομένων να είναι έτοιμη, προχωράμε στο τελευταίο βήμα των ρυθμίσεων του Management Server και δίνουμε την εντολή:

```
# cloudstack-setup-management
```

Στην περίπτωση που ο servlet container μας είναι ο Tomcat7, τότε πρέπει να δώσουμε την παραπάνω εντολή με την παράμετρο -tomcat7, όπως φαίνεται παρακάτω:

```
# cloudstack-setup-management -tomcat7
```

3.7.2.9. Εγκατάσταση του CloudStack Management

Το CloudStack χρησιμοποιεί έναν αριθμό από system Vms για να μπορεί να παρέχει λειτουργικότητα και πρόσβαση στις κονσόλες των εικονικών μηχανών, παροχή σε διάφορες δικτυακές υπηρεσίες (firewall, DNS κ.λπ.), καθώς και διαχείριση του αποθηκευτικού χώρου. Με την παρακάτω εντολή, το σύστημά μας θα αποκτήσει αυτές τις εικονικές μηχανές οι οποίες είναι έτοιμες προς χρήση με την εκκίνηση του cloud μας.

```
# /usr/share/cloudstack-common/scripts/storage/secondary/cloud-install-
  sys-tmplt \
  >-m /export/secondary \
  >-u http://cloudstack.apt-get.eu/systemvm/4.11/systemvmtemplate-4.11.1-
  kvm.qcow2.bz2 \
  >-h kvm -F
```

Με την εκτέλεση αυτής της εντολής εγκαθίσταται το system VM στη θέση `/export/secondary/template/tmpl/1/3.`

Τώρα έχουμε έτοιμο τον Management Server μας.

3.7.2.10. Ετοιμασία Hypervisor

Πριν ξεκινήσουμε να ρυθμίζουμε το Cloudstack, πρέπει να ετοιμάσουμε τον Hypervisor μας.

Κάνουμε εγκατάσταση των απαραίτητων πακέτων για το KVM, τα οποία στην περίπτωσή μας, όπου ο Management Server θα είναι και ο host του Hypervisor μας, εμπεριέχονται στο πακέτο cloudstack-agent και

μπορούμε απλά να εκτελέσουμε την παρακάτω εντολή, με την οποία εγκαθίστανται και όλα τα απαραίτητα πακέτα για το KVM:

```
#  yum -y install cloudstack-agent
```

Τώρα ρυθμίζουμε το KVM, δηλαδή το QEMU και το libvirt. Αρχικά, για το QEMU, επεξεργαζόμαστε το αρχείο `/etc/libvirt/qemu.conf`:

```
#  vi /etc/libvirt/qemu.conf
```

και προσθέτουμε την παρακάτω γραμμή:

```
vnc_listen=0.0.0.0
```

Τώρα για να μπορεί να γίνει live migration πρέπει το libvirt να είναι ρυθμισμένο να ακούει τις TCP αιτήσεις και να μην προσπαθεί να χρησιμοποιήσει το Multicast DNS advertisement.

```
#  vi /etc/libvirt/libvirt.conf
```

και προσθέτουμε τις γραμμές:

```
listen_tls=0
listen_tcp=1
tcp_port=”16059”
auth_tcp=”none”
mdns_adv=0
```

Η παράμετρος listen_tcp=1 μέσα σε αυτό το αρχείο δεν είναι αρκετή από μόνη της και πρέπει να αλλάξουμε τις παραμέτρους και στο αρχείο /etc/sysconfig/libvirtd, ως εξής:

```
#  vi /etc/sysconfig/libvirtd
```

και βγάζουμε την παρακάτω γραμμή από τα σχόλια:

```
LIBVIRTD_ARGS=”--listen”
```

Επανεκκινούμε την υπηρεσία του libvirt:

```
#  service libvirtd restart
```

Και ελέγχουμε να δούμε αν το KVM είναι ενεργό και εκτελείται:

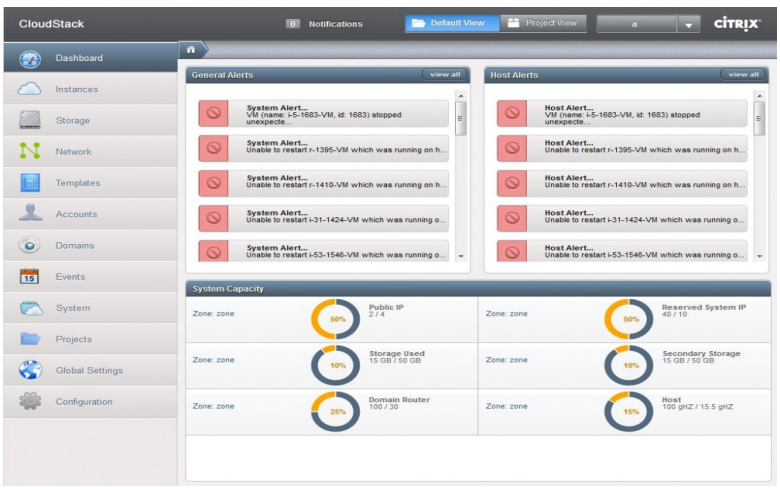
```
#  lsmod | grep kvm
```

3.7.2.11. Ρυθμίσεις Cloud

Μπορούμε πλέον να συνδεθούμε μέσω ενός web browser στην διεύθυνση http://192.168.10.245:8080/client και να ρυθμίσουμε το Cloud μας, φτιάχνοντας τις ζώνες του, ρυθμίζοντας τα pods του και το Cluster και προσθέτοντας τον πρωτεύοντα αποθηκευτικό χώρο στο Cluster και τον δευτερεύοντα στις ζώνες. Για να συνδεθούμε δίνουμε τα παρακάτω όνομα χρήστη και κωδικό:

```
username: admin
password: password
```

Δίνοντας τα παραπάνω στοιχεία, μπαίνουμε πια στην αρχική οθόνη του CloudStack όπου μπορούμε να διαλέξουμε να κάνουμε διάφορες ενέργειες και ρυθμίσεις στο Cloud μας και μοιάζει κάπως με την παρακάτω εικόνα:



Εικόνα 56: Δείγμα εικόνας του Dashboard του CloudStack[14].

Επιλέγουμε να συνεχίσουμε με τις βασικές ρυθμίσεις του CloudStack, οπότε και επιλέγουμε το *Basic Setup*.

3.7.2.11.1. Δημιουργία και Ρύθμιση Ζώνης

Η ζώνη είναι η μεγαλύτερη οντότητα στο CloudStack. Δημιουργώντας μία δίνουμε στα αντίστοιχα αναγραφόμενα πεδία τις παρακάτω τιμές:

- ***Name****:* Zone1
- ***Public DNS1****:* 192.168.10.16
- ***Public DNS2****:* 8.8.8.8
- ***Private DNS1****:* 192.168.10.16
- ***Private DNS2****:* 8.8.8.8

3.7.2.11.2 Ρύθμιση Pod

Αφού δημιουργήσουμε τη ζώνη, πρέπει να ρυθμίσουμε το pod. Στην περίπτωσή μας δώσαμε τα παρακάτω στοιχεία στα αντίστοιχα πεδία:

- ***Name:*** Pod1
- ***Gateway:*** 192.168.10.1
- ***Netmask:*** 255.255.255.0
- ***Start/end reserved system IPs:*** 192.168.10.10-192.168.10.20
- ***Guest gateway:*** 192.168.10.1
- ***Guest netmask:*** 255.255.255.0
- ***Guest start/end IP:*** 192.168.10.30-192.168.10.70

3.7.2.11.3 Ρυθμίσεις Cluster

Με τη δημιουργία και του pod, προχωράμε στις τελικές ρυθμίσεις για το Cluster:

- ***Name:*** Cluster1
- ***Hypervisor:*** KVM

Και τώρα πρέπει να προσθέσουμε τον πρώτο μας host στο Cluster:

- ***Hostname:*** 192.168.10.16, εδώ μπορούμε να χρησιμοποιήσουμε είτε την IP μας, 192.168.10.245, ή την IP του DNS server μας εφόσον είναι σωστά ρυθμισμένος και ενεργός.

- ***Username:*** root
- ***Password:*** ****, όπου και δίνουμε τον κωδικό του χρήστη root του συστήματος μας (αυτόν που έχουμε και στα CentOS).

3.7.2.11.4. Πρωτεύοντας Αποθηκευτικός Χώρος

Στη συνέχεια παρέχουμε τις πληροφορίες για τον πρωτεύοντα αποθηκευτικό μας χώρο. Διαλέγουμε το NFS και δίνουμε τα παρακάτω στοιχεία:

- ***Name:*** Primary1
- ***Server:*** 192.168.10.245
- ***Path:*** /export/primary

3.7.2.11.5 Δευτερεύοντας Αποθηκευτικός Χώρος

Και επαναλαμβάνουμε για τη ρύθμιση και σύνδεση και του δευτερεύοντα αποθηκευτικού χώρου, δίνοντας τις τιμές:

- ***NFS server:*** 192.168.10.245
- ***Path:*** /export/secondary

Πλέον, με τα πάντα ρυθμισμένα και συνδεδεμένα, πατώντας το κουμπί *Launch* το Cloud μας ξεκινάει την εγκατάστασή του. Ο χρόνος διάρκειας για την εγκατάσταση του Cloud εξαρτάται από την ταχύτητα που έχει η σύνδεσή μας με το δίκτυο.

3.8. Παρατηρήσεις και Συμπεράσματα

Γενικά, το KVM είναι μία σταθερή αξία και υπάρχει αρκετά χρόνια στα συστήματά μας. Είναι πολύ εύκολο και γρήγορο στην χρήση του. Δεν αντιμετωπίσαμε κανένα πρόβλημα με την εγκατάστασή του ή με την εγκατάσταση των εικονικών μηχανών σε αυτό. Το γραφικό του περιβάλλον (virt-manager) είναι βασικό, αλλά δεν είναι εμπλουτισμένο με περισσευούμενα γραφικά στοιχεία και λειτουργίες. Συνολικά, είναι λιτό, απλό και δυνατό στις λειτουργίες του.

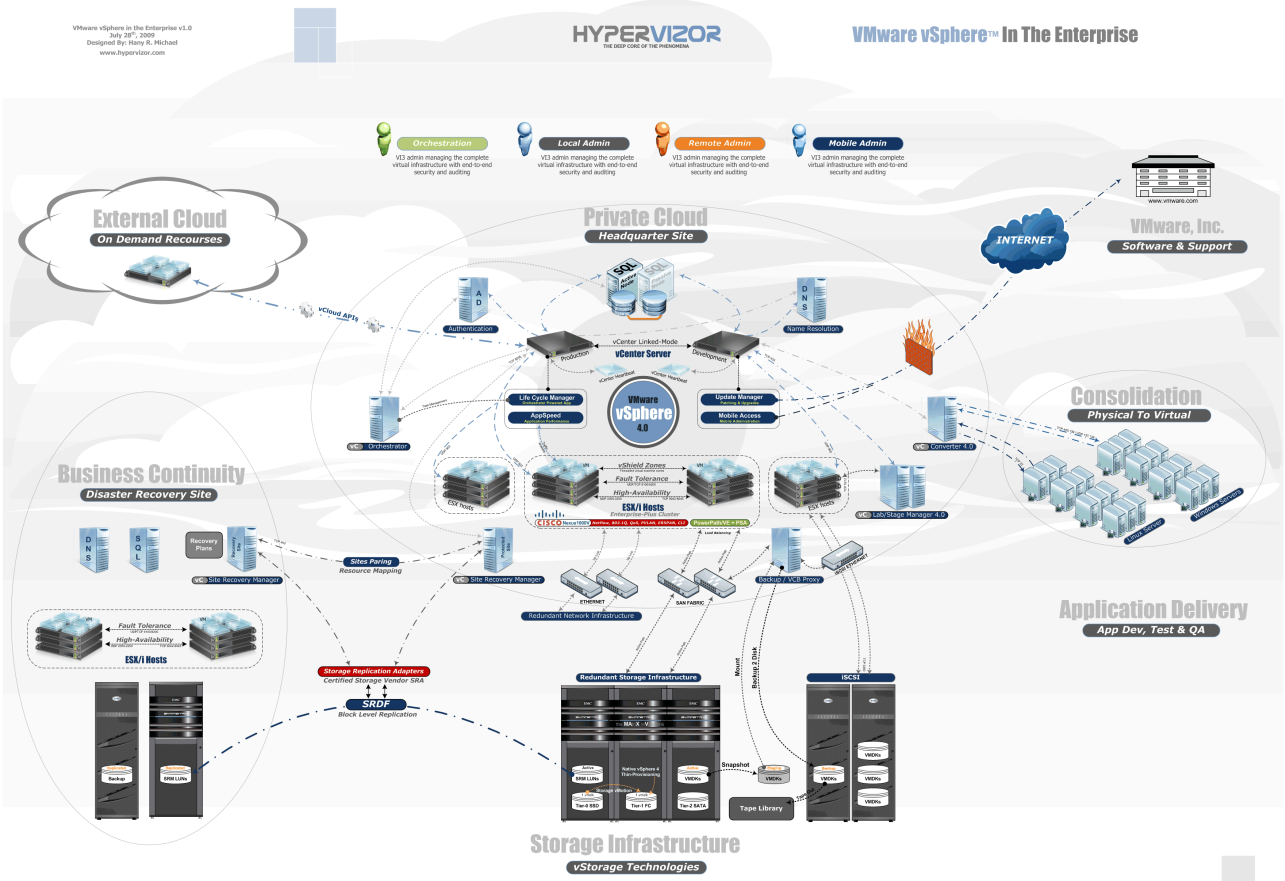
Το CloudStack, ενώ βγήκε στην αγορά σαν υποπροϊόν του OpenStack της NASA και της RACKSPACE, είναι ένα πολύ δυνατό εργαλείο και έχει ήδη υιοθετηθεί από πολλούς και μεγάλους οργανισμούς που επιθυμούν να προσφέρουν υπηρεσίες Cloud IaaS στους χρήστες τους παρέχοντας μία ολοκληρωμένη σουίτα. Η εγκατάστασή του ήταν αρκετά εύκολη, δεν αντιμετωπίσαμε κάποιο σημαντικό πρόβλημα και η εμπειρία χρήσης του ήταν πολύ ικανοποιητική.

ΚΕΦΑΛΑΙΟ 4 – VMWARE

4.1. Περίληψη Κεφαλαίου

Σε αυτό το κεφάλαιο θα εξετάσουμε την πλατφόρμα εικονικοποίησης VMware. Θα εγκαταστήσουμε σε έναν server τον Hypervisor της Vmware, VMware ESXi 6.5 και θα τον διαχειριστούμε απομακρυσμένα ενώ και ο VMware ESXi και ο υπολογιστής μας θα βρίσκονται στο ίδιο τοπικό δίκτυο μέσω ενός προγράμματος περιήγησης. Θα δημιουργήσουμε και διαχειριστούμε εικονικές μηχανές μέσω του VMware ESXi και θα συνδέσουμε τον Hypervisor με το CloudStack. Τέλος, θα κάνουμε καταγραφή ενός ήδη υπάρχοντος συστήματος VMware ESXi 5.1.

4.2. Γενικές Πληροφορίες



Εικόνα 57: Συνολική εικόνα του VMware vSphere

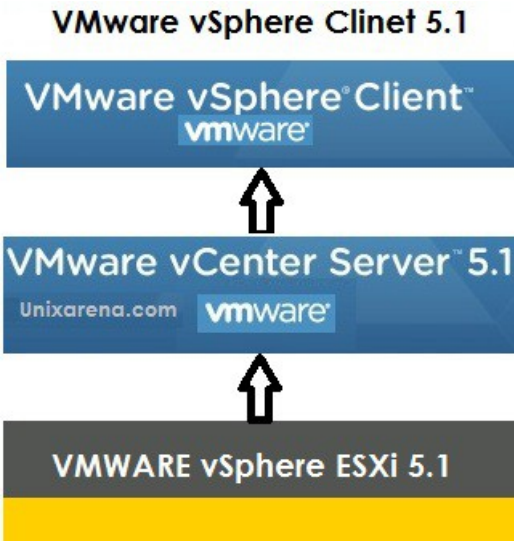
Το VMware είναι λογισμικό εικονικοποίησης και ανήκει στην εταιρεία VMware Inc. η οποία ιδρύθηκε το 1998 και εδρεύει στο Palo Alto της California. Τα προϊόντα της VMware είναι σχεδιασμένα για πλειάδα λειτουργικών συστημάτων (Microsoft Windows, Linux, Mac OS X).

Τα προϊόντα της VMware παρέχουν ένα πλήρες εικονικοποιημένο υλικό στο guest λειτουργικό σύστημα. Παρέχουν δηλαδή, εικονική κάρτα γραφικών, εικονική κάρτα δικτύων, εικονικό σκληρό δίσκο καθώς και εικονικούς drivers για παράλληλες, σειριακές και usb θύρες. Καθότι το υλικό είναι εικονικοποιημένο δεν

υπάρχει κανένα πρόβλημα συμβατότητας με την μεταφορά των εικονικών μηχανών από έναν υπολογιστή σε κάποιον άλλον.

Ο διαχειριστής του συστήματος, μπορεί να δημιουργήσει εικονικές μηχανές, να κάνει παύση των λειτουργιών του, να το μεταφέρει σε έναν άλλον τοπικό υπολογιστή, αν το αντιγράψει ή να το κλωνοποιήσει ή ακόμα και να το ορίσει ως template για επόμενες εικονικές μηχανές. Η ευελιξία που παρέχεται με τη χρήση της Εικονικοποίησης είναι ο κύριος λόγος για τον οποίο το Virtualization έχει κερδίσει τόσο μεγάλο κοινό και υποστηρικτές.

Τα προϊόντα της VMware είναι κλειστού κώδικα και με μειωμένα τα δικαιώματα του guest συστήματος. Τα προϊόντα VMware ανήκουν και στις δύο κατηγορίες Hypervisor, την bare-metal και την desktop έκδοση. Στην περίπτωση του bare-metal, το VMware παίρνει την θέση του τοπικού, φυσικού λειτουργικού συστήματος στο μηχάνημά μας (VMware ESX, ESXi). Στην desktop περίπτωση, το VMware εγκαθίσταται ως ένα λογισμικό πάνω στο ήδη υπάρχον λειτουργικό, είτε αυτό είναι Linux ή Windows ή Mac OS X (VMware Workstation, Fusion, vSphere).



Εικόνα 58: Ιεραρχία VMware vSphere ESXi.

Το VMware προσφέρει πια και το ***"Hands on Lab"*** όπου μπορούμε να δοκιμάσουμε το VMware σε ένα ασφαλές περιβάλλον εξομοίωσης χωρίς να χρειαστεί να το έχουμε αγοράσει. Δυστυχώς, όμως για να έχουμε πρόσβαση σε αυτό πρέπει να κάνουμε εγγραφή στο site τους και τελικά δεν καταφέραμε να δούμε το ***"Hands on Lab"*** της VMware. Ωστόσο μπορέσαμε να κατεβάσουμε την δοκιμαστική έκδοση του VMware ESXi και να την εγκαταστήσουμε για να μπορέσουμε να αποκτήσουμε μία γνώση της χρήσης του VMware λογισμικού.

4.3. VMware ESXi

Ο VMware ESXi είναι ο Hypervisor της Vmware. Είναι Hypervisor τύπου-1 και αναπτύχθηκε για τη δημιουργία και διαμοιρασμό εικονικών υπολογιστών στους πελάτες της. Καθότι είναι Hypervisor τύπου-1, αυτό σημαίνει ότι έχει δικό της λειτουργικό σύστημα και εγκαθίσταται ως ένα στον server. Η αρχιτεκτονική του VMware ESXi από το λειτουργικό σύστημα που ανέπτυξε η VMware και ονομάζεται Vmkernel.

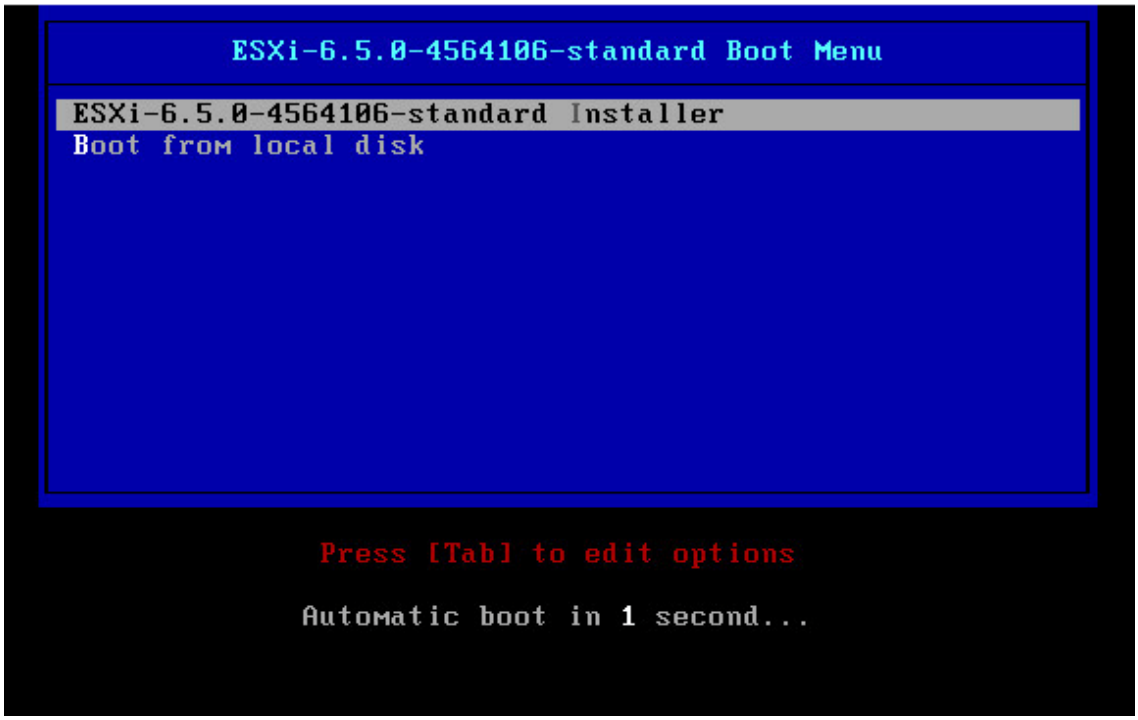
Ο VMware ESXi 6.5 παρέχεται από την VMware για δοκιμαστική χρήση χρονικής περιόδου εξήντα (60) ημερών. Διαλέξαμε να εξετάσουμε την νέα έκδοση του VMware ESXi και παρακάτω παρατίθενται οι

λεπτομέρειες εγκατάστασής τους, η δημιουργία εικονικών μηχανών και, σε δεύτερη φάση, η εισαγωγή του host του VMware στο CloudStack.

4.3.1. Εγκατάσταση VMware ESXi

Μπορούμε να κατεβάσουμε το .iso bootable αρχείο από την επίσημη ιστοσελίδα της VMware. Κάνουμε burn το αρχείο αυτό σε ένα CD-ROM ή το περνάμε σε ένα USB drive με χρήση μία εφαρμογής όπως είναι η Rufus ή η Linux Live με την οποία το USB drive μπορεί να χρησιμοποιηθεί ως μέσο εγκατάστασης ενός λειτουργικού.

Όταν κάνουμε εκκίνηση στον server μέσω του μέσου εγκατάστασης του λειτουργικού του VMware ESXi που έχουμε επιλέξει η παρακάτω είναι η πρώτη εικόνα που θα δούμε.



Εικόνα 59: Αρχική εικόνα εγκατάστασης του VMware ESXi.

Επιλέγουμε ότι επιθυμούμε να ξεκινήσει η εγκατάσταση του VMware ESXi και συνεχίζουμε με την εγκατάσταση.

Η εγκατάσταση αυτή είναι πολύ αυτοματοποιημένη και οι μόνες φορές, κατά τη διάρκειά της, που θα χρειαστεί να εισάγουμε κάποια πληροφορία είναι όταν θα μας ζητηθεί να επιλέξουμε σε ποιον δίσκο επιθυμούμε να γίνει η εγκατάσταση του VMware ESXi και να δώσουμε έναν κωδικό για τον root χρήστη.

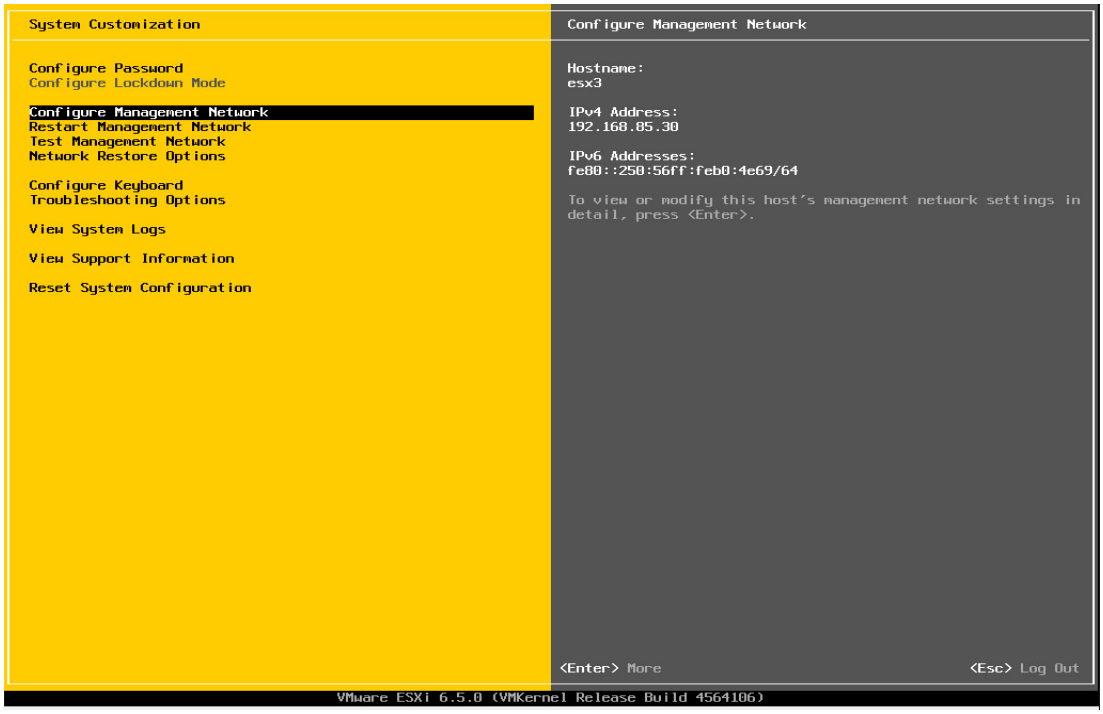
Με το πέρας της εγκατάστασης του VMware ESXi, ο server χρειάζεται να κάνει επανεκκίνηση και έπειτα και από αυτό είναι έτοιμος για λειτουργία. Η αρχική οθόνη του VMware ESXi είναι η παρακάτω:

Εικόνα 60: Αρχική οθόνη VMware ESXi.

Κανονικά, τώρα πρέπει να ρυθμίσουμε το δίκτυο του server, αλλιώς θα έχει τη διεύθυνση loopback 0.0.0.0. Στην περίπτωσή μας, πήρε αυτόματα τη διεύθυνση IP που θέλαμε καθώς ήταν έτσι ρυθμισμένη η πόρτα του υπολογιστή από την αρχή, καθώς πρωτύτερα φιλοξενούσε μία άλλη εγκατάσταση. Σε περίπτωση που δεν είναι ήδη ρυθμισμένη η πόρτα του υπολογιστή, τότε θα χρειαστεί μετά την πρώτη εκκίνηση του server να ρυθμίσουμε το δίκτυό του. Για να το κάνουμε αυτό ακολουθούμε τα παρακάτω βήματα:

1. Ενώ βρισκόμαστε στην αρχική οθόνη του VMware ESXi, πατάμε το πλήκτρο F2.
2. Εμφανίζεται παράθυρο στο οποίο μας ζητείται να εισάγουμε τον κωδικό του root χρήστη.
3. Με το που γίνει η εισαγωγή μας στο σύστημα, εμφανίζεται η παρακάτω εικόνα (59), στην οποία βλέπουμε ότι υπάρχει μία λίστα από ρυθμίσεις που μπορούμε να κάνουμε πάνω στον server μας. Εμείς θέλουμε να ορίσουμε μία IP διεύθυνση για τον VMware ESXi, οπότε επιλέγουμε το *Configure Management Network.*
4. Βρισκόμαστε πλέον μέσα στο μενού για τη ρύθμιση του δικτύου όπου μπορούμε να δώσουμε την IP διεύθυνση που επιθυμούμε. Πατάμε το Enter και δίνουμε την IP, την μάσκα του δικτύου και τουλάχιστον μία διεύθυνση gateway. Αποθηκεύουμε τις επιλογές μας και βγαίνουμε από το υπομενού αυτό.

Η παρακάτω εικόνα (59) αποτελεί δείγμα των ρυθμίσεων που πρέπει να κάνουμε.



Εικόνα 61: Ρύθμιση δικτύου του VMware ESXi.

4.3.2. Πρόσβαση μέσω VMware vSphere vClient

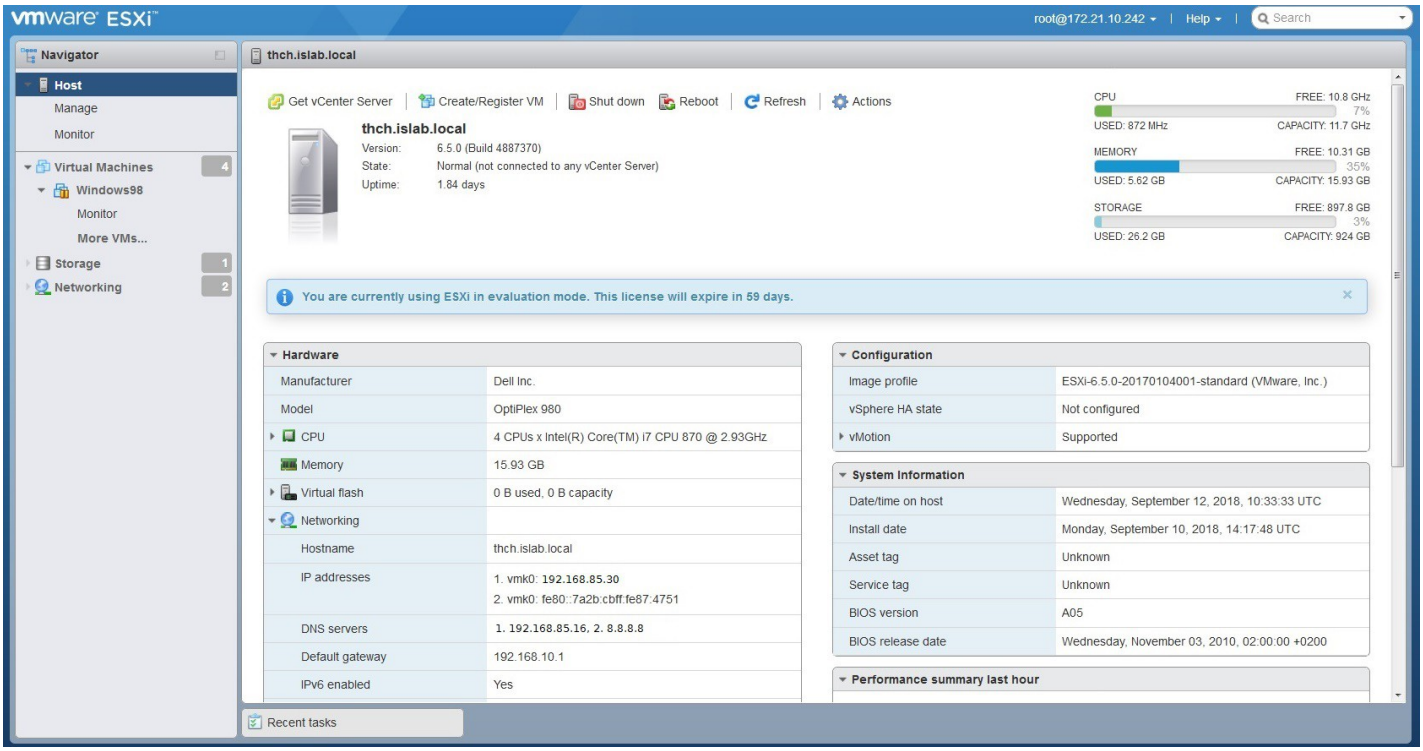
Με τον VMware ESXi έτοιμο και με μία στατική IP διεύθυνση δοσμένη σε αυτόν, μπορούμε μέσω άλλου υπολογιστή που βρίσκεται στο ίδιο τοπικό δίκτυο να προσπελάσουμε τον VMware ESXi server με δύο τρόπους.

Ο πρώτος τρόπος και ο πιο γνωστός, καθώς αυτός ίσχυε από την αρχή της διανομής των προϊόντων της Vmware, είναι μέσω του VMware vSphere Client, όπου εμφανίζεται ένα παράθυρο και εισάγουμε την IP διεύθυνση του VMware ESXi server και τα στοιχεία του root χρήστη μας για να εισαχθούμε στο σύστημα.

Εικόνα 62: Εισαγωγή στο VMware ESXi μέσω του VMware vSphere Client.

Το vSphere αποτελεί και το κυρίως προϊόν της VMware.

Ο δεύτερος τρόπος είναι μέσω ενός web browser. Ανοίγοντας έναν web browser και δίνοντας στο πεδίο URL την IP διεύθυνση ως http://192.168.85.30:8080, μπαίνουμε στην αρχική οθόνη της VMware όπου και μας ζητείται να δώσουμε τα διαπιστευτήριά μας (όνομα χρήστη και κωδικό). Με την εισαγωγή των έγκυρων στοιχείων βρισκόμαστε πια στο γραφικό περιβάλλον που παρέχει η VMware και μπορούμε να δημιουργήσουμε, αναπτύξουμε, αντιγράψουμε, διαθέσουμε εικονικές μηχανές και υπηρεσίες μέσω αυτών.

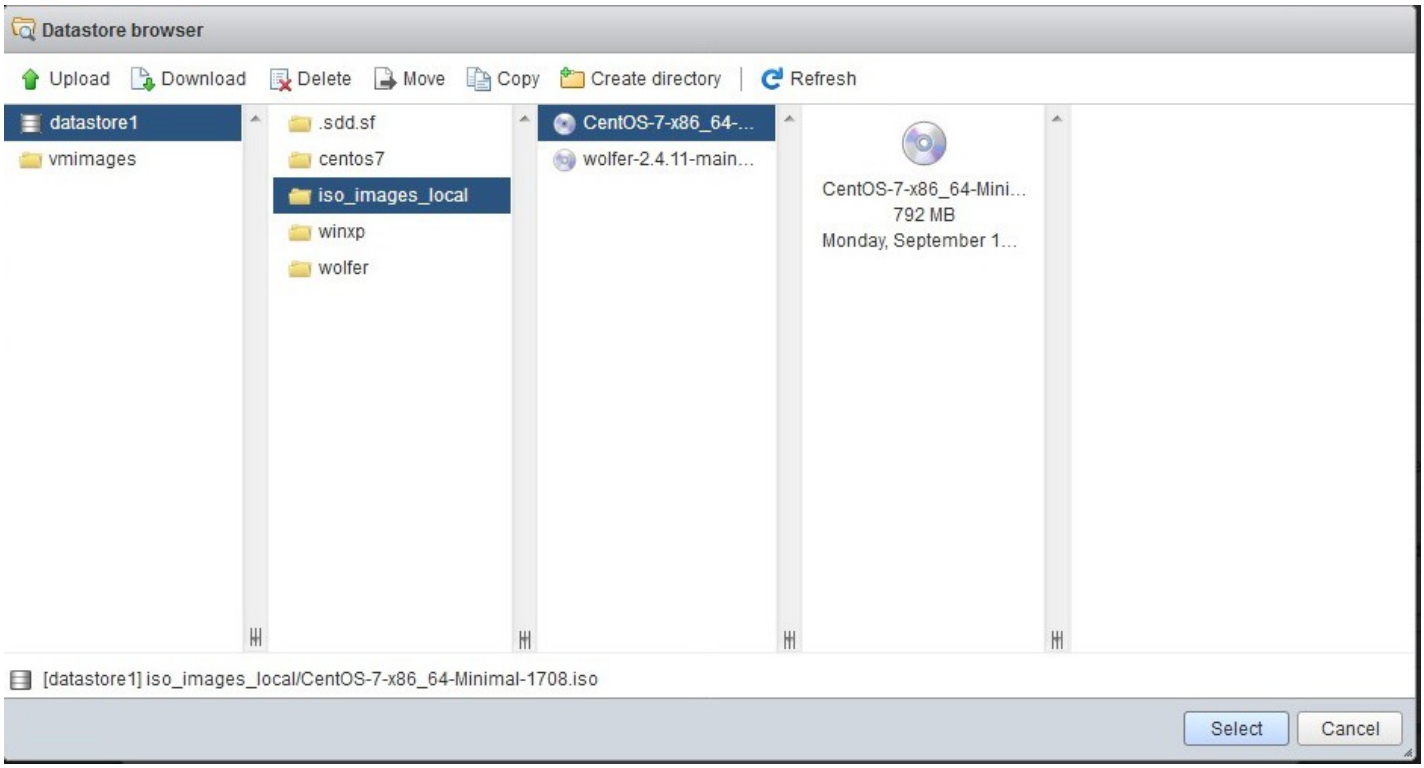


Εικόνα 63: Αρχική οθόνη VMware ESXi 6.5 μέσω web browser.

4.3.3. Προσθήκη ISO στον VMware ESXi

Για να δημιουργήσουμε μία εικονική μηχανή, χρειάζεται να έχουμε περάσει το .iso αρχείο της σε έναν φάκελο μέσα στο VMware ESXi, ο οποίος θα είναι κοινός και θα ορίζεται ως ο CD/DVD Driver της εικονικής μηχανής αρχικά. Έτσι θα φορτώσει από το CD/DVD Driver στην πρώτη εκκίνηση της εικονικής μηχανής και θα μπορέσουμε να κάνουμε εγκατάσταση στο λειτουργικό σύστημα που θέλουμε.

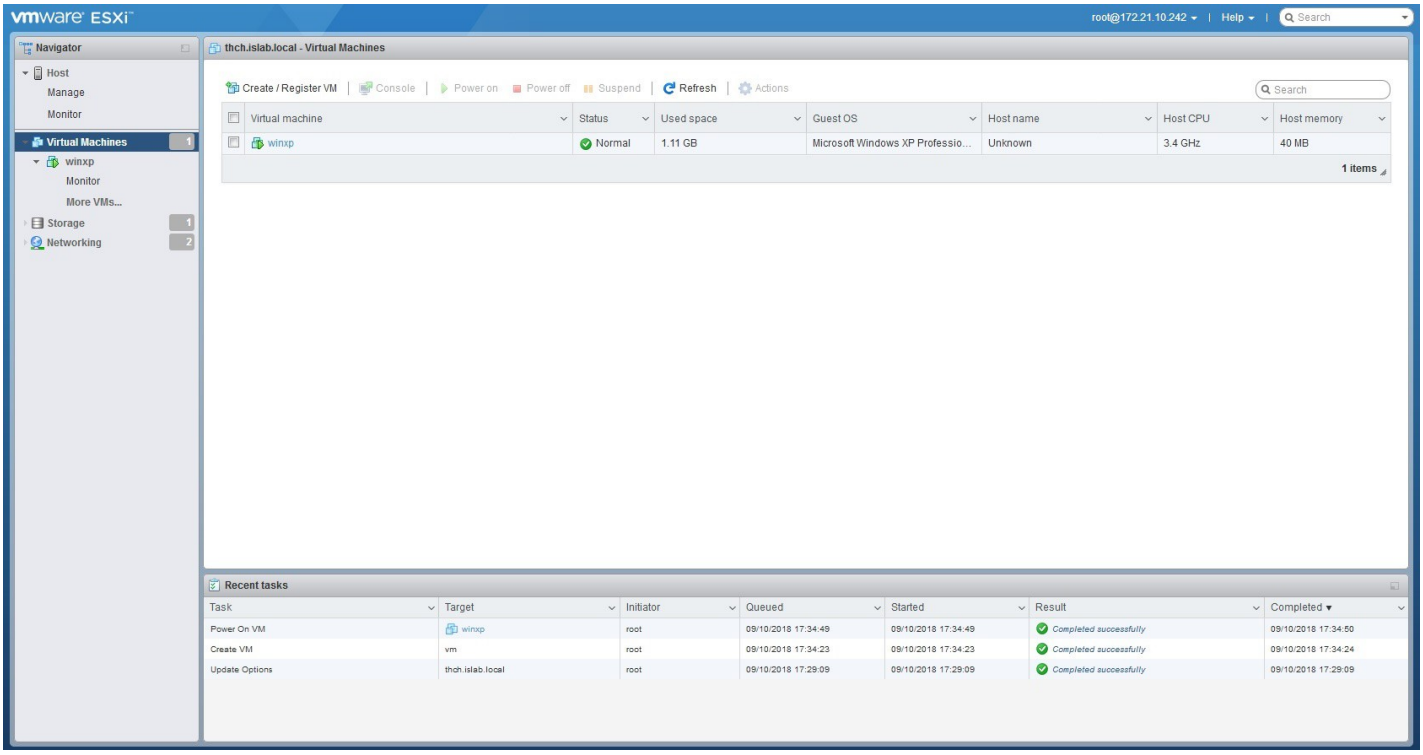
Η διαδικασία εισαγωγής αρχείων .iso σε κοινό φάκελο στο VMware ESXi είναι αρκετά απλή και άμεση. Το μόνο μειονέκτημα που βρήκαμε ήταν ότι τα αρχεία .iso πρέπει να φορτωθούν ένα-ένα καθώς δεν μπορεί να γίνει πολλαπλή επιλογή αρχείων σε αυτό το σημείο. Το παράθυρο εισαγωγής των αρχείων φαίνεται στην επόμενη σελίδα.



Εικόνα 64: Παράθυρο εισαγωγής αρχείων .iso στο κοινό αποθηκευτικό χώρο του VMware ESXi 6.5.

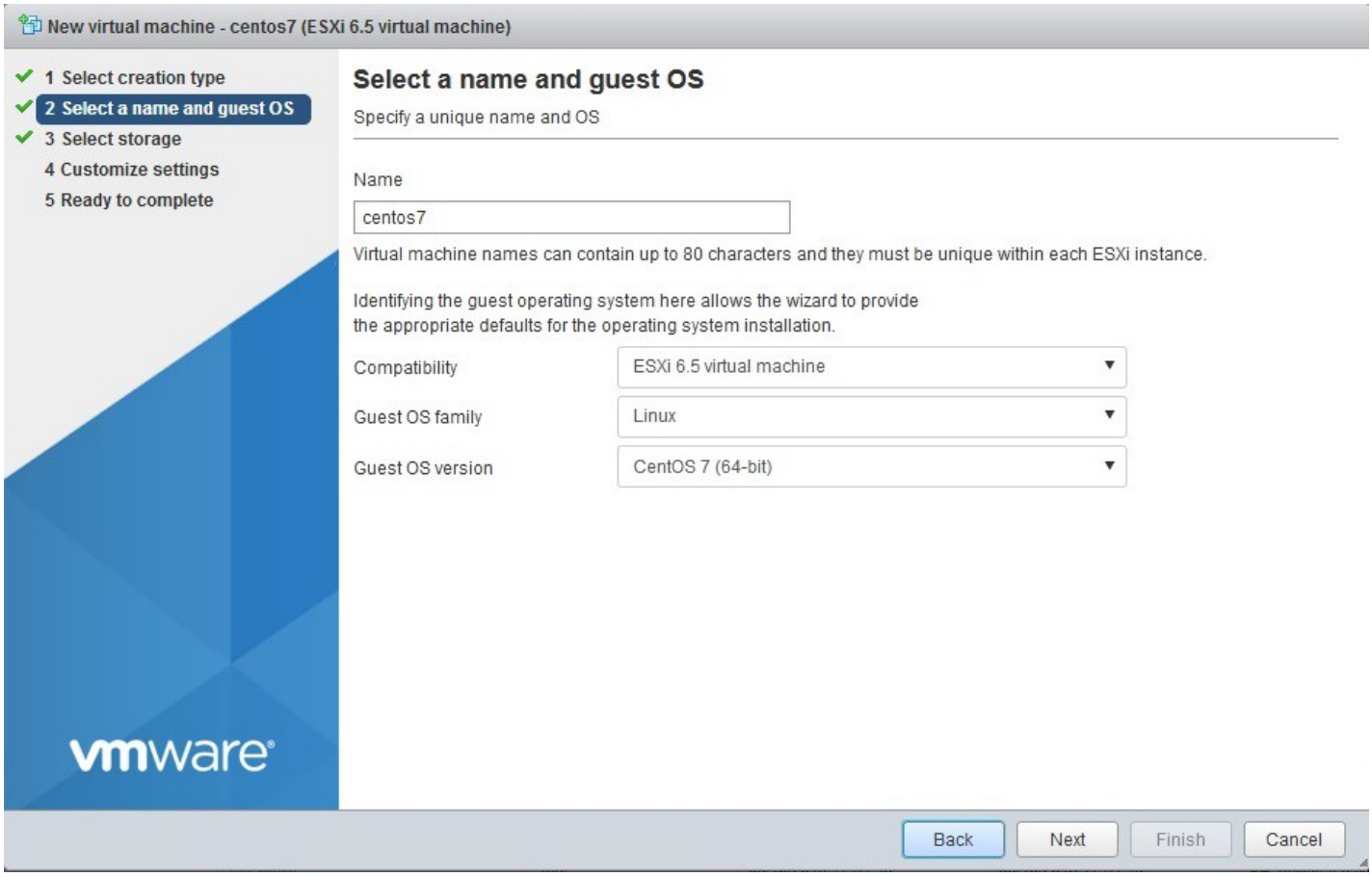
4.3.4. Δημιουργία Εικονικών Μηχανών

Η δημιουργία των εικονικών μηχανών στο VMware είναι πολύ εύκολη και άμεση. Επιλέγουμε το *Create/Register VM,* με το οποίο ξεκινά να εκτελείται ένας wizard ο οποίος μας καθοδηγεί στη δημιουργία της εικονικής μηχανής.



Εικόνα 65: Σελίδα με τις εικονικές μηχανές του VMware ESXi όπως αυτές φαίνονται από το πρόγραμμα περιήγησης.

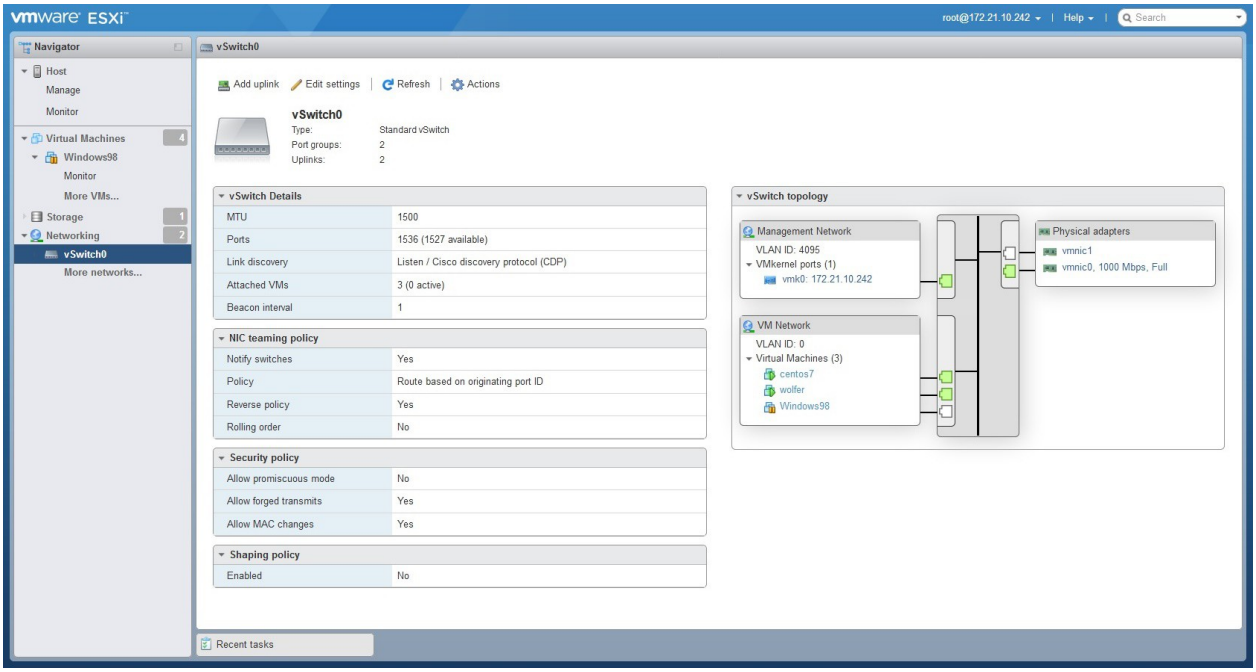
Η διαδικασία δημιουργίας νέας εικονικής μηχανής είναι πολύ απλή και μοιάζει πολύ με ό,τι έχουμε δει μέχρι τώρα. Είναι μια αυτοματοποιημένη διαδικασία η οποία σκοπό έχει να κάνει εύκολη την όλη λειτουργία και να δημιουργεί τις εικονικές μηχανές που χρειάζεται μία επιχείρηση ή ένας οργανισμός όσο πιο γρήγορα και σταθερά γίνεται ώστε να βγει άμεσα προς εκμετάλλευση.



Εικόνα 66: Ο wizard δημιουργίας νέας εικονικής μηχανής.

4.3.5. VLANs, Τοπολογίες και Συνδέσεις στον VMware ESXi

Στο VMware, αυτόματα, όλες οι εικονικές μηχανές είναι τοποθετημένες πάνω σε ένα εικονικό switch και ανήκουν στο ίδιο VLAN. Σκοπός αυτού είναι να δώσει τη δυνατότητα στους πελάτες της, να δημιουργήσουν εικονικά την τοπολογία που επιθυμούν για το δίκτυό τους χωρίς να είναι ανάγκη να το προσπαθήσουν κατευθείαν στα πραγματικά switches. Έτσι, μπορούμε να δημιουργούμε πρώτα εικονικά τις τοπολογίες και συνδέσεις που επιθυμούμε μέχρι να βρούμε την ιδανική για τις ανάγκες μας. Τότε, μπορούμε να συνδέσουμε τα φυσικά μηχανήματα στα φυσικά switches με την καταγεγραμμένη συνδεσμολογία που έχουμε κάνει.



Εικόνα 67: Εικονικό switch.

4.4. Καταγραφή Υπάρχοντος Συστήματος VMware με δύο nodes ESXi 5.1

Κάνοντας μία καταγραφή για το σύστημα του VMware που ήδη υπάρχει στον Δημόκριτο, μπορέσαμε να έρθουμε πιο κοντά στις πραγματικές καταστάσεις και ανάγκες ενός οργανισμού.

Εικόνα 68: Αρχικό παράθυρο VMware vSphere Client.

Για να εισαχθούμε στο ήδη υπάρχον σύστημα, πρέπει ο υπολογιστής στον οποίο εργαζόμαστε να έχει εγκατεστημένο το VMware vSphere Client. Μόλις το κάνουμε εγκατάσταση, εμφανίζεται το παράθυρο

εισαγωγής χρήστη, στο οποίο και πληκτρολογούμε τα διαπιστευτήριά μας (IP διεύθυνση server, όνομα χρήστη και κωδικό χρήστη).

Στο production system του οργανισμού, εκτός των άλλων, είναι δύο nodes του VMware οι οποίοι είναι υπεύθυνοι για την εκτέλεση και παροχή υπηρεσιών DNS, email, LDAP, αυθεντικοποίησης των χρηστών του campus, identity provider, ηλεκτρονικό πρωτόκολλο, antispam και frontend. Επίσης, υπάρχει γενικός και hosted web server και MySQL server. Ουσιαστικά, όλες αυτές οι υπηρεσίες παρέχονται από αυτούς τους δύο severs μέσω εικονικών μηχανών. Κρατούνται αντίγραφα αυτών των εικονικών μηχανών στον άλλον node και υπάρχει η επιθυμία για μεταγωγή σε πιο ευέλικτο σύστημα.

4.5. Προσθήκη του Hypervisor στο CloudStack

Το CloudStack παρέχει την δυνατότητα να μπορούν να ενοποιηθούν ήδη υπάρχουσες εγκαταστάσεις με διαφορετικές τεχνολογίες virtualization. Με το CloudStack μπορούμε να έχουμε δηλαδή ένα ενοποιημένο σύστημα το οποίο μπορεί να έχει έναν host με Hypervisor το KVM και έναν ακόμα με Hypervisor το VMware. Ο κάθε Hypervisor θα ανήκει φυσικά σε διαφορετικό Cluster, αφού μιλάμε για διαφορές μεταξύ των Hypervisor hosts και ο καθένας θα συνεχίσει να έχει το δικό του πρωτεύοντα χώρο και θα μπορούν να συνεχίσουν να διαθέτουν τις υπηρεσίες τους όπως έκαναν και πριν την ενοποίηση των συστημάτων .

Στο προηγούμενο κεφάλαιο, είδαμε πώς μπορούμε να προσθέσουμε στο CloudStack έναν host με Hypervisor το KVM. Αντίστοιχα, μπορούμε να κάνουμε το ίδιο και για το VMware. Η διαδικασία αυτή είναι αρκετά πιο μεγάλη συγκριτικά με αυτήν για την εισαγωγή του host με το KVM, και πιο περίπλοκη ωστόσο πολύ ενδιαφέρουσα και με πολλές δυνατότητες εξέλιξης των ήδη υπαρχόντων συστημάτων. Κατά την γνώμη μου, το CloudStack και η ενοποίηση των διαφορετικών Hypervisors μέσω αυτού είναι ένα μεγάλο project από μόνο του και αποφασίστηκε να μην αναλυθεί περαιτέρω σε αυτή τη φάση, αλλά να είναι στα άμεσα μελλοντικά σχέδιά μας για τη διαχείριση των συστημάτων.

ΚΕΦΑΛΑΙΟ 5 – ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΧΡΗΣΕΙΣ

Σκοπός της εργασίας αυτής ήταν η οικειοποίηση με την τεχνολογία της εικονικοποίησης και τους τρόπους εφαρμογής της ώστε να μπορεί να εφαρμοστεί μία τεχνική πάνω στα ήδη υπάρχοντα συστήματα με απώτερο σκοπό την ενοποίησή τους σε συνεργασία με μία ανοιχτή τεχνολογία Cloud. Το ζητούμενο είναι να μπορεί να δημιουργηθεί ένα σταθερό και ευέλικτο Software Defined Data Center (SDDC) το οποίο θα παρέχει τις υπηρεσίες που χρειάζονται οι χρήστες του και θα μπορεί να είναι ασφαλές και redundant.

Πριν από λίγα χρόνια είχε γίνει πάλι μία προσπάθεια για να μπει το oVirt στην γραμμή παραγωγής και να είναι αυτό η τεχνολογία που θα προσφέρει τις υπηρεσίες της στους χρήστες του οργανισμού. Ωστόσο, τότε το oVirt ήταν ακόμα σε πρότερα στάδιά του και δεν μπορούσε να εφαρμοστεί self-hosted engine πάνω στον ίδιο server, κάτι που ήταν και ζητούμενο. Ο λόγος που εξετάσαμε τα KVM και VMware ήταν η ανάγκη για να ενοποιηθούν τα ήδη υπάρχοντα συστήματα κάτω από μία “ομπρέλα” που στην πορεία ανακαλύψαμε ότι θα μπορεί να είναι το CloudStack. Το VMware είναι ένα πολύ σταθερό προϊόν που προσφέρει λειτουργίες και υπηρεσίες ποιότητας. Ωστόσο κάθε τρία χρόνια λήγει το license του και πρέπει συνεχώς να αγοράζουμε το επόμενο. Επίσης στο VMware δεν είναι ορατά όλα του τα εξαρτήματα και πακέτα σε κάποιον που θέλει να γίνει developer, καθώς το VMware είναι ένα κλειστού τύπου λογισμικό. Οπότε, παρόλο που υπάρχει η δυνατότητα για επεκτάσεις και τροποποιήσεις, δεν αποφεύγεται το κλείδωμα στην συγκεκριμένη πλατφόρμα. Με το oVirt, παρουσιάζονται περισσότερες ευκαιρίες για να εμπλακούμε βαθιά με το devel του, ενώ στο Vmware, ως κλειστό λογισμικό, αυτές οι πλευρές παραμένουν κρυμμένες από το ευρύ κοινό. Για ένα εκπαιδευτικό ίδρυμα και για έναν εκπαιδευτικό και ερευνητικό οργανισμό, μερικές φορές προτιμάται η στροφή προς τις Open Source λύσεις, καθώς είναι διαθέσιμες οι γνώσεις ώστε να εφαρμοστεί μία Open Source λύση αλλά είναι και επιθυμητή η εμπλοκή και ο πειραματισμός που θα γίνουν με αυτήν και θα οδηγήσουν σε νέες γνώσεις που θα αποκτηθούν από αυτή τη διαδικασία.

Μελλοντικά, σκοπεύουμε να ενοποιήσουμε τα συστήματα που υπηρετούν μέχρι τώρα τις ανάγκες του Ε.Κ.Ε.Φ.Ε. Δημόκριτος. Το σύστημα που λειτουργεί σε VMware ESXi Hypervisor με δύο (2) nodes, αλλά και το σταθερό KVM, θα παραμείνουν ως έχουν, αλλά θα μπορούσαν να ενοποιηθούν τα συστήματα αυτά κάτω από το CloudStack.

Παράρτημα 1 – Gdeploy Configuration file

```
#gdeploy configuration generated by cockpit-gluster plugin
[hosts]
ov-node1.islab.local
ov-node2.islab.local
ov-node3.islab.local


[script1:ov-node1.islab.local]
action=execute
ignore_script_errors=no
file=/usr/share/gdeploy/scripts/grafton-sanity-check.sh -d sdb -h ov-
node1.islab.local,ov-node2.islab.local,ov-node3.islab.local


[script1:ov-node2.islab.local]
action=execute
ignore_script_errors=no
file=/usr/share/gdeploy/scripts/grafton-sanity-check.sh -d sdb -h ov-
node1.islab.local,ov-node2.islab.local,ov-node3.islab.local


[script1:ov-node3.islab.local]
action=execute
ignore_script_errors=no
file=/usr/share/gdeploy/scripts/grafton-sanity-check.sh -d sdb -h ov-
node1.islab.local,ov-node2.islab.local,ov-node3.islab.local


[disktype]
jbod


[diskcount]
1
```

```
[stripesize]
256

[service1]
action=enable
service=chronyd

[service2]
action=restart
service=chronyd

[shell2]
action=execute
command=vdsm-tool configure --force

[script3]
action=execute
file=/usr/share/gdeploy/scripts/blacklist_all_disks.sh
ignore_script_errors=no

[pv1:ov-node1.islab.local]
action=create
devices=sdb
ignore_pv_errors=no

[pv1:ov-node2.islab.local]
action=create
devices=sdb
ignore_pv_errors=no

[pv1:ov-node3.islab.local]
action=create
```

```
devices=sdb
ignore_pv_errors=no

[vg1:ov-node1.islab.local]
action=create
vgname=gluster_vg_sdb
pvname=sdb
ignore_vg_errors=no

[vg1:ov-node2.islab.local]
action=create
vgname=gluster_vg_sdb
pvname=sdb
ignore_vg_errors=no

[vg1:ov-node3.islab.local]
action=create
vgname=gluster_vg_sdb
pvname=sdb
ignore_vg_errors=no

[lv1:ov-node1.islab.local]
action=create
poolname=gluster_thinpool_sdb
ignore_lv_errors=no
vgname=gluster_vg_sdb
lvtype=thinpool
size=101GB
poolmetadatasize=1GB

[lv2:ov-node2.islab.local]
action=create
```

```
poolname=gluster_thinpool_sdb
ignore_lv_errors=no
vgname=gluster_vg_sdb
lvtype=thinpool
size=101GB
poolmetadatasize=1GB

[lv3:ov-node3.islab.local]
action=create
poolname=gluster_thinpool_sdb
ignore_lv_errors=no
vgname=gluster_vg_sdb
lvtype=thinpool
size=101GB
poolmetadatasize=1GB

[lv4:ov-node1.islab.local]
action=create
lvname=gluster_lv_engine
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/engine
size=80GB
lvtype=thick

[lv5:ov-node1.islab.local]
action=create
lvname=gluster_lv_data
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/data
lvtype=thinlv
```

```
poolname=gluster_thinpool_sdb
virtualsize=50GB

[lv6:ov-node1.islab.local]
action=create
lvname=gluster_lv_vmstore
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/vmstore
lvtype=thinlv
poolname=gluster_thinpool_sdb
virtualsize=50GB

[lv7:ov-node2.islab.local]
action=create
lvname=gluster_lv_engine
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/engine
size=80GB
lvtype=thick

[lv8:ov-node2.islab.local]
action=create
lvname=gluster_lv_data
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/data
lvtype=thinlv
poolname=gluster_thinpool_sdb
virtualsize=50GB
```

```
[lv9:ov-node2.islab.local]
action=create
lvname=gluster_lv_vmstore
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/vmstore
lvtype=thinlv
poolname=gluster_thinpool_sdb
virtualsize=50GB

[lv10:ov-node3.islab.local]
action=create
lvname=gluster_lv_engine
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/engine
size=80GB
lvtype=thick

[lv11:ov-node3.islab.local]
action=create
lvname=gluster_lv_data
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/data
lvtype=thinlv
poolname=gluster_thinpool_sdb
virtualsize=50GB

[lv12:ov-node3.islab.local]
action=create
lvname=gluster_lv_vmstore
```

```
ignore_lv_errors=no
vgname=gluster_vg_sdb
mount=/gluster_bricks/vmstore
lvtype=thinlv
poolname=gluster_thinpool_sdb
virtualsize=50GB

[selinux]
yes

[service3]
action=restart
service=glusterd
slice_setup=yes

[firewalld]
action=add
ports=111/tcp,2049/tcp,54321/tcp,5900/tcp,5900-
6923/tcp,5666/tcp,16514/tcp,54322/tcp
services=glusterfs

[script2]
action=execute
file=/usr/share/gdeploy/scripts/disable-gluster-hooks.sh

[shell3]
action=execute
command=usermod -a -G gluster qemu

[volume1]
action=create
volname=engine
```

```
transport=tcp

replica=yes

replica_count=3

key=group,storage.owner-uid,storage.owner-gid,network.ping-
timeout,performance.strict-o-direct,network.remote-dio,cluster.granular-
entry-heal

value=virt,36,36,30,on,off,enable

brick_dirs=ov-node1.islab.local:/gluster_bricks/engine/engine,ov-
node2.islab.local:/gluster_bricks/engine/engine,ov-
node3.islab.local:/gluster_bricks/engine/engine

ignore_volume_errors=no


[volume2]

action=create

volname=data

transport=tcp

replica=yes

replica_count=3

key=group,storage.owner-uid,storage.owner-gid,network.ping-
timeout,performance.strict-o-direct,network.remote-dio,cluster.granular-
entry-heal

value=virt,36,36,30,on,off,enable

brick_dirs=ov-node1.islab.local:/gluster_bricks/data/data,ov-
node2.islab.local:/gluster_bricks/data/data,ov-
node3.islab.local:/gluster_bricks/data/data

ignore_volume_errors=no

arbiter_count=1


[volume3]

action=create

volname=vmstore

transport=tcp

replica=yes

replica_count=3
```

```
key=group,storage.owner-uid,storage.owner-gid,network.ping-
timeout,performance.strict-o-direct,network.remote-dio,cluster.granular-
entry-heal

value=virt,36,36,30,on,off,enable

brick_dirs=ov-node1.islab.local:/gluster_bricks/vmstore/vmstore,ov-
node2.islab.local:/gluster_bricks/vmstore/vmstore,ov-
node3.islab.local:/gluster_bricks/vmstore/vmstore

ignore_volume_errors=no

arbiter_count=1
```

Clusters

Το Cluster είναι μία λογική ομάδα από hosts που έχουν κοινό αποθηκευτικό χώρο και ίδιου τύπου CPU (είτε Intel ή AMD). Αν οι hosts έχουν διαφορετικής γενιάς CPU, τότε μπορούν να χρησιμοποιήσουν μόνο τα χαρακτηριστικά που είναι κοινά ανάμεσα στις γενιές.

Κάθε Cluster πρέπει να ανήκει σε ένα Data Center και κάθε host σε αυτό το σύστημα πρέπει να ανήκει σε ένα Cluster. Οι εικονικές μηχανές διανέμονται δυναμικά στους hosts που βρίσκονται στο Cluster και μπορούν να μεταναστεύσουν από τον έναν host στον άλλον, ανάλογα με τις διάφορες πολιτικές και κανονισμούς που εφαρμόζονται στο Cluster στην καρτέλα *Clusters* και στο *Configuration tool* κατά τη διάρκεια της εκτέλεσής του. Το Cluster είναι το υψηλότερο επίπεδο στο οποίο μπορεί να οριστεί η ενέργεια και οι πολιτικές κατανομής φορτίων.

Τα Cluster εκτελούν είτε εικονικές μηχανές ή τους Gluster Storage Servers. Ένα Cluster δεν μπορεί να υποστηρίξει το Virtualization και τους Storage hosts ταυτόχρονα.

https://www.ovirt.org/documentation/admin-guide/chap-Clusters/

GlusterFS (Gluster File System)

Το GlusterFS είναι ένας τύπος κατανεμημένου συστήματος αρχείων δικτύου, ή αλλιώς ένας τύπος Distributed Network File System (DNFS), με το οποίο, όπως και με οποιοδήποτε άλλον τύπο DNFS, μπορούμε να δημιουργήσουμε ένα σύστημα αρχείων που μπορεί να επεκταθεί και να κατανεμηθεί σε άλλα συστήματα, κάτω από το ίδιο global namespace.

Το GlusterFS είναι ένα ανοιχτού τύπου λογισμικό και μπορεί να συγκεντρώνει τους αποθηκευτικούς χώρους από διαφορετικούς server και να τους διαθέτει κάτω από το ίδιο global namespace, βοηθώντας έτσι στη δημιουργία χώρων που μπορούν να χρησιμοποιηθούν για παροχή αποθηκευτικού χώρου νέφους (Cloud) αλλά και για streaming πολυμέσων για τους χρήστες μας. Επίσης, με το GlusterFS μπορούμε να κάνουμε χρήση απλού υλικού (hardware) και όχι εξειδικευμένου εξοπλισμού που συνήθως κοστίζει ακριβά.

https://searchstorage.techtarget.com/definition/GlusterFS-Gluster-File-System

Network File System (NFS)

Το Network File System (NFS) είναι ένα κατανεμημένο σύστημα αρχείων που επιτρέπει στους χρήστες που βρίσκονται στα τερματικά τους να έχουν πρόσβαση στα αρχεία τους μέσω του τοπικού δικτύου του οργανισμού και βρίσκονται αποθηκευμένα σε χώρους που μοιάζουν με τον τοπικό αποθηκευτικό χώρο των υπολογιστών. Το NFS, όπως και πολλά άλλα πρωτόκολλα, χρησιμοποιεί το Open Network Computing Remote Procedure Call (ONC RPC) σύστημα. Είναι ανοιχτού τύπου standard και ορίζεται στο Request for Comments (RFC), επιτρέποντας σε οποιονδήποτε να το επεξεργαστεί και αν θέλει να κάνει κάποιες τοπικές αλλαγές στο σύστημά του.

https://en.wikipedia.org/wiki/Network_File_System

Network address translation (NAT)

Το Network address translation (NAT) είναι μία μέθοδος ανακατεύθυνσης και μετάφρασης μίας IP διεύθυνσης σε μία άλλη, αλλάζοντας τις πληροφορίες του δικτύου που βρίσκονται αποθηκευμένες στο header του IP πλαισίου των πακέτων καθώς αυτά μεταφέρονται στο δίκτυο από κάποια συσκευή router.

Χάρη στο NAT η διευθυνσιοδότηση Ipv4 επέζησε τόσο πολύ, καθώς μία Ipv4διεύθυνση μπορεί να χρησιμοποιηθεί για όλα τα τερματικά που βρίσκονται σε ένα ιδιωτικό δίκτυο.

https://en.wikipedia.org/wiki/Network_address_translation

Templates

Το template είναι το αντίγραφο μίας ήδη υπάρχουσας εικονικής μηχανής το οποίο δημιουργούμε για να απλοποιήσουμε την δημιουργία πανομοιότυπων εικονικών μηχανών. Τα templates αποθηκεύουν τις ρυθμίσεις του λογισμικού, του υλικού καθώς και τις εφαρμογές που έχουν εγκατασταθεί στην εικονική μηχανή από την οποία δημιουργείται το template. Η εικονική μηχανή πάνω στην οποία είναι βασισμένο το template ονομάζεται πηγαία εικονική μηχανή ή αλλιώς source virtual machine.

https://www.ovirt.org/documentation/vmm-guide/chap-Templates/

Virtual Machine Pools

Μία “πισίνα” εικονικών μηχανών, ή αλλιώς virtual machine pool, είναι μία ομάδα εικονικών μηχανών οι οποίες είναι κλώνοι του ίδιου template και μπορούν να χρησιμοποιηθούν On-Demand από οποιονδήποτε χρήστη που βρίσκεται σε αυτή την ομάδα. Τα virtual machine pools επιτρέπουν στους διαχειριστές να επεξεργάζονται γρήγορα και αποδοτικά ένα σετ από γενικοποιημένες εικονικές μηχανές για τους χρήστες τους.

https://www.ovirt.org/documentation/admin-guide/chap-Pools/

Vdsm

Το Vdsm είναι ένας daemon που απαιτείται από κάποιον Virtualization Manager όπως είναι το oVirt-engine ή ο Red Hat Enterprise Virtualization Manager για να διαχειρίζεται τους Linux hosts και τους KVM virtual machine guests. Το Vdsm διαχειρίζεται και παρακολουθεί τον αποθηκευτικό χώρο του host, την μνήμη και τα δίκτυα. Επίσης είναι υπεύθυνος για τη δημιουργία των εικονικών μηχανών, άλλες εργασίες, συγκέντρωση στατιστικών στοιχείων και συλλογή των logs.

https://www.ovirt.org/develop/developer-guide/vdsm/vdsm/

LDAP (Lightweight Directory Access Protocol)

Το LDAP (Lightweight Directory Access Protocol) είναι ένα λογισμικό πρωτόκολλο και είναι μία πιο ελαφριά έκδοση του Directory Access Protocol (DAP), το οποίο είναι μέρος του X.500, ένα standard για υπηρεσιών ευρετηρίων σε ένα δίκτυο. Το LDAP είναι ελαφρύτερο επειδή στην αρχική του έκδοση δεν συμπεριλαμβάνονται χαρακτηριστικά ασφαλείας. Το LDAP δημιουργήθηκε στο University of Michigan και έχει υιοθετηθεί από τουλάχιστον 40 εταιρείες και οργανισμούς. Η Microsoft το συμπεριλαμβάνει ως μέρος του Active Directory της και είναι μέρος αρκετών από τα προγράμματά της όπως είναι το Outlook Express. Η Cisco επίσης το υποστηρίζει στα προϊόντα δικτύωσής της.

https://searchmobilecomputing.techtarget.com/definition/LDAP

ΒΙΒΛΙΟΓΡΑΦΙΑ

Apache Software Foundation, 2014, “Cloudstack Installation Documentation, Release 4.11.0.0”, Apache Software Foundation

De la Rosa Jose, 2014, “KVM Virtualization in RHEL 7 Made Easy”, Dell Inc.

Gluster Community, 2016, “GlusterFS Documentation”, Gluster Community

Gluster Inc, 2011, “Cloud Storage for the Modern Data Center”, Gluster Inc.

Keating Jesse, 2015, “Mastering Ansible”, Packt Publishing Ltd.

Membrey Peter, Verhoeven Tim, Angenendt Ralph, 2009, “The Definitive Guide to CentOS”, Apress

Matotek Dennis , Turnbull James, Lieverdink Peter, 2017, “Pro Linux System Administration”, Apress

Red Hat Enterprise Linux, 2018, “Virtualization Deployment and Administration Guide”, Red Hat Inc.

Red Hat Enterprise Linux, 2007, “Virtualization Guide: Red Hat Virtualization” Red Hat Inc.

[4]Vacca John R., 2017, “Cloud Computing Security: Foundations and Challenges”, CRC Press

[1]http://docs.ansible.com/ansible/modules_by_category.html(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

http://abregman.com/2015/12/25/ansible-write-and-run-your-first-playbook/ (Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[2]https://gdeploy.readthedocs.io/en/latest/ (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

[3]https://docs.gluster.org/en/v3/Administrator%20Guide/GlusterFS%20Introduction (Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[5]https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

[6]http://europa.eu/rapid/press-release_MEMO-12-713_el.htm(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[7]http://en.wikipedia.org/wiki/Cloud_computing(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[8]https://cloudstack.apache.org/(Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

[9]https://oVirt.org/documentation/self-hosted/Self-Hosted-EngineGuide (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

[10]https://oVirt.org/documentation/self-hosted/chap-Deploying_Self-Hosted_Engine (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

[11]https://oVirt.org/Node(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[12]https://resources.ovirt.org/pub/ovirt-4.2-snapshot/rpm/el7/noarch/ovirt-engine-appliance-4.2-20180617.1.el7.noarch.rpm(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[13]http://docs.cloudstack.apache.org/en/4.11.1.0/quickinstallationguide/qig.html(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

[14]https://www.idevnews.com/views/images/uploads/general/citrix_cloudstack_1000.jpg (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://oVirt.org/documentation/self-hosted/chap-Installing_Additional_Hosts_to_a_Self-Hosted_Engine

https://oVirt.org/documentation/install-guide/chap-oVirt_Nodes (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://www.ovirt.org/documentation/admin-guide/chap-Logical_Networks/ (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://www.ovirt.org/develop/developer-guide/engine/automatic-fencing/(Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://www.youtube.com/watch?v=Pasz-lv3gTY(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://bugzilla.redhat.com/show_bug.cgi?id=1463122 (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

http://wiki.debian.org/DebianSoftware.(Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://www.debian.org/(Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://www.debian.org/doc/FAQ(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://www.debian.org/doc/user-manuals#quick-reference(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://www.debian.org/doc/ddp(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://lists.debian.org(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://www.tldp.org(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://www.linux-kvm.org/page/Main_Page(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Virtualization/3.5/html/Administration_Guide/ (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

http://www.redhat.com/en/topics/virtualization/what-is-kvm.html(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

http://developers.redhat.com/blog/2016/08/18/setting-up-kvm-on-rhel(Τελευταία προσπέλαση 22 Σεπτεμβρίου 2018)

http://www.redhat.com/en/topics/virtualization# (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)

https://linode.com/docs/databases/mysql/how-to-install-mysql-on-centos-7/ (Τελευταία προσπέλαση 27 Σεπτεμβρίου 2018)