

Τ.Ε.Ι. ΠΕΙΡΑΙΑ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ



ΕΓΚΑΤΑΣΤΑΣΗ STORAGE AREA NETWORK
ΚΑΙ OCEANOS ΤΟΥ ΤΜΗΜΑΤΟΣ



Σπουδαστής: ΒΛΑΣΣΗΣ ΝΙΚΟΛΑΟΣ
Επιβλέπων Καθηγητής: ΓΕΩΡΓΙΟΣ ΔΙΔΙΝΤΑΣ
Απρίλης 2014

Περιεχόμενα

Περίληψη.....4

Abstract..... 4

Κεφάλαιο 1 : Διαφορές SAN-NAS

1.1 Πρωτόκολλο Fibre Channel.....6

1.1.1 FC0 (Φυσικό επίπεδο) 6

1.1.2 FC-1 (κωδικοποίηση, Ordered sets, Link control protocol) 8

1.1.3 FC-2: data transfer 9

1.1.4 FC-3: common services 10

1.1.5 FC-4 application protocols 11

Κεφάλαιο 2 : Διαθέσιμο Υλικό

2.1 Disk subsystems 13

2.2 JBOD (Just a Bunch Of Disks) 13

2.3 Intelligent disk subsystem 14

2.4 Enterprise Directory 140m 33

2.5 CX Storage System 35

Κεφάλαιο 3 : SYNNEFO

3.1 Astakos.....40

3.2 Authentication use cases..... 40

3.3 Pithos object storage service 42

3.4 Network Service (Cyclades) 46

3.5 Image Registry Service (Plankton) 46

3.6 OpenStack 49

3.7 OpenStack object storage system 50

3.8 Google Ganeti 56

Κεφάλαιο 4 : Υλοποίηση

4.1 Συμπεράσματα θεωρίας 69

4.2 Προτεινόμενες τοπολογίες για τη storage area network..... 69

Περίληψη

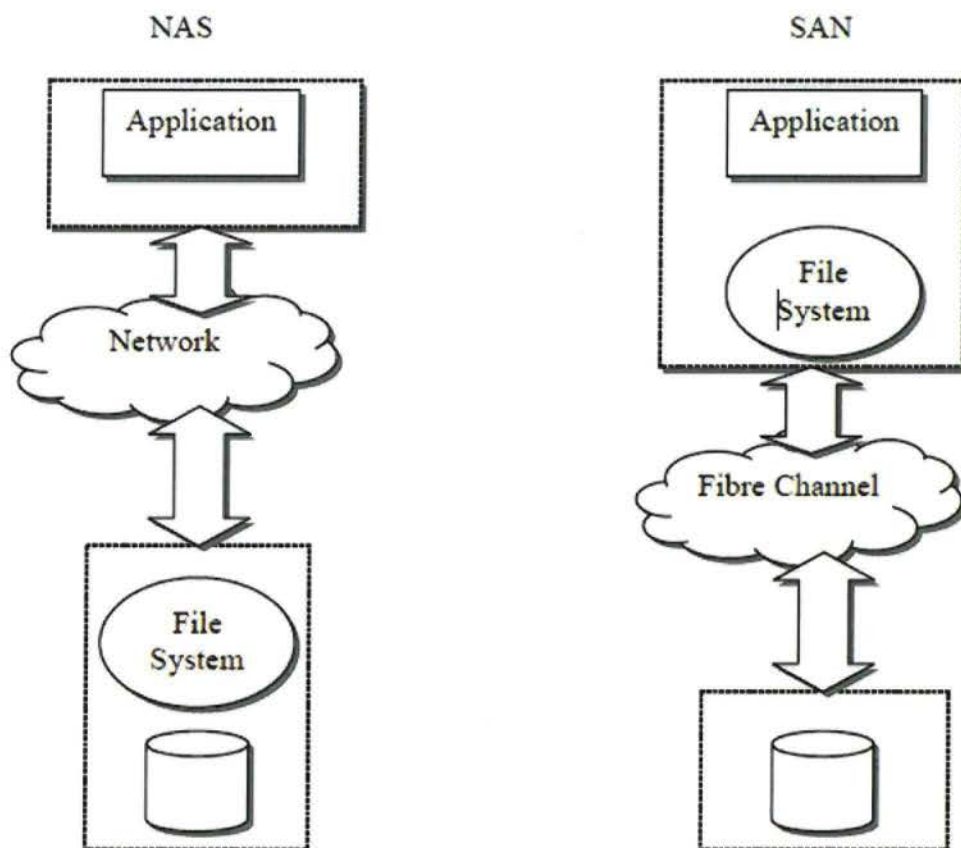
Το πρώτο κεφάλαιο αποτελείται από δύο μέρη. Στο πρώτο του μέρος αναλύονται τα *SAN* και παραθέτονται σε συγκριτική μελέτη με τα *NAS*. Στο δεύτερο μέρος του κεφαλαίου αναλύεται το πρωτόκολλο *FC* το οποίο αποτελείται από (*FC-0,FC-1,FC-2,FC-3,FC-4*) και παρατίθεται σε συγκριτική μελέτη με το *OSI protocol*. Το δεύτερο κεφάλαιο αναφέρεται στα βασικά κομμάτια του project από πλευράς υλικού. Στην αρχή αναλύονται τα *disk array* και *enterprise directory*, ενώ στην συνέχεια γίνεται μια εκτενής αναφορά σε *cx-700* και *ed-140m*. Πρακτικά αυτό σημαίνει πως αναφέρονται εκτενέστερα οι δυνατότητες των συγκεκριμένων μηχανημάτων και από πλευράς υλικού αλλά και λογισμικού. Στο τρίτο κεφάλαιο αναλύεται το project του *grnet* (*synnefo*) το οποίο αποτελείται από 5 μικρότερα project γραμμένα σε *python* (*Astakos, Pithos, Cyclades, Plakton, Archipelagos*). Αναλύονται πιο διεξοδικά η δομή τους καθώς και οι απαιτήσεις τους, οι οποίες είναι άρρηκτα συνδεδεμένες με το *google ganeti* και το *OpenStack Storage Api*, οπότε στο συγκεκριμένο κεφάλαιο γίνεται εκτενής αναφορά και στα συγκεκριμένα projects. Στο τέταρτο και τελευταίο κεφάλαιο περιγράφεται η μελέτη για τη δομή που πρέπει να έχει το σύστημά μας λογικά (τοπολογίες, πρωτόκολλα) καθώς και υλικά. Επίσης περιέχονται προτάσεις για το ποιες τεχνολογίες θα πρέπει να χρησιμοποιηθούν και πώς θα εξασφαλιστεί η λειτουργικότητα της εφαρμογής.

Abstract

The first chapter consists of two parts. In the first part *SAN* are elaborated and are presented in comparison with *NAS*. In the second part protocol *FC*, which consists of (*FC-0,FC-1,FC-2,FC-3,FC-4*), is analyzed and is compared with the *OSI protocol*. The second chapter refers to the essential parts of this project regarding hardware. Firstly, *disk array* and *enterprise directory* are analyzed. Consequently, there is an extensive reference to *cx-700* and *ed-140m*, which in fact means that the capabilities of that machinery are described in detail hardware- and software-wise. In the third chapter *grnet* project (*synnefo*), which is composed of five minor projects (*Astakos, Pithos, Cyclades, Plakton, Archipelagos*) that are written in *python*, is described. The structure and the requirements of those projects, that are essentially related to *google ganeti* and *OpenStack Storage Api* are analyzed in detail. Therefore, in this particular chapter there is an extensive reference to those projects. In the fourth (and last) chapter, the study about the structure of our system is described regarding logical design (topologies, protocols) and hardware. There are also suggestions regarding the technologies that should be used and ways in which the functionality of the application can be ensured are mentioned.

Κεφάλαιο 1- Διαφορές SAN-NAS

Για το SAN χρειαζόμαστε εξειδικευμένο υλικό, στην πραγματικότητα αυτό που χρειάζεται είναι σκληροί δίσκοι, μια τυπική αρχιτεκτονική υπολογιστή και ειδικό Λ.Σ. Στην πραγματικότητα λόγω ειδικού υλικού το κόστος για την υλοποίηση του SAN είναι μεγαλύτερο αφού πέρα από τα παραπάνω χρειάζεται και εξειδικευμένο δίκτυο αφού οι υπολογιστές συνδέονται μέσω αυτού στα αποθηκευτικά μέσα. Δυο παραδείγματα είναι το FC-SAN και το ISCSI. Επιγραμματικά να αναφέρω πώς στο πρώτο τα δεδομένα μεταφέρονται από δίκτυο οπτικών ινών, στην ουσία υπάρχει μία συσκευή που λέγεται host bus adapter και πρακτικά μετατρέπει τα SCSI αιτήματα σε FC πακέτα, ενώ το δεύτερο μεταφέρει τα SCSI αιτήματα μέσα από IP δίκτυο. Στο επίπεδο μεταφοράς δεδομένων τώρα στο NAS το file system βρίσκεται μέσα στη συσκευή και τα αιτήματα των υπολογιστών απαντώνται από την ίδια συσκευή, στο SAN οι δίσκοι απλά διαχειρίζονται τα δεδομένα σε επίπεδο block, πρακτικά αυτό σημαίνει πως οι hosts είναι επιφορτισμένοι με τη μεταφορά των αρχείων σε επίπεδο file system. Λόγω της κοινής χρήσης αρχείων από πολλούς σταθμούς γεννήθηκε η ανάγκη να δημιουργηθεί εξειδικευμένο file system .Οι διαφορές που έχουν οι δύο τεχνικές επηρεάζει άμεσα και την επεκτασιμότητά τους. Το NAS σαν client - server τεχνική που είναι κάτω από ορισμένες συνθήκες αναγκαστική αποτελεί bottleneck .Από την άλλη στα SAN οι ταχύτητες είναι μεγαλύτερες και ανάλογα με την τοπολογία και το μέσο που επιλέγουμε έχουμε τη δυνατότητα να χρησιμοποιήσουμε περισσότερους ή λιγότερους hosts. Όπως σε όλα τα υπολογιστικά συστήματα έτσι και εδώ οι τεχνικές caching είναι αρκετά χρήσιμες για την βέλτιστη απόδοση του συστήματος. Στο NAS το caching χωρίζεται σε δύο μέρη (server side, client side). Στην πλευρά του διακομιστή το caching γίνεται όπως σε κάθε file system , κάθε φορά που διαβάζει κάποιο αρχείο, τα δεδομένα τα γράφει στην cache οπότε αν ξαναζητηθούν μπορεί να τα διαβάσει απευθείας από εκεί .Στην πλευρά του client τώρα εξαρτάται από την δυνατότητα του client να αποθηκεύει τα δεδομένα που χρησιμοποιεί επανειλημμένα στην cache ούτως ώστε να μην χρειάζεται να τα ζητά πάλι από τους δίσκους. Στο SAN το caching εξαρτάται μόνο από το file system που χρησιμοποιούμε και η επιλογή είναι άρρηκτα συνδεδεμένη με τις δυνατότητες που έχουν οι hosts καθώς και αν έχουν την δυνατότητα να έχουν πρόσβαση σε κοινά μέσα.



1.1 Πρωτόκολλο Fibre Channel

Το συγκεκριμένο πρωτόκολλο χρησιμοποιείται επί το πλείστον για δίκτυα που συνδέουν αποθηκευτικά μέσα με υπολογιστές, πρόκειται για το βασικό πρωτόκολλο με το οποίο υλοποιούνται τα Storage area network. Οι βασικές τοπολογίες του είναι τρεις: point-to-point, αστέρα και Arbitrary Loop. Όπως και σε κάθε υπολογιστικό σύστημα έτσι και εδώ η επιλογή της τοπολογίας είναι συνάρτηση του μελλοντικού φόρτου, του διαθέσιμου χρηματικού ποσού. Στη δική μας περίπτωση τα μηχανήματα για την δημιουργία της πιο αξιόπιστης επιλογής υπάρχουν. Το FC πρωτόκολλο αποτελείται από 5 layers (FC-0, FC-1, FC-2, FC-3, FC-4)

1.1.1 FC0 (Φυσικό επίπεδο)

Ορίζει το φυσικό μέσο μετάδοσης καθώς και ποιες τιμές μεταφράζονται σε λογικό "0" και "1" αντίστοιχα. Σε αντίθεση με το SCSI τα δεδομένα μεταφέροντα σειριακά πράγμα που σημαίνει πως το clock rate μπορεί να φτάσει μέχρι ένα σημείο. Η μετάδοση είναι full duplex πράγμα που σημαίνει πως έχουμε 200 MB/s και προς

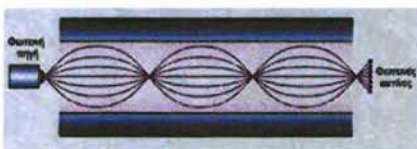
τις δύο κατευθύνσεις. Επίσης το bit error είναι 1 κάθε 10^{12} bits.

Τύποι οπτικών ινών:

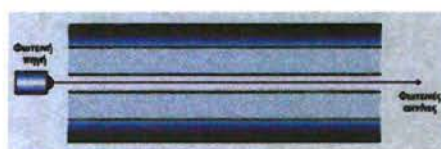
1. Μονότροπες(single-mode) οπτικές ίνες :
2. Πολύτροπες (multi-mode) οπτικές ίνες.

Έχουν διαφορετικό μήκος κύματος, οι πολύτροπες διαφορετικές δέσμες φωτός που διαθλώνται σε διαφορετική γωνία μπορούν να ταξιδεύουν ταυτόχρονα μέσα στο κανάλι, επίσης η μονότροπη μπορεί να επιλεγεί για μεγαλύτερες αποστάσεις. [1]

Πολύτροπη οπτική ίνα



Μονότροπη οπτική ίνα



Τύποι connector:

1. FC connector
2. E2000 connector
3. ESCON connector
4. LC connector
5. LuxCis connector
6. MIC (FDDI) connector
7. MPO connector
8. MT-RJ connector
9. SC connector
10. SMA 905 connectors
11. ST connector
12. TOSLINK connector

Στην ουσία είναι το τερματικό του καλωδίου , η συσκευή που το συνδέει με τον πομπό και τον δέκτη.[2]



LC connector (duplex) SC connector (duplex)

Πλεονεκτήματα οπτικών ινών:

1. Χαμηλό κόστος για καλώδιο ίδιας απόστασης και δυνατοτήτων
2. Υψηλό bandwidth
3. Μικρή εξασθένιση του σήματος χάρη της ποιότητας του μέσου μετάδοσης
4. Μικρές απαιτήσεις σε ενέργεια
5. Αμιγώς ψηφιακό σήμα
6. Υψηλή διαθεσιμότητα
7. Μικρές διαστάσεις και βάρος

Το φυσικό επίπεδο στο μοντέλο OSI είναι το χαμηλότερο και αυτό που καθορίζει το φυσικό μέσο μετάδοσης τα βύσματα καθώς και την κωδικοποίηση των σημάτων (ποιες τιμές αντιπροσωπεύουν "0" και "1" αντίστοιχα). Όπως καταλαβαίνουμε από τα παραπάνω το FC-0 αντιστοιχίζεται απόλυτα με το φυσικό επίπεδο του OSI. Με τη θεμελιακή διαφορά πως στο ένα μιλάμε για μετάδοση δεδομένων μέσω οπτικής ίνας ενώ στο OSI το μέσο μετάδοσης είναι ασαφές αφού το συγκεκριμένο μοντέλο χρησιμοποιείται για ασύρματα και ενσύρματα δίκτυα.

1.1.2 FC-1 (κωδικοποίηση, Ordered sets, Link control protocol)

Το συγκεκριμένο επίπεδο καθορίζει την κωδικοποίηση των δεδομένων πριν τη μετάδοση στο φυσικό μέσο. Επίσης καθορίζει ορισμένες λέξεις που είναι χρήσιμες για τη διαχείριση της σύνδεσης.

8B/10B encoding: Είναι παρόμοιο με το 4B/5B το οποίο κρίθηκε ακατάλληλο λόγω έλλειψης DC balance (μπορεί να προκαλέσει υπερθέρμανση στο laser με δυσάρεστες συνέπειες). Ένα παράδειγμα για να καταλάβουμε πώς λειτουργεί η συγκεκριμένη κωδικοποίηση: Παίρνουμε τον αριθμό 45 ο οποίος στο δυαδικό σύστημα είναι 01000101 και για να συνεχίσουμε τη διαδικασία θα πρέπει να βρούμε το $RD = (\text{number of 1s}) - (\text{number of 0s}) = 3 - 5$ είναι φανερό πως το μηδενικά είναι περισσότερα οπότε για να ισοσταθμίσουμε το σήμα μας πρέπει να του προσθέσουμε άσσους. Άρα το αποτέλεσμα είναι

E	D	C	B	A	H	G	F
0	0	1	0	1	0	1	0

A	B	C	D	E	F	G	H
a	b	c	d	e	i	f	g
1	0	1	0	0	1	0	1

Ordered sets

Στο FC protocol υπάρχουν δύο τύποι λέξεων, τα δεδομένα που βρίσκονται ανάμεσα σε SOF(START OF FRAME) και EOF(END OF FRAME) αποτελούνται από 10 bits και είναι η ωφέλιμη πληροφορία.

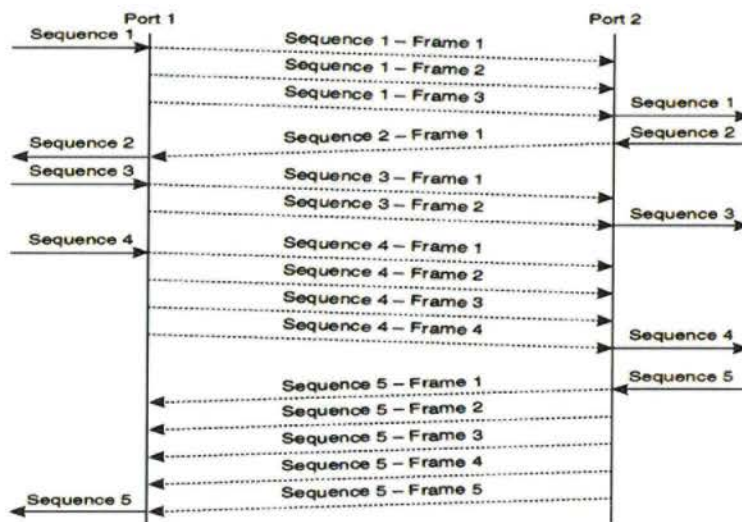
Από εκεί και πέρα υπάρχουν και το Ordered sets που αποτελούνται από 40 bits και βρίσκονται ανάμεσα στο EOF(END OF FRAME) και O F(START OF FRAME), έχουν ένα ειδικό χαρακτήρα ο οποίος δείχνει στον δέκτη πως είναι Ordered sets ούτως ώστε αυτός με την σειρά του να διαβάσει τα επόμενα 40 bits.

Link control protocol

Με την βοήθεια του Ordered sets το FC-1 ορίζει κάποια πρωτόκολλα στο επίπεδο σύνδεσης. (π.χ.arbitrated loop). Osi data link O κύριος στόχος του συγκεκριμένου επιπέδου είναι η μετατροπή του φυσικού σήματος σε πακέτο έτοιμο προς μετάδοση, επίσης εδώ βρίσκονται οι διάφοροι μέθοδοι κωδικοποίησης υπάρχουν πρωτόκολλα που ακούν στο μοντέλο osi και χρησιμοποιούν 8b/10b ή ακόμα και μεθόδους κωδικοποίησης με παρόμοια βάση. (gigabit Ethernet). Οπότε το συμπέρασμα που βγαίνει είναι πως τα 2 επίπεδα συμπίπτουν απόλυτα.

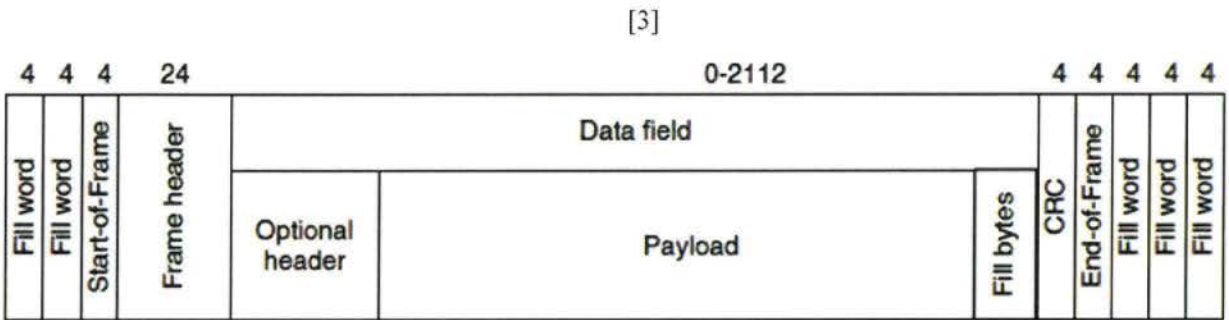
1.1.3 FC-2: data transfer

Ορίζει τον τρόπο με τον οποίο είναι οργανωμένα τα δεδομένα που είναι προς μετάδοση ,επίσης καθορίζει το ρυθμό μετάδοσης ,με τον οποίο στέλνει ο πομπός ,έτσι ώστε να μπορεί ο δέκτης να συγχρονιστεί και να διαβάσει τα δεδομένα. Το FC-2 επίπεδο χρησιμοποιεί δένδροειδή δομή για τη μεταφορά των δεδομένων, στο ανώτερο επίπεδο δημιουργεί μια λογική σύνδεση ανάμεσα σε δύο τερματικές συσκευές.



[Πηγή 3]

Ένα sequence είναι η μεγαλύτερη μονάδα δεδομένων και μεταδίδεται σειριακά, το FC-2 εγγυάται ότι τα δεδομένα φτάνουν με τη σειρά που έφυγαν. Επίσης για να προχωρήσουν τα δεδομένα στο πιο πάνω επίπεδο του πρωτοκόλλου θα πρέπει να φτάσουν με επιτυχία όλα τα sequence. Ένα FC frame μπορεί να περιέχει χρήσιμα δεδομένα ή δεδομένα ελέγχου τα οποία μπορούν να περιέχουν 2.112 bytes χρήσιμης πληροφορίας. Ένα frame του πρωτοκόλλου αυτού αποτελείται από header, useful data, Checksum (crc), SOF και EOF τέλος υπάρχουν επίσης 6 λέξεις που καθορίζουν το διάστημα ανάμεσα σε δύο frames.



- To frame header περιέχει τα ακόλουθα στοιχεία:
1. To Rout control που αποτελείται από δύο πεδία.
 2. To routing και το πεδίο πληροφόρησης, το routing bits προσδιορίζει την δρομολόγηση του πακέτου και τη χρήση του π.χ. R_CTL = 0X0N και n = 0 7 ξέρουμε πως το frame ανήκει στο FC-4 data device
 3. To information field προσδιορίζει επιπλέον πληροφορίες για τον τύπο του frame και εξαρτάται από το routing bits.
 4. Το id του host του παραλήπτη, το id του αποστολέα, το id του sequence καθώς και το Exchange ID
 5. Frame type το πεδίο αυτό διατηρείται εκτός αν είναι f_bsy. σε αυτήν την περίπτωση αναγνωρίζεται ο λόγος που είναι απασχολημένο το fabric και το πακέτο στέλνεται εκ νέου, f_ctl περιέχει πληροφορίες σχετικά με τον έλεγχο πλαισίου αναγνώρισης Exchange control, sequence control, Acknowledgment, sequence id είναι το πεδίο που δημιουργεί την ακολουθία
 6. Data filed control είναι το πεδίο εκείνο που καθορίζει αν υπάρχουν προαιρετικές κεφαλίδες κατά την έναρξη των δεδομένων
 7. Sequence count είναι το πεδίο που δείχνει τη θέση του frame μέσα στο sequence
 8. Original exchange id ορίζει στο frame πως είναι μέρος μίας συνομιλίας.
 9. Responder exchange είναι πεδίο που αναγνωρίζει frame ειδικής ανταλλαγής.
 10. Offset που δείχνει το πρώτο byte δεδομένων.

Service classes

To FC standard ορίζει 3 classes για την ανταλλαγή δεδομένων ανάμεσα σε 2 τερματικές συσκευές. Το Class 1 δημιουργεί μια σύνδεση πριν την αποστολή του frame, στη συνέχεια όλα τα frame ακολουθούν την ίδια δρομολόγηση. Τα Class 2 και Class 3 από την άλλη είναι packet-oriented services, τα πακέτα απλά δρομολογούνται μέσα από το δίκτυο, το πλεονέκτημα εδώ είναι πως η πόρτα μπορεί να μεταδώσει και άλλα πακέτα κατά τη διάρκεια της μετάδοσης.

Επίπεδο δικτύου

Στο μοντέλο OSI είναι το επίπεδο που καθορίζει τον τρόπο δρομολόγησης των πακέτων. Υπάρχουν 2 βασικές επιλογές, στατικά με πίνακες που δεν αλλάζουν αρκετά συχνά και δυναμικά στοιχεία που αλλάζουν κάθε φορά που συνδέεται κάποιος στο δίκτυο. Όπως καταλαβαίνουμε από τα παραπάνω, τα δύο αυτά επίπεδα εκτελούν σχεδόν ίδιες διεργασίες με της ιδιαιτερότητες των πρωτοκόλλων να μην αφήνουν ανεπηρέαστες τις λειτουργίες αυτές.

1.1.4 FC-3: common services

Είναι επιφορτισμένο με το να βρει τα μονοπάτια και να επιλέξει αυτό από το οποίο θα σταλούν τα δεδομένα, συμπίεζει τα δεδομένα για να σταλούν, τα κωδικοποιεί και κάνει mirroring τους δίσκους. Μερικές από τις υπηρεσίες που κάνει είναι οι εξής:

LOGIN: Δύο πόρτες θέλουν να επικοινωνήσουν και για να μπορέσει αυτό να είναι εφικτό θα πρέπει να δημιουργηθεί ένα session ανάμεσα στις δύο πόρτες. *Fabric login (FLOGI)* είναι η διαδικασία κατά την οποία μια N_PORT επικοινωνεί με μία F_PORT χρησιμοποιείται σε τοπολογίες αστέρα όπου υπάρχει ένα fabric, N_PORT LOGIN είναι η διαδικασία κατά την οποία η F_PORT δημιουργεί session με την N_PORT ούτως ώστε να της μεταδώσει τα δεδομένα που πήρε από την άλλη N_PORT, Process login (PRLI) θεωρείται οποιαδήποτε διαδικασία κατά την οποία εκτελείται κάποια εφαρμογή του παραπάνω επιπέδου (FC-4).

Addressing

Έχουμε δύο τύπου διευθύνσεις τις μοναδικές ανά τον κόσμο που ονομάζονται WWNs και εκείνες που μπορεί να υπάρχουν και αλλού και ονομάζονται (ids), κάθε πόρτα έχει το δικό της WorldWide Port Names (WWPNs) ενώ ανήκει και σε κάποιο WorldWide Node Names (WWNNs). Όπως καταλαβαίνουμε και από τα ονόματα κάθε πόρτα έχει το δικό της χαρακτηριστικό και σε ένα node μπορούμε να έχουμε αρκετές πόρτες, ένα τέτοιο παράδειγμα είναι ένα disk array το οποίο μπορεί να έχει παραπάνω από μία πόρτα.

Fabric services: name server and co

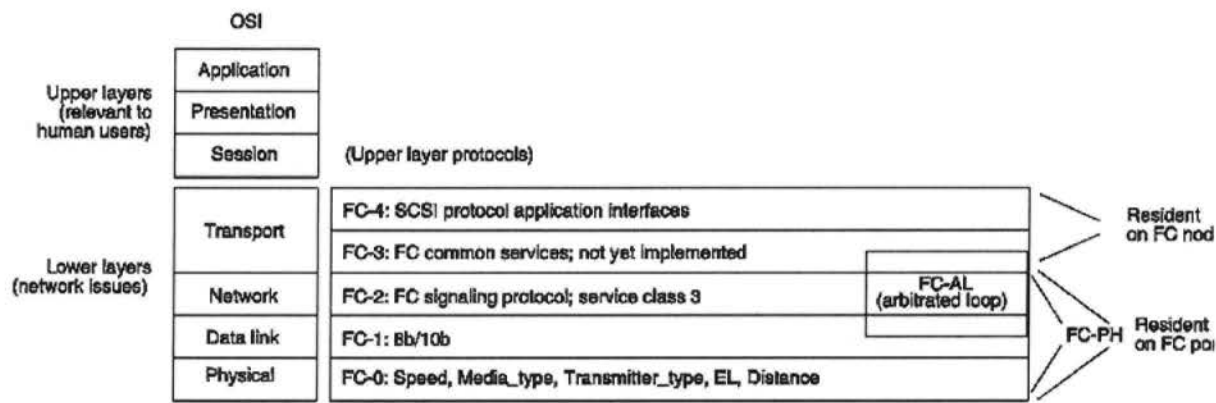
Στην τοπολογία αστέρα το switch έχει μερικές υπηρεσίες για τη δρομολόγηση των δεδομένων οι οποίες αποκαλούνται Fabric services. Πρακτικά η διεργασία με την οποία είναι επιφορτισμένο είναι να αντιστοιχίζει τοις N_PORTS με αντίστοιχα ID.

1.1.5 FC-4 application protocols

Στο συγκεκριμένο επίπεδο έχουμε τις εφαρμογές που χρησιμοποιεί το SAN για να διατελέσει οποιαδήποτε λειτουργία αυτό επιθυμεί, το κλασικότερο παράδειγμα είναι το SCSI πρωτόκολλο το οποίο χρησιμοποιείται για τη διαχείριση των απομακρυσμένων δίσκων, στην ουσία από την πλευρά του host, το host bus adapter είναι αυτό που μετατρέπει τα SCSI αιτήματα σε fibre channel πακέτα .Άλλη μία εφαρμογή είναι τον IPFC πρακτικά είναι IP δίκτυα μέσα από το FC, στην ουσία αυτό που κάνει το IPFC είναι να ορίζει πως τα IP πακέτα θα μετατραπούν σε FC, όπως κάθε πρωτόκολλο σε αυτό το επίπεδο αναγνωρίζεται σαν device driver .

Επίπεδο μεταφοράς

Το επίπεδο μεταφοράς είναι εκείνο που παραλαμβάνει τα δεδομένα από το επίπεδο συνόδου και στην συνέχεια τα χωρίζει σε πακέτα, στο FC πρωτόκολλο μπορεί να μην έχουμε επίπεδο συνόδου τα fc-3 και fc-4 όμως αναλαμβάνουν τη δημιουργία των πακέτων στη μορφή που θα πρέπει να είναι ανάλογα με τη λειτουργία που διατελούν, τα πακέτα δημιουργούνται από το fc-3 επίπεδο ενώ το fc-4 αναλαμβάνει την αναγνώριση των εφαρμογών που θέλουμε να λειτουργήσουμε μέσα από το fibre channel protocol.



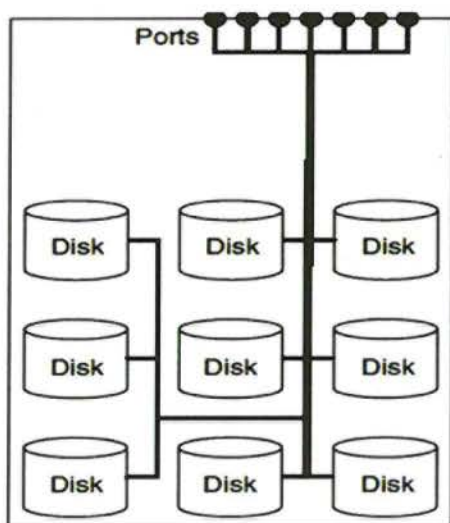
Κεφάλαιο 2 - Διαθέσιμο υλικό

2.1 Disk subsystems

Τα disk subsystem καλύπτουν τις αυξημένες ανάγκες για κοινά αποθηκευτικά μέσα σε ένα δίκτυο. Αυτό μπορεί να έχει πολλές εφαρμογές μερικά παραδείγματα είναι: cloud από server που διαμοιράζονται τον φόρτο εργασίας παρόλα αυτά είναι ανάγκη η πρόσβαση στα ίδια δεδομένα και ένα άλλο παράδειγμα θα μπορούσε να είναι η κεντρική διαχείριση των δεδομένων μιας εταιρίας. Υπάρχουν 2 βασικές κατηγορίες τα JBOD και τα intelligent disk subsystem . Η βασική τους διαφορά είναι η ύπαρξη ή όχι controller αφού στα JBOD (JUST A BUNCH OF DISKS) αυτό δεν υφίσταται .

2.2 JBOD (JUST A BUNCH OF DISKS)

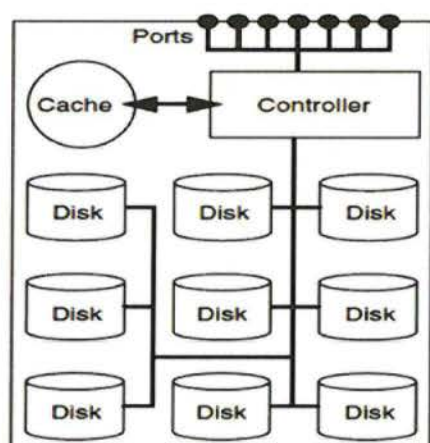
Όπως αναφέραμε και πιο πάνω η βασική διαφορά είναι στην ύπαρξη ή όχι controller είτε αυτός είναι raid controller η intelligent controller . Σε αυτήν την κατηγορία υπάρχει απλά ένας χώρος από δίσκους . Τα τυπικά JBOD έχουν από 8 έως 16 δίσκους και για κάθε δίσκο χρειάζεται και διαφορετική διεύθυνση και ο κάθε host που βλέπει αυτά τα μηχανήματα αναγνωρίζει κάθε δίσκο σαν ξεχωριστή συσκευή. Είναι εύκολα κατανοητό λοιπόν πως την οποιαδήποτε διαχείριση των δίσκων την κάνει ο host, που τους χρησιμοποιεί πρακτικά, αυτό σημαίνει πως, αν θέλουμε τεχνικές storage virtualization αυτό θα πρέπει να γίνει μέσω κάποιου οδηγού σε υπολογιστή και αυτό σε ορισμένα πρωτόκολλα όπως SCSI και FC μπορεί να οδηγήσει σε bottleneck διευθύνσεων.



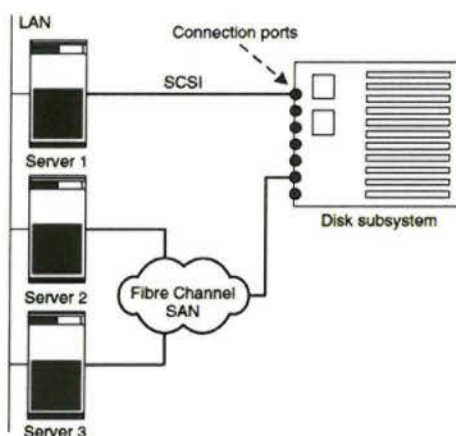
2.3 Intelligent disk subsystem

Ο συγκεκριμένος τύπος όπως προαναφέραμε διαθέτει controller που μπορούν να εμφανίζονται προς τους server ως ένας ενιαίος αποθηκευτικός χώρος ή και παραπάνω (δημιουργία Lun) αφού η ύπαρξη controller επιτρέπει την αποτελεσματική διαχείριση των δίσκων. Επιπλέον το storage virtualization πλέον πέφτει από πλευράς διαχείρισης στον controller . Για τη σύνδεση των συστημάτων αυτών χρησιμοποιούνται συγκεκριμένα πρωτόκολλα τα πιο γνωστά που χρησιμοποιούνται και περισσότερο είναι scsi , fibre channel και iscsi .

Scsi: Είναι ένα πρωτόκολλο μεταφοράς που ορίζει τον τρόπο με τον οποίο ο υπολογιστής θα επικοινωνήσει με αποθηκευτικά μέσα που το χρησιμοποιούν, πέρα από την χρήση σαν πρωτόκολλο για τον εαυτό του το χρησιμοποιούν και τα άλλα δυο που προαναφέραμε αφού για παράδειγμα στο fibre channel ο host bus adapter αυτό που κάνει είναι να μετατρέπει τα scsi αιτήματα σε πακέτα του FC και το iscsi είναι η χρήση του ίδιου του scsi μέσα από τα κοινά δίκτυα ip.



Intelligent disk subsystem



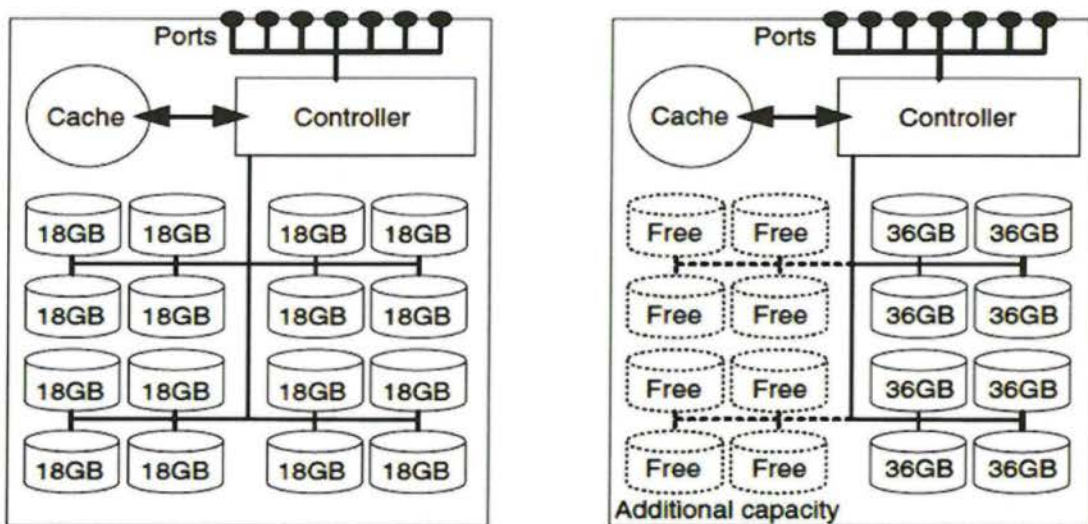
Συνδεσμολογία απλών san(scsi,fc)

Τα Intelligent disk subsystem έχουν διαφορετικά μεγέθη. Ένα μικρό υποσύστημα αυτού του είδους μπορεί να έχει δυο θύρες 6-8 σκληρούς δίσκους και ανάλογα με τους δίσκους μερικά terabyte. Μεγάλα συστήματα έχουν 10 συνδέσεις και οι servers μπορούν να τους βλέπουν μέσα από storage networks και μπορούν να αποθηκεύσουν πάνω από ένα petabyte από δεδομένα

Σκληροί δίσκοι και εσωτερική διάταξη

Σε φυσικό επίπεδο αυτός που αποθηκεύει τα δεδομένα στους σκληρούς δίσκους είναι ο controller και αυτό πρακτικά γίνεται αφού μόνο αυτός ξέρει φυσικά ποιοι δίσκοι αντιστοιχούν σε ποιο lun . Η χωρητικότητα του υποσυστήματος εξαρτάται από δύο παράγοντες: Από τον αριθμό των σκληρών που έχουμε στο υποσύστημά μας καθώς και από τη χωρητικότητα τους . Παίζει σημαντικό ρόλο στις αποδόσεις του συστήματος η επιλογή σκληρών δίσκων αφού όσο περισσότερους δίσκους έχουμε τόσο ο φόρτος εγγραφής και ανάγνωσης μοιράζεται όλο και σε περισσότερες κεφαλές με αποτέλεσμα να είναι πιο αποδοτικό το σύστημα. Βέβαια στις περισσότερες εφαρμογές δεν είναι αναγκαστικό μέτρο είναι μόνο σε ειδικές περιπτώσεις.

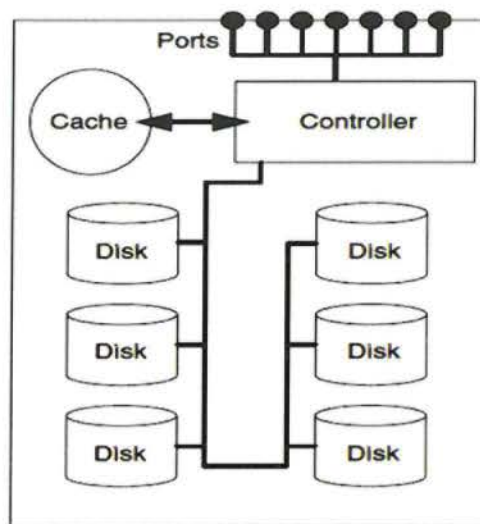
[3]



Ένας από τους σημαντικότερους παράγοντες για τις επιδόσεις στη μνήμη είναι ο τρόπος με τον οποίο συνδέονται με τον επεξεργαστή που στην δική μας περίπτωση είναι ο controller .

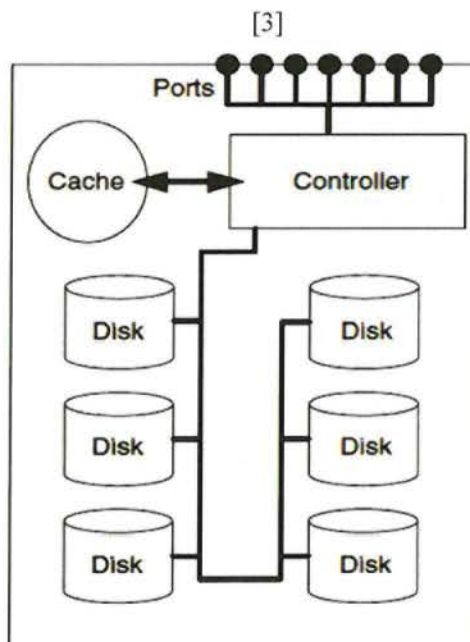
δίσκων είναι τα εξής :

Active: εδώ οι δίσκοι είναι συνδεδεμένοι με ένα μόνο καλώδιο . Όπως καταλαβαίνουμε αν αυτό το καλώδιο πέσει τότε δεν είναι δυνατή η πρόσβαση στα δεδομένα.



[3]

Active/passive: Εδώ τα εσωτερικά κανάλια που συνδέουν τον δίσκους είναι δύο. Κάτω από φυσιολογικές συνθήκες χρησιμοποιείται ο πρώτος και μόνο αν πέσει χρησιμοποιείται ο δεύτερος.



[3]

CACHING

Σε όλα τα υπολογιστικά συστήματα η μνήμη cache χρησιμοποιείται για την βελτιστοποίηση της απόδοσης του συστήματος σε επίπεδο ταχύτητας. Π.Χ. Η μνήμη cache του υπολογιστή βελτιώνει την ταχύτητα του

[16]

επεξεργαστή αφού είναι γρηγορότερη της ram και τα δεδομένα που χρησιμοποιούνται από την cpu συχνότερα κρατιούνται εκεί πέρα . Στα storage subsystem η μνήμη cache χρησιμοποιείται για την βέλτιστη ανάγνωση και εγγραφή στους δίσκους και χωρίζεται σε δύο βασικές κατηγορίες την cache των σκληρών δίσκων και την cache του controller .

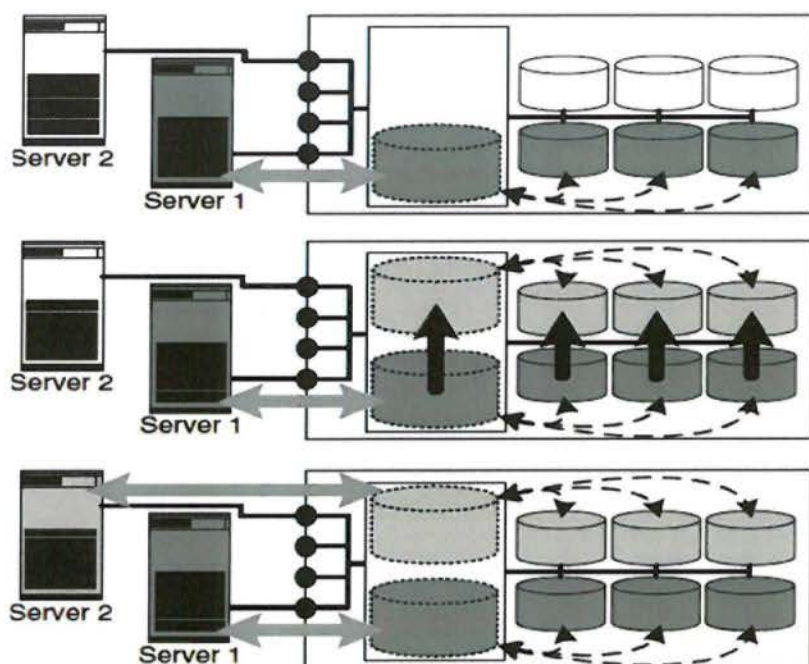
Cache του σκληρού δίσκου: κάθε σκληρός δίσκος που εισάγεται σε ένα τέτοιο σύστημα έχει μια μικρή cache μνήμη, αφού η ταχύτητα εγγραφής και ανάγνωσης είναι κατά πολύ μικρότερη από εκείνη στις εισόδους και εξόδους του controller.

Cache εγγραφής σε controller: Στα Intelligent disk subsystem έχουμε cache ξεχωρη από αυτή των σκληρών δίσκων η οποία είναι και αρκετά μεγαλύτερη, ως εκ τούτου μπορούμε να κάνουμε buffer πολύ περισσότερα δεδομένα για την καλύτερη διαχείρισή τους και την αποτελεσματικότερη απόκριση. Πολλές εφαρμογές στέλνουν δεδομένα σε blocks, αν λοιπόν ο server στείλει τα δεδομένα σε blocks τότε ο δίσκος τα κρατάει στην cache στέλνει πίσω πως τα αποθήκευσε και μετά τα περνά στο φυσικό επίπεδο ούτως ώστε να κρατήσει χώρο και για άλλα δεδομένα.

Cache ανάγνωσης σε controller: Στην ανάγνωση από το σκληρό δίσκο τα πράγματα για την βελτιστοποίηση της απόδοσης ακόμα και με την τεχνική της Cache είναι δύσκολα και αυτό γιατί δεν μπορούμε να ξέρουμε με ακρίβεια τι θα ζητήσει ο host από ένα σύνολο από δεδομένα οπότε εδώ μπαίνουν κάποιες τεχνικές πρόβλεψης ούτως ώστε τα δεδομένα που θα κρατήσουμε στον buffer να έχουν μεγάλη πιθανότητα να ζητηθούν από τους χρήστες της εφαρμογής και πρακτικά αυτό μπορεί να γίνει με την γνώση που πηγάζει από την κίνηση των δεδομένων κατά το παρελθόν.

Στιγμιαία αντίγραφα

Σε ένα υποσύστημα μπορούν να αντιγραφούν τα δεδομένα εικονικά μέσα σε μερικά δευτερόλεπτα. Αυτό πρακτικά σημαίνει πως ενώ το υποσύστημα δίνει την αίσθηση πως η αντιγραφή των δεδομένων χρειάζεται λίγα δευτερόλεπτα στην πραγματικότητα όμως η αντιγραφή παίρνει πολύ περισσότερο χρόνο τουλάχιστον για μεγέθη που αγγίζουν και τα terabytes.



[Πηγή 3]

Αυτό που απεικονίζει η παραπάνω εικόνα είναι μια τέτοια λειτουργία πρακτικά ο server 1 λειτουργεί με τα δεδομένα του lun1 όπου τα δεδομένα αντιγράφονται οπότε ο server 2 λειτουργεί με τα αντιγραμμένα δεδομένα. Αυτή η τεχνική χρησιμοποιείται περισσότερο για την δημιουργία test δεδομένων για backup δεδομένων καθώς και για data mining.

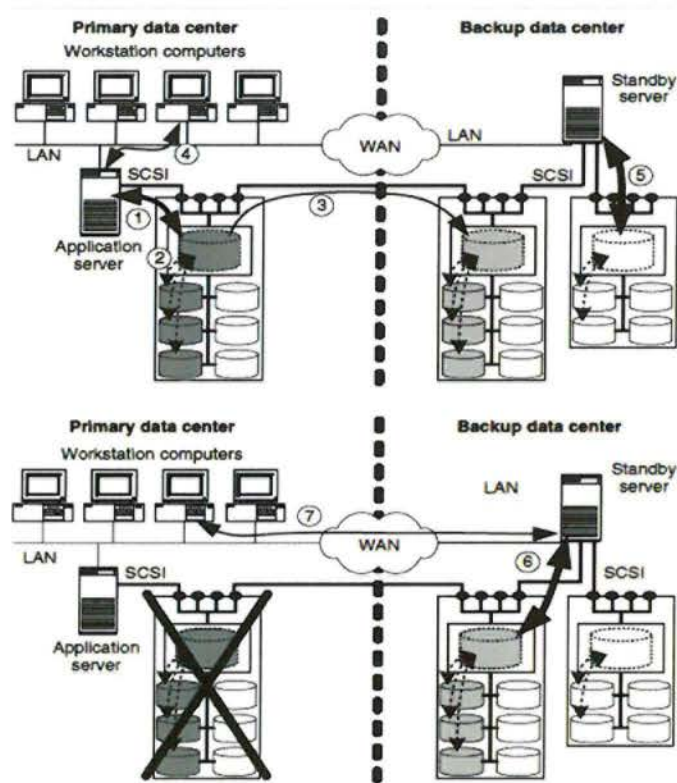
Υπάρχουν διαφορετικοί τρόποι για την υλοποίηση της συγκεκριμένης λειτουργίας, όλες όμως έχουν το κοινό πως δίνουν την αίσθηση της αντιγραφής των δεδομένων σε πολύ μικρό χρονικό διάστημα.

Όλες οι εφαρμογές χρειάζονται controller και cache για να μπορούν να γράφουν εκεί τα δεδομένα που αντιγράφονται.

Remote mirroring

Τα στιγμιαία αντίγραφα που προαναφέραμε προστατεύουν τα δεδομένα από αστοχίες του λογισμικού, δεν κάνουν το ίδιο όμως και σε αστοχίες του υλικού, αν πέσουν οι δίσκοι, αν για κάποιο λόγο χαλάσουν τα συστήματα, αυτό γεννά την ανάγκη να κρατάμε backup των δεδομένων μας σε απομακρυσμένους διακομιστές.

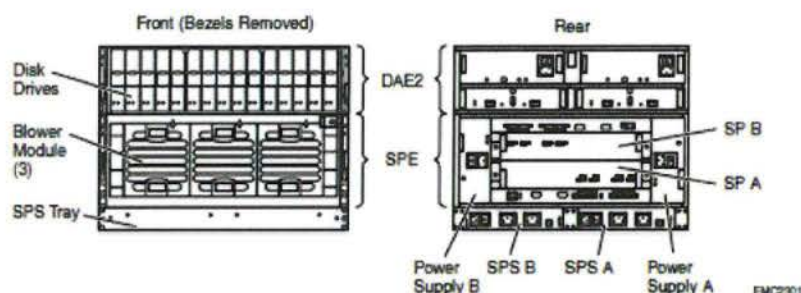
[Πηγή 3]



2.4 CX 700 Storage System

Περιλαμβάνει management συστήματα διασύνδεσης με fibre channel και ένα ή περισσότερα storage system. Πρακτικά συμμετέχει σε ένα storage network και είναι το μηχανήμα που παρέχει τα αποθηκευτικά μέσα. Όπως προαναφέραμε το fibre channel είναι ένα σειριακό πρωτόκολλο που επιτρέπει την μεταφορά δεδομένων από ένα τέτοιο υποσύστημα σε κάποιους server. Είναι χαμηλού επιπέδου και η τέταρτη βαθμίδα του υποστηρίζει άλλα πρωτόκολλα όπως scsi, ip.

Αναγνώριση βασικών στοιχείων
cx700 spe setup guide(emc)

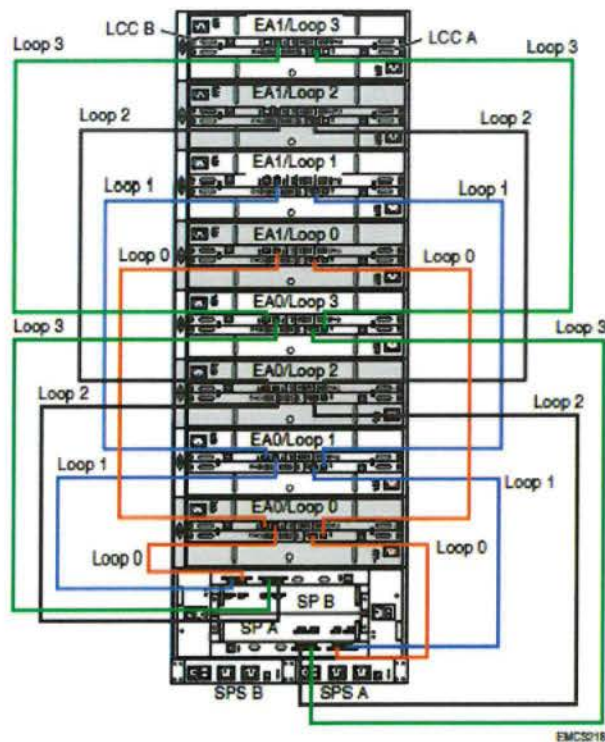


Πρώτα ας ξεκινήσουμε με τα power supply. Η καμπίνα που έχουμε στο εργαστήριο έχει 2 τροφοδοτικά τα οποία καταλήγουν σε πολλαπλές εξόδους ούτως ώστε να μπορούμε να δίνουμε ρεύμα σε δίσκους αλλά και στα υπόλοιπα εξαρτήματα. Στο spe βρίσκονται τα 2 storage processor.

Storage processor: Είναι ο επεξεργαστής που λειτουργεί σαν controller, το σύστημα μας έχει 2 και η διαχείριση τους γίνεται από δύο Ethernet πόρτες οι οποίες έχουν 10.1.101.127 και 10.1.101.126 αντίστοιχα και βρίσκονται στο 10.0.0.0 με μάσκα 255.0.0.0 αντίστοιχα η εφαρμογή που χρησιμοποιούν τα συγκεκριμένα μηχανήματα είναι το navisphere. **Προσοχή: τα sp τρέγουν λειτουργικό κανονικά και κάνουν boot από τους 5 πρώτους σκληρούς δίσκους του DAE0.** Τα sps είναι η τροφοδοσία στις μονάδες ψύξης είναι διαφορετικές (σειριακά καλώδια) αφού τα sps χρειάζονται dc τροφοδοσία. Με τους host του storage network συνδέονται μέσα από οπτικές ίνες υπάρχουν 2 πόρτες ανά sp και χρησιμοποιούν πολύτροπη οπτική ίνα διπλής κατεύθυνσης με LC connector, αυτό πρακτικά εγγυάται σε φυσικό επίπεδο την full duplex επικοινωνία.

Το σύστημα που χρησιμοποιεί για management είναι το Navisphere, τρέχει από τον internet explorer και χρειάζεται java plug-in για να μπορέσει ο explorer να παίξει την εφαρμογή. Από εκείνη την εφαρμογή μπορούμε να διαχειριστούμε πλήρως τα sp είτε η σύνδεσή μας είναι point-to-point είτε χρησιμοποιούμε κάποιο enterprise directory. Μερικές από τις εφαρμογές είναι δημιουργία lun , storage group .

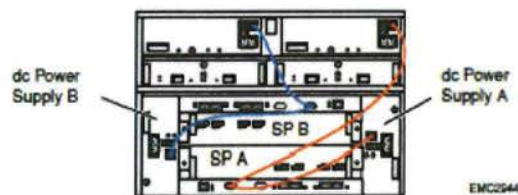
Storage System Back-End Connections: Εδώ αναφερόμαστε στη συνδεσμολογία των DAE. Κάθε sp έχει 4 back end πόρτες και εμείς διαθέτουμε 8 BE. Για να μπορέσουμε να συνδέσουμε όλα BE με τους sp θα πρέπει να δημιουργήσουμε κι άλλες loop έτσι ώστε τα 8 BE να μετατραπούν σε 4 DAE. Αυτό είναι εύκολα εφικτό αφού το cx700 σε κάθε DAE παρέχει 2 πόρτες. Κάθε DAE απαιτεί μοναδική enclosure address (EA), π.χ. Το πρώτο DAE έχει την μοναδική διεύθυνση EA 0, όσο ανεβαίνουμε έτσι αυξάνονται και οι διευθύνσεις, το δεύτερο DAE θα έχει EA 1 και πάει λέγοντας.



Cx700 spe setup guide(emc) Σύνδεση με το ρεύμα

Ξεκινώντας από το SPSs θα πρέπει να είναι συνδεδεμένα με power distribution units (PDUs) ακριβώς όπως δείχνει το παρακάτω παράδειγμα.

CX700 spe setup guide(emc)



Πέρα από τα SPSs θα πρέπει να είναι συνδεδεμένα στο ρεύμα και όλα τα BE. Επίσης υπάρχουν διακόπτες για το BE 0 όπου και βρίσκονται οι δίσκοι από τους οποίους ξεκινούν οι SP και για να μπορέσουμε να διαχειριστούμε το σύστημά μας θα πρέπει αυτούς να τους ανοίξουμε. Καλό θα ήταν να προσέξουμε να δίνουμε ρεύμα από διαφορετική πηγή στο εργαστήριο. Αυτό είναι εφικτό σε ένα βαθμό από την άποψη πώς τους δίνουμε από διαφορετική ασφάλεια από τον ίδιο πίνακα κι έτσι μπορούμε να ασφαλίσουμε το σύστημα μας γιατί σε περίπτωση που πέσει η μια ασφάλεια θα συνεχίσει να παίρνει ρεύμα από τη δεύτερη πηγή.

EMC ControlCenter Navisphere

To Navisphere ControlCenter έχει τα ακόλουθα εξαρτήματα.

- Navisphere Host and SP Agents
- Navisphere Manager
- Navisphere Integrator
- Navisphere Storage Management Server
- Navisphere CLI
- Navisphere Analyzer

Navisphere Host and SP Agents

To Host and SP Agents καταγράφει τα host bus adapters του server που συνδέονται στο storage system. Αυτές οι εγγραφές είναι που καθορίζουν και την πρόσβαση στα δεδομένα του storage system. Επιπλέον έχει και τις ακόλουθες λειτουργίες.

- Καταγράφει τα storage system event και ειδοποιεί τον admin με e-mail
- Ανακτά τα LUN WWN καθώς και πληροφορίες για το Symmetrix storage systems

Navisphere Manager

Είναι ένα σύστημα κεντρικής διαχείρισης και παραμετροποίησης του CLARiiON storage systems και διαθέτει τις ακόλουθες λειτουργίες:

- Αναγνώριση του CLARiiON storage system
- Πληροφορίες για την κατάσταση και παραμετροποίηση του συστήματος
- Διαχείριση διάφορων γεγονότων
- Κατανομή και παραμετροποίηση του αποθηκευτικού χώρου

Navisphere Integrator

Επιτρέπει στο storage systems που διαχειρίζεται το Navisphere να χρησιμοποιεί το enterprise management platforms (EMP) και του παρέχει τις παρακάτω εφαρμογές.

- HP OpenView® management software
- Tivoli NetView® management software
- CA Unicenter™ TNG Framework

Navisphere Storage Management Server

Είναι εγκατεστημένο και εκτελείται και στους δύο sp. Ένα storage system με αυτήν την εφαρμογή θεωρείται storage management server και παρέχει τις παρακάτω λειτουργίες:

- Δημιουργεί και απαντά σε ερωτήματα από το Navisphere Manager 6.X.
- Προωθεί ερωτήματα στον sp για επεξεργασία.
- Παρέχει πληροφορίες και ανανέωση στις παραμέτρους του Navisphere Manager 6.X.

- Πιστοποιεί τους χρήστες και τα δικαιώματα τους για τα αρχεία και τις ενέργειες που μπορούν να κάνουν
- Κρατά logs αρχεία με τα log in και τις ενέργειες των χρηστών

Navisphere Cli

Το cli συμπληρώνει το unisphere ως μία εναλλακτική μέθοδο για τη διαχείριση των storage system . Παρέχει ένα τερματικό μέσω του οποίου μπορούμε να διαχειριστούμε τους πόσους του συστήματός μας (σκληρούς δίσκους). Το κύριο πλεονέκτημα του συγκεκριμένου kit είναι πως μας παρέχει την δυνατότητα να εκτελέσουμε αυτόματες διαδικασίες, μέσω συναρτήσεων, στο Linux, αυτό μπορεί να γίνει μέσω κάποιων shell scripts ενώ στα windows μέσω κάποιων batch files.

Σύγκριση unisphere με navisphere

Το unisphere είναι μία εφαρμογή που εκτελείται σε φυλλομετρητή , είναι γραμμένη σε java και προσφέρει ένα εύχρηστο διαχειριστικό περιβάλλον. Στο σημείο αυτό πρέπει να τονίσουμε πως ο διακομιστής που φιλοξενεί την εφαρμογή αυτή τρέχει πάνω στους SPS (storage processors) .

Από την άλλη το Navisphere Cli είναι ένα περιβάλλον γραμμής εντολών με server base και web base εντολές . Οι server base είναι εντολές που επικοινωνούν απευθείας με το SPS ενώ οι web base είναι εντολές που επικοινωνούν με το διακομιστή που τρέχει πάνω στα SPS.

Unisphere	Navisphere Cli
Χρησιμοποιεί γραφικό περιβάλλον	Χρησιμοποιεί γραμμή εντολών
Απαιτεί την αλληλεπίδραση του χρήστη	Έχει την δυνατότητα αυτοματοποιημένης εκτέλεσης

Secure Cli

Το secure cli είναι μία παραλλαγή του cli που παρέχει ασφάλεια στην εκτέλεση των εντολών και αυτό πρακτικά σημαίνει δύο πράγματα: πρώτον ότι χρησιμοποιεί έλεγχο πως ο χρήστης που εκτελεί το συγκεκριμένο σενάριο είναι όντως πιστοποιημένος και δεύτερον χρησιμοποιεί κρυπτογράφηση για την ασφάλεια των στοιχείων του χρήστη. Εδώ πρέπει να τονίσουμε πως για να εκτελεστούν κάποιες εργασίες μέσω του cli χρειαζόμαστε κωδικό και όνομα χρήστη, είναι τα ίδια με αυτά που χρειαζόμαστε για να εισέλθουμε στο unisphere και από το διακομιστή που τρέχει στους SPS μπορούμε να διαχειριστούμε αυτά τα στοιχεία καθώς και να δημιουργήσουμε χρήστες.

Χρήση του Secure Cli

Ένα storage system δε θα δεχτεί κάποια εντολή εκτός αν ο χρήστης που την εκτελεί είναι πιστοποιημένος. Ο έλεγχος αυτός μπορεί να γίνει με δύο τρόπους α) με τη χρήση παραμέτρων στην εντολή και β) με τη χρήση ειδικού αρχείου που περιέχει τα στοιχεία σύνδεσης .

Παραδείγματα της εντολής :

naviseccli – help για να πάρετε σαν έξοδο τις πιθανές παραμέτρους της εντολής.

- ❧ address(-h) ορίζει τη διεύθυνση που έχει ο storage processor.
- ❧ Addusersecurity χρησιμοποιείται για τη δημιουργία χρήστη στο αρχείο που χρησιμοποιεί η εντολή .
- ❧ F αποθηκεύει την έξοδο της εντολής σε αρχείο.
- ❧ M επιστρέφει μόνο τους αριθμούς.
- ❧ nopoll(-np) πρακτικά απενεργοποιεί τις εντολές που εκτελούνται με την απλή cli.
- ❧ Parse (-p) ελέγχεται η σύνταξη της εντολής.
- ❧ Port καθορίζει την πόρτα του storage system
- ❧ q απενεργοποιεί τα μηνύματα λάθους.
- ❧ removeusersecurity χρησιμοποιείται για τη διαγραφή χρήστη στο αρχείο που χρησιμοποιεί η εντολή .
- ❧ Scope(0,1,2) ξεκαθαρίζει τον τύπο του λογαριασμού (0 global,2 local,3 ldap). Ldap : Ο χρήστης global όταν δημιουργείται έχει πρόσβαση στο SPS μέσω του domain, τα δικαιώματα που θα έχει καθορίζονται από τα schema του ldap.
- ❧ secfilepath καθορίζει το μονοπάτι στο οποίο βρίσκεται το αρχείο ασφάλειας το by default είναι το /home.
- ❧ User ορίζει το χρήστη που θέλει να αλληλεπιδράσει με το σύστημά μας.
- ❧ V Ενεργοποιεί τη λεπτομερή καταγραφή σφαλμάτων.
- ❧ Xml στέλνει την έξοδο της εντολής σε Xml αρχείο.

Luns,disk id

Σε unisphere και Navisphere Cli, τα Luns id είναι ένας μοναδικός αριθμός που καθορίζεται όταν δημιουργείται το Luns και μπορεί να γίνει με δύο τρόπους α) με αύξοντα αριθμό, το σύστημα δίνει από μόνο του έναν αριθμό. β) με επιλογή του χρήστη . Από την άλλη σε συσκευές όπως hba ,sps, switch έχουμε μοναδικές στον κόσμο διευθύνσεις, κάτι σαν τις mac (world wide name (wwn) world wide port name (wwpn)) .

Storage setup

Στο σημείο αυτό θα ασχοληθούμε με το πώς μπορούμε να διαμοιραστούμε τις αποθηκευτικές μας δυνατότητες με τη χρήση του secure cli.

Για αρχή αυτό που πρέπει να γίνει είναι να δημιουργήσουμε τα raid group στα οποία θα φιλοξενήσουμε τα Lun μας.

Στο σημείο αυτό ας δούμε μερικά παραδείγματα:

```
Naviseccli -h xx.xx.xx.xx createerg 0_0_0_0_1 0_0_2 0_0_3 0_0_4 0_0_5 0_0_6 0_0_7 0_0_8
```

Το παραπάνω παράδειγμα δημιουργεί ένα raid group για τους δίσκους 0_0_0 0_0_8 που βρίσκονται στο δίαυλο 0.

```
Naviseccli -h xx.xx.xx.xx createerg 0_0_0 0_0_2 0_0_4 0_0_5 0_0_7 0_0_8
```

Το παραπάνω παράδειγμα δημιουργεί ένα raid group για τους δίσκους 0_0_0 0_0_2 0_0_4 0_0_5 0_0_7 0_0_8 που βρίσκονται στον δίαυλο 0.

Στη συνέχεια θα πρέπει να δεσμεύσουμε κάποιο Lun σε αυτό το raid group.

Naviseccli -h xx.xx.xx.xx bind r5 0 -rg 0

Η παραπάνω εντολή δεσμεύει το raid group και φτιάχνει ένα Lun με raid 0.

Τέλος για να μπορέσουμε να έχουμε πρόσβαση στους σκληρούς δίσκου θα πρέπει να δημιουργήσουμε storage group στα οποία θα προσθέσουμε τα Lun καθώς και τα hba που βρίσκονται στα μηχανήματα που θέλουμε να έχουν πρόσβαση στους δίσκους.

Δημιουργία storage group : Naviseccli -h xx.xx.xx.xx storagegroup -create -gname storage1

προσθήκη Lun στο storage group : Naviseccli -h xx.xx.xx.xx storagegroup -addhlu -gname storage1 -hlu xx -alu xx

Ενώ για να προσθέσουμε έναν υπολογιστή στο storage group:Naviseccli -h xx.xx.xx.xx storagegroup -connecthost – host hba1 -gname storage1

Επιπλέον έχουμε τη δυνατότητα να διαχειριστούμε τη μνήμη cash του disk array

προβολή της κατάστασης στην οποία βρίσκεται η μνήμη :Naviseccli -h xx.xx.xx.xx getsp

naviseccli -h xx.xx.xx.xx setcashe -wcza 650 -wc 1 – p 8 -rcza 280 -rc 1

OpenStack storage

Στο openstack το storage χωρίζεται σε πολλά κομμάτια πράγμα που μπορεί να προκαλέσει σύγχυση ακόμα και στους πιο έμπειρους μηχανικούς .

	Ephemeral storage	Block Storage	Object storage
Χρησιμοποιούνται	Για την εκτέλεση του Λ.Σ.	Προσθήκη επιπλέον αποθηκευτικής δυνατότητας στην εικονική μηχανή	Αποθήκευση δεδομένου (ακόμη και εικονική μηχανή)
Πρόσβαση μέσω	Σύστημα αρχείων	Συσκευή όπως /dev/rdc	Client εφαρμογές
Διαχειρίσιμα μέσω	Openstack computer (nova)	Openstack block storage (cinder)	Openstack object stotage (swift)
Επιμένουν όσο	Το επιλέξει ο διαχειριστής	Το διαγράψει ο χρήστης	Το διαγράψει ο χρήστης
Πρόσβαση από	εικονικής μηχανής	εικονικής μηχανής	Οπουδήποτε
Καθορισμός μεγέθους	Διαχειριστής	Χρήστης	Συσκευή
Παραδείγματα	10 gbyte Πρώτος δίσκος 30 gbyte Δεύτερος δίσκος	1 TB δίσκος	10 TB δίσκος

Αξίζει να σημειώσουμε πως σε περίπτωση που εγκαταστήσουμε μόνο το Openstack compute (nova) τότε ο χρήστης δε θα έχει πρόσβαση στα αποθηκευτικά μέσα .

Object storage (swift):με το object storage οι χρήστες έχουν πρόσβαση σε binary αρχεία μέσω μιας εύχρηστης

εφαρμογής . Μία παρόμοια εφαρμογή με το swift είναι το Amazon s3 . Αν οι χρήστες του cloud θέλουν να διαχειρίζονται μεγάλους όγκους δεδομένων τότε πρέπει να τους δίνεται η δυνατότητα του Object storage Block storage (cinder):το block storage ορίζει μια block συσκευή του χρήστη. Ο χρήστης αλληλεπιδρά με το block storage με το να επισυνάπτει δεδομένα στις εικονικές μηχανές που είναι σε λειτουργία.

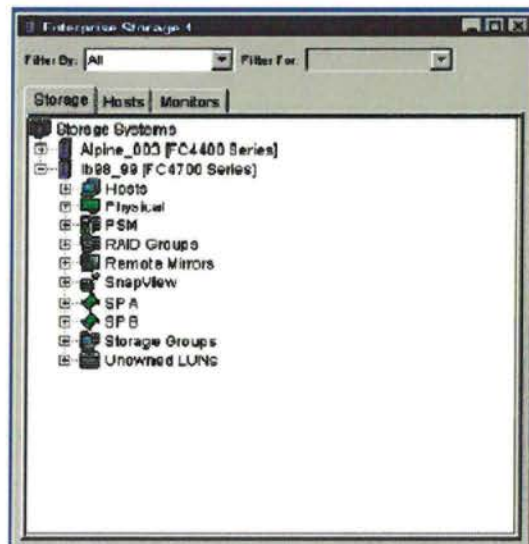
File level storage: Στο επίπεδο αυτό ο χρήστης έχει πρόσβαση στα δεδομένα μέσω του συστήματος αρχείων που παρέχει το λειτουργικό σύστημα.

Navisphere Analyzer

Είναι ένα εργαλείο που επιτρέπει στον διαχειριστή να βλέπει γραφικά κάποια κρίσιμα δεδομένα για την λειτουργία του συστήματος όπως είναι η μεταφορά δεδομένων. Ως αποτέλεσμα επιτρέπει να αναλύουμε το σύστημά μας και να προβλέπουμε αλλά και να αντιμετωπίζουμε διάφορα προβλήματα που θα μπορούσαν να σχετίζονται από λάθος σχεδιασμό ή κάποιες αστοχίες υλικού και λογισμικού.

Navisphere Environments

Για να μπορέσουμε να εγκαταστήσουμε τον client και να διαχειριστούμε σωστά το σύστημά μας θα πρέπει πρώτα να έχουμε εγκαταστήσει τον internet explorer καθώς και plug in της java. Επίσης θα πρέπει να έχει τρέξει το λειτουργικό του sp. Αυτό εξακριβώνεται από ένα πράσινο λαμπάκι που έχει η ethernet πόρτα που είναι επιφορτισμένη για την διαχείριση. Αφού βεβαιωθούμε πως ο sp τρέχει κανονικά τότε ξεκινάμε τον browser και πληκτρολογούμε την διεύθυνση του sp. Θα πρέπει να ξεκινήσει η εφαρμογή, έπειτα κάνουμε login με τα στοιχεία που μας έχουν δοθεί.



Tree of Navisphere Environments

Τα στοιχεία του navisphere βρίσκονται σε δένδροειδή δομή ακριβώς όπως και οι φάκελοι στα windows. Αυτό βοηθά τον administrator να εξοικειωθεί άμεσα με το περιβάλλον της εφαρμογής. Παρακάτω παρατίθεται

μια εικόνα με τα στοιχεία του navisphere.

Εδώ βλέπουμε την δενδρική δομή του συστήματος **hosts**, είναι οι server που μέσω κάποιου host bus adapter συνδέονται με το σύστημά μας είτε απευθείας είτε μέσω κάποιου enterprise directory. Το **storage group** αποτελείται από κάποιους hosts καθώς και από τα LUN με τα οποία μπορούν οι συγκεκριμένοι hosts να αλληλεπιδρούν, είναι λοιπόν μια υποτυπώδης διασύνδεση lun με hosts.

PSM LUN είναι σύστημα αποθήκευσης αποκλειστικά για τα Sps. Εκεί αποθηκεύονται κρίσιμες πληροφορίες.

Raid groups : για να δημιουργήσουμε κάποια lun πρέπει να τα εμπεριέχουμε μέσα σε κάποιο raid group (τεχνικές για ασφάλεια δεδομένων μας με την χρήση back up δίσκων ή τεχνικές ανάκτησης των δεδομένων σε περίπτωση αστοχίας του υλικού).

SP A είναι ο πρώτος storage possessor και αντίστοιχα **SP B** είναι ο δεύτερος storage possessor. Remote Mirrors είναι τυχόν απομακρυσμένα συστήματα που κρατούν αντίγραφα .

Unowned LUNs είναι κάποια LUN που δεν ανήκουν σε κανένα SP.

SnapView container περιέχει στιγμιότυπα από την cache, snapshots, and snapshot sessions.

Εισαγωγή στα LUNs

Είναι ένα σύνολο από σκληρούς δίσκους που δημιουργούν μια λογική μονάδα αποθήκευσης, την οποία ο διακομιστής βλέπει σαν ένα ενιαίο δίσκο. Στην δημιουργία του LUN δηλώνουμε τους δίσκους που περιλαμβάνει καθώς και τα bus που συνδέουν τους δίσκους με το υπόλοιπο σύστημα. Αυτό υποδηλώνει και την ακριβή θέση του δίσκου μέσα στο storage system αφού η αρίθμηση των δίσκων αφορά κάθε BE ξεχωριστά με αποτέλεσμα τα νούμερα αυτά να επαναλαμβάνονται. Η επιλογή του αριθμού των δίσκων εξαρτάται αποκλειστικά από το raid που θα χρησιμοποιήσουμε και αυτό αφού διαφορετικά raid επίπεδα έχουν και διαφορετικές απαιτήσεις. Πριν από την δημιουργία κάποιου lun λοιπόν θα πρέπει να δημιουργήσουμε κάποιο raid group. Μερικά παραδείγματα παρατίθενται παρακάτω.

RAID 5 : Είναι το αποδοτικότερο από όλα τα υπόλοιπα. Ναι μεν χρειάζεται περισσότερους δίσκους από τα υπόλοιπα raid αλλά με την ύπαρξη στοιχείων ισοτιμίας και τον διαμοιρασμό των δεδομένων σε όλους του δίσκους σε περίπτωση αστοχίας υλικού για κάποιους από τους σκληρούς δίσκους που διαθέτουμε τότε τα δεδομένα μπορούν να ανακτηθούν χωρίς κανένα πρόβλημα. Εδώ λοιπόν συνεχόμενα blocks γράφονται εναλλάξ στους δίσκους μαζί με στοιχεία ισοτιμίας για την ανάκτηση των δεδομένων. Επίσης παρέχει μεγάλη ταχύτητα κατά την ανάγνωση αφού ο φόρτος μοιράζεται σε πολλές κεφαλές. Έστω ότι γράφουμε μια σειρά από τα δεδομένα μας στους τέσσερις δίσκους και θέλουμε να δημιουργήσουμε τα στοιχεία ισοτιμίας ούτως ώστε να τα αποθηκεύουμε στην συνέχεια, αυτό πρακτικά θα είναι το αποτέλεσμα μιας Exclusive-OR της σειράς που μόλις γράψαμε. Κατά την εγγραφή των δεδομένων τότε το ψηφίο ισοτιμίας δημιουργείται εκ νέου. Τώρα σε περίπτωση καταστροφής κάποιου δίσκου έχουμε την ίδια διαδικασία για όλα τα στοιχεία τα οποία είχε.

RAID 4 : Εδώ τα δεδομένα κρατιούνται σε πολλούς δίσκους και τα parity bits βρίσκονται σε ένα ξεχωριστό δίσκο. Η διαφορά του με το προηγούμενο RAID είναι πως το 5 μοιράζει τα parity bits μέσα στους δίσκους .

RAID 3 : Είναι παρόμοιο με το 4 με την μόνη διαφορά πως έχει διαγράμμιση βασισμένη στο byte ενώ το 4 και το 5 δημιουργούν με δεδομένα ισοτιμίας με βάση τα block.

Ιδιότητες του LUN

Προσδιορίζουν τα ατομικά χαρακτηριστικά του κάθε lun. Έχουμε την δυνατότητα να δώσουμε τα βασικά χαρακτηριστικά του lun όταν το δημιουργούμε και μπορούμε να τα αλλάξουμε ενώ τα έχουμε δημιουργήσει. Τα στοιχεία που χαρακτηρίζουν ένα lun είναι τα ακόλουθα :

- LUN ID εκχωρείται κατά την δημιουργία και δεν αλλάζει.
- LUN SIZE εκχωρείται κατά την δημιουργία και εξαρτάται άμεσα από το raid group
- Element size
- Rebuild priority
- Verify priority
- Enable Read Cache
- Enable Write Cache
- Enable auto assign
- Number of LUNs to bind
- alignment offset
- Default Owner

Απαιτήσεις raid level για δίσκους

Raid type	RAID 5	RAID 3	RAID	RAID	RAID	DISK	HOT SPAIRE
Αριθμός δίσκων που απαιτούνται	3 – 16	5 or 9	4, 6, 8, 10, 12, 14, 16	2	03/16/13	1	1

Element size : Είναι ο αριθμός των τομέων όπου το storage μπορεί να γράψει ή να διαβάσει χωρίς να χρειαστεί πρόσβαση σε άλλο δίσκο. Το element size μπορεί να επηρεάσει το επίπεδο ενός raid. Όσο μικρότερο είναι το element size τόσο πιο αποτελεσματική είναι η εγγραφή και η ανάγνωση.

Rebuild priority: Είναι η προτεραιότητα με την οποία επανακτούμε τα δεδομένα σε περίπτωση αστοχίας δίσκου.

Verify priority: Είναι ο βαθμός προτεραιότητας ο οποίος ελέγχει την ισοτιμία για τυχόν αστοχίες, είναι σημαντική η σωστή διαχείριση του αφού μεγάλη προτεραιότητα σε μεγάλο όγκο πληροφορίας μπορεί να σημαίνει μεγάλο φόρτο στους sr. Από την άλλη μικρή προτεραιότητα σε κρίσιμα δεδομένα μπορεί να δημιουργήσει κίνδυνο για την ακεραιότητα των δεδομένων.

Τιμές	Χρονικό περιθώριο επαναδημιουργίας
ASAP(as quickly as possible)	0
HIGH	6
MENDIUM	12
LOW	18

Default Owner: Είναι ο sr που θα του ανήκει το lun.

Enable Read Cache : Ενεργοποιεί ή απενεργοποιεί την δυνατότητα του LUN να χρησιμοποιεί την cache μνήμη του controller για ανάγνωση. Είναι άλλο ένα κομμάτι του σχεδιασμού για το σύστημα μας αφού θα βοηθούσε να έχουμε τα δεδομένα που χρησιμοποιούνται περισσότερο.

Enable write Cache : Ενεργοποιεί ή απενεργοποιεί την δυνατότητα του LUN να χρησιμοποιεί την cache μνήμη του controller για γράψιμο , είναι άλλο ένα κομμάτι του σχεδιασμού για το σύστημα μας αφού θα βοηθούσε να έχουμε τα δεδομένα που χρησιμοποιούνται περισσότερο.

Alignment offset: Ορίζει το Logical Block Address (LBA). Βοηθά στη ευθυγράμμιση των blocks .

Enable Auto Assign: Αυτή η δυνατότητα μπορεί αν είναι είτε ενεργοποιημένη είτε όχι. Καθορίζει το αν θα μπορούν τα δεδομένα να χρησιμοποιούν τον δεύτερο sr σε περίπτωση που ο ένας από τους δύο πέσει.

Επέκταση raid group: Μπορούμε να επεκτείνουμε ένα raid group με επιπλέον δίσκους αυτό όμως δεν σημαίνει πως επεκτείνεται και η αποθηκευτική δυνατότητα του χρήστη σε ήδη δεσμευμένα lun . Αντί γι αυτό, μοιράζει την δυνατότητα των νέων δίσκων σε όλο το raid group ανοίγοντας χώρο και για άλλα lun.

RAID Group Defragmentation : Μετά από την συνεχή χρήση και την αλλαγή ίσως σε μεγέθη και lun είναι λογικό να δημιουργηθούν διάφορα κενά στον δίσκο πράγμα που θα μειώνει αισθητά τις δυνατότητες του συστήματος μας σε αποθηκευτικό χώρο. Με αυτήν την δυνατότητα μπορούμε αυτόν τον κενό χώρο να τον καλύψουμε και να έχουμε την βέλτιστη εκμετάλλευση του storage array.

Ιδιότητες RAID Group

Raid group id : είναι μοναδική διεύθυνση του Raid group

Expansion/defragmentation priority : Καθορίζει πόσο γρήγορα συμβαίνει τυχόν επέκταση ή ανασυγκρότηση.

Δημιουργία Raid group

Αφού εισάγουμε τα στοιχεία μας επιτυχώς και μπορούμε στο σύστημα τότε διαλέγουμε το storage system στο οποίο θέλουμε να δημιουργήσουμε το raid group και πατάμε δεξί κλικ και δημιουργία raid group. Εκχωρούμε raid group id οπότε, για να δημιουργήσουμε το raid group έχουμε δύο επιλογές: ή θα διαλέξει το σύστημα αυτόματα κάποιους δίσκους (ανάλογα με τον αριθμό που του έχουμε δηλώσει) ή θα του επιλέξουμε εμείς .

Δημιουργία LUN

Για να δημιουργήσουμε το lun πλοηγούμαστε στο storage system και πατάμε δεξί κλικ και δημιουργία lun, έπειτα επιλέγουμε το raid που θέλουμε να χρησιμοποιήσουμε (να υπάρχει το αντίστοιχο raid group) και εισάγουμε τα αντίστοιχα στοιχεία LUN ID, Element size, Rebuild priority, Verify priority, Enable read cache, and write caching, Number of LUNs, Default owner, Enable auto assign, Alignment offset, LUN size.

Πρόσβαση στα δεδομένα και storage group

Ένα κοινό storage system παρέχει την δυνατότητα να ελέγξουμε ποιος έχει πρόσβαση σε ποια δεδομένα. Αυτό σημαίνει πως ορίζει στον server σε ποια lun έχει πρόσβαση για εγγραφή και ανάγνωση. Το storage group είναι μία ομάδα με ένα ή περισσότερα lun και έναν ή περισσότερους server. Ένας server μπορεί να έχει πρόσβαση μόνο στα lun που είναι στο ίδιο storage group με αυτόν. Εάν δεν ενεργοποιήσουμε την δυνατότητα για έλεγχο στα δεδομένα δεν μπορούμε να δημιουργήσουμε storage group, οπότε όλοι οι server έχουν πρόσβαση σε όλα τα lun.

Για να μπορέσουμε να ενεργοποιήσουμε την συγκεκριμένη λειτουργία θα πρέπει να είναι εγκατεστημένο το Access Logix.

Στο Storage tab του Enterprise Storage dialog box, πλοηγούμαστε στο εικονίδιο του storage system για το οποίο θέλουμε να ενεργοποιήσουμε την λειτουργία πατάμε δεξί κλικ και ιδιότητες (properties) access control enable.

Ιδιότητες του Storage group

Storage system label : Είναι ένα όνομα που δίνεται στο storage system στο οποίο δημιουργούμε το νέο storage group.

Storage group name : Το όνομα του storage group by default είναι κάπως έτσι storage group n όπου το n είναι ένας αύξων αριθμός, παρόλα αυτά υπάρχει η δυνατότητα να το αλλάξουμε είτε κατά την διάρκεια που το δημιουργούμε είτε αργότερα.

Sharing State : υπάρχουν δύο πιθανές τιμές Dedicated και Sharable. Η δεύτερη επιτρέπει στον storage group να είναι συνδεδεμένος με πολλούς server και συνήθως χρησιμοποιείται για cluster . Το Dedicated επιτρέπει στο storage group μόνο έναν server

Δημιουργία storage group : Δεξί κλικ στο storage system στο οποίο θέλουμε να δημιουργήσουμε το storage group και έπειτα create storage group. Δίνουμε όνομα και sharing state έπειτα πατάμε ok .

Σύνδεση server με storage group

θα πρέπει να συνδέσουμε ένα host στο storage system ούτως ώστε να μπορεί να αλληλοεπιδρά με τα LUN . Ένας host μπορεί να συνδεθεί μόνο σε ένα storage system, αν επιχειρήσουμε να το βάλουμε σε δεύτερο τότε αυτόματα θα αποσυνδεθεί από το πρώτο. Όταν συνδέουμε έναν host στο storage group συνδέεται μέσα από το HBA που συνδέεται και με το storage system. Από εκεί και πέρα αν έχει παραπάνω HBA που συνδέονται με το SP υπάρχουν παραπάνω από ένα μονοπάτια. Έχουμε την δυνατότητα να συνδέσουμε παραπάνω από ένα host στο storage group αρκεί να έχουν το ίδιο λειτουργικό σύστημα το οποίο να υποστηρίζει πολλά LUN ή να διαθέτουν layered software(π.χ. Microsoft Cluster).

Για να συνδέσουμε κάποιο host στο storage group που θέλουμε πατάμε πάνω του δεξί κλικ και έπειτα διαλέγουμε τον host και τον μεταφέρουμε στους συνδεδεμένους hosts.

Περιγραφή των monitor event

Παρακολουθεί την κατάσταση του storage system για καθορισμένα monitor event και ειδοποιεί ανάλογα. Μπορούμε να του καθορίσουμε τον τρόπο με τον οποίο θα αντιδρά σε συγκεκριμένα monitor event , καθώς και επίσης μπορούμε να ορίσουμε σε ποιά monitor event θα αντιδρά. Μερικοί τρόποι με τους οποίους αντιδρά :

- Μέσω κάποιου email.
- Στέλνοντας snmp trap σε κάποιο network management tool.
- Χρησιμοποιώντας το emc phone home modem.
- Κάποια προκαθορισμένη αντίδραση.

Κατανεμημένη και κεντρική Παρακολούθηση

Μπορούμε να επιλέξουμε ανάμεσα σε κατανεμημένη και κεντρική παρακολούθηση: στην πρώτη, κάθε server αναλαμβάνει να παρακολουθεί τα δικά του event , ενώ στην δεύτερη υπάρχει κάποιος που κάνει την συγκεκριμένη δουλειά.

Λογισμικό για Event monitor

Event Monitor gui : Είναι κομμάτι του Navisphere Manager και τρέχει στο web browser .Το χρησιμοποιούμε για να παραμετροποιήσουμε το σύστημα παρακολούθησης, τα λάθη καθώς και για να βλέπουμε τα logs από το history events.

Event Monitor: Είναι μέρος του Navisphere Agent και μπορούμε να το παραμετροποιήσουμε ως ακολούθως.

Self-Monitor agent: Παρακολουθεί τα events του server που είναι εγκατεστημένο.

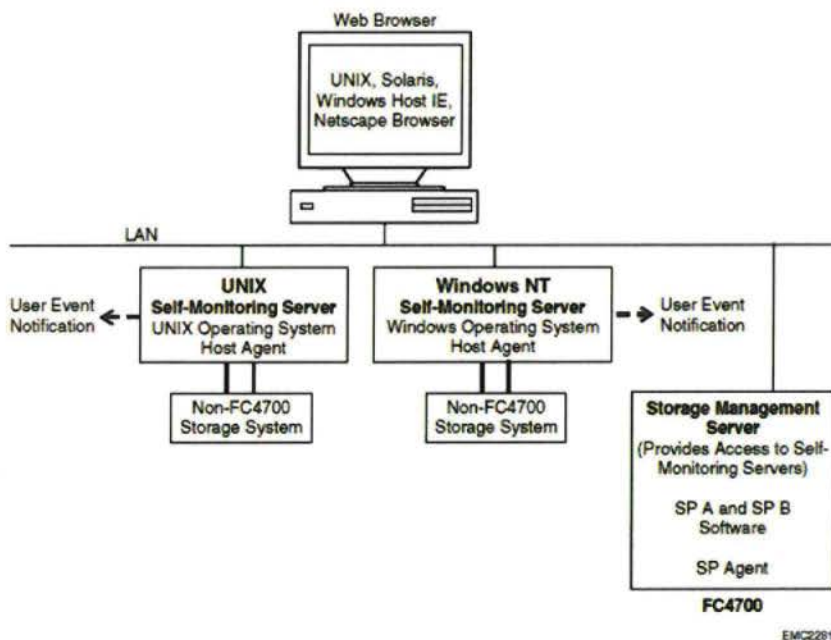
Centralized Monitoring Agent : Ο server παρακολουθεί τα events όλων και τους ειδοποιεί.

Monitored Agent : Είναι μέρος του Navisphere Agent και μπορούμε να το παραμετροποιήσουμε ως ακολούθως.

Self-Monitor agent : παρακολουθεί τα events του server που είναι εγκατεστημένο.

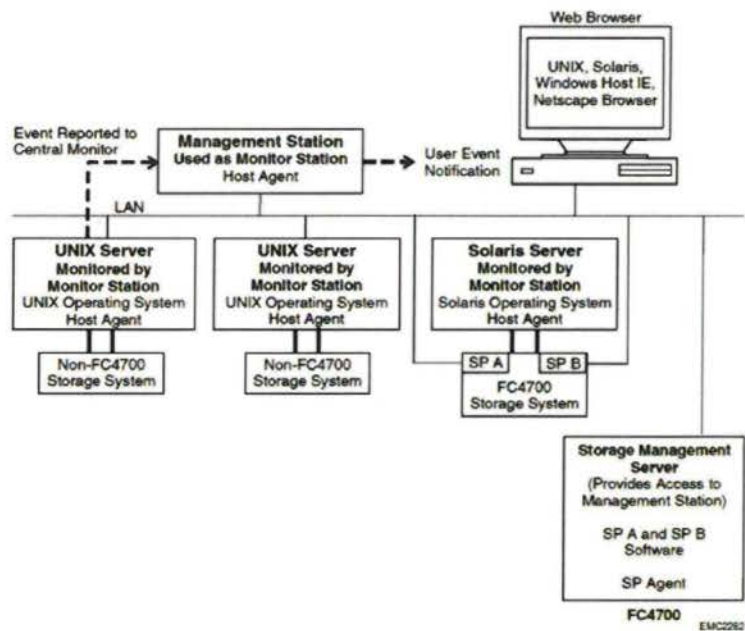
Centralized Monitoring Agent: Ο server παρακολουθεί τα events όλων και τους ειδοποιεί σε κάποιο.

Monitored Agent: Σε έναν server εγκαθίσταται η εφαρμογή που παρακολουθεί τα events, αυτή είναι η εφαρμογή του Centralized Monitoring.



Κατανεμημένο σύστημα Παρακολούθησης

2.3 Intelligent disk subsystem



Centralized Monitoring System

Monitoring Storage-System Operation

Εικονίδιο	Σημασία
	Όλα τα εξαρτήματα λειτουργούν κανονικά
	Ένα ή περισσότερα εξαρτήματα έχουν βλάβη.
	Κάποιο εξάρτημα βρίσκεται σε μεταβατική κατάσταση

	To storage system είναι κάτω
	To storage system δεν υποστηρίζεται

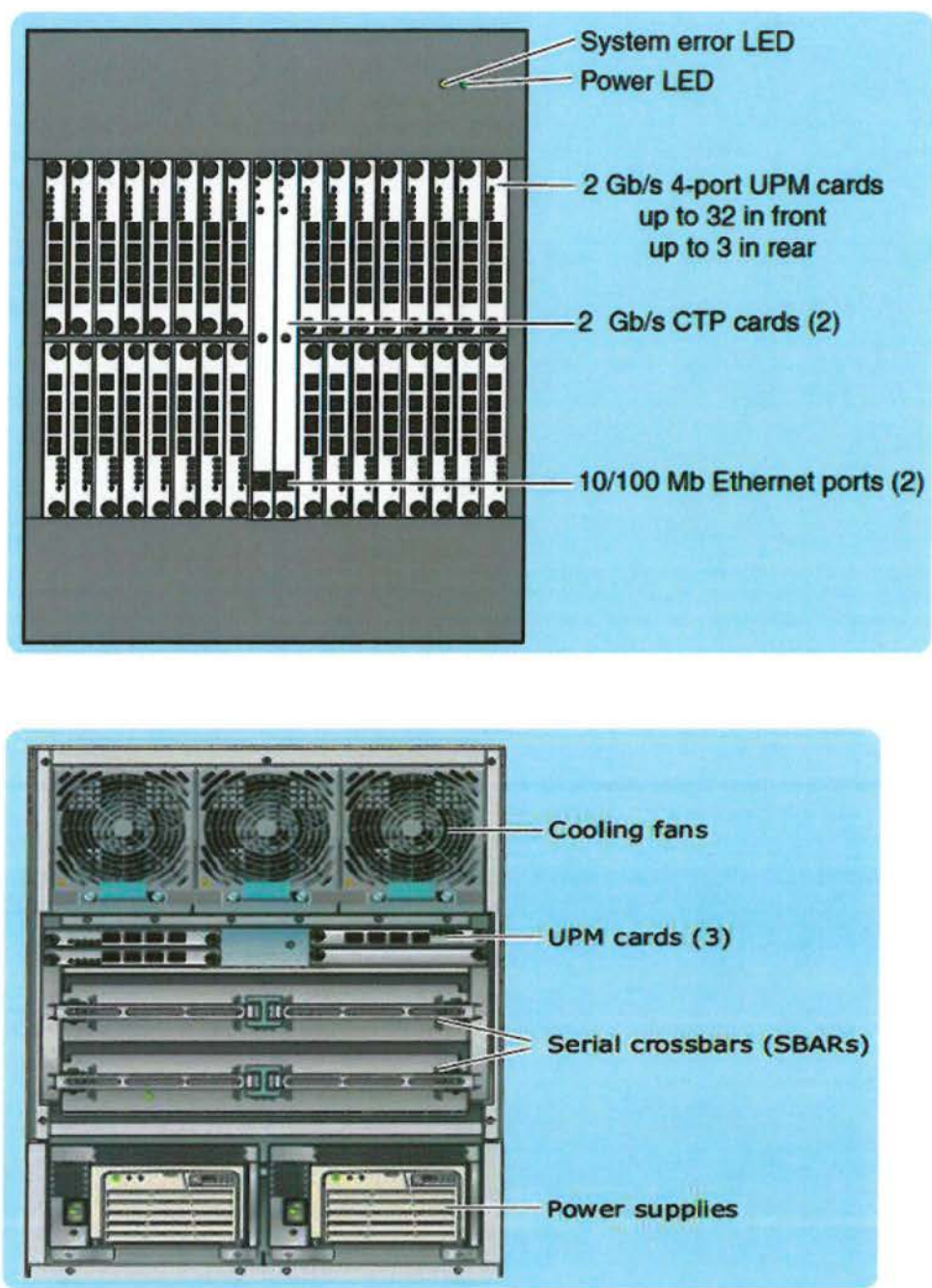
Προβλήματα για FAN A και FAN B

Σε κάθε σύστημα fibre channel storage system εικονίδιο με ένδειξη fan a σημαίνει ότι κάποια μονάδα ανεμιστήρα έχει ένα ή περισσότερα errors. Από την άλλη error με την ένδειξη fan b σημαίνει πως ο ανεμιστήρας κάποιου SP έχει ένα ή περισσότερα προβλήματα. Το fan a σημαίνει πως κάποιος από τους δύο ανεμιστήρες έχει πέσει και το σύστημα παραμένει ενεργό φορτώνοντας τον άλλον περισσότερο, η αντίδραση σε αυτήν την περίπτωση θα πρέπει να είναι άμεση αντικατάσταση του ανεμιστήρα ούτως ώστε να μην δημιουργηθεί πρόβλημα και στον άλλο και σβήσει το σύστημά μας.

Συμπεράσματα για cx700

Το συγκεκριμένο storage system αποτελεί μία πολύ αξιόπιστη λύση για εργασίες παρόμοιες με αυτές που θέλουμε στο ΤΕΙ είναι όμως παλιάς τεχνολογίας και δεν είναι σε θέση πλέον να σηκώνει συστήματα μεγάλων εταιρειών όπως και έκανε παλιότερα αφού ο φόρτος είναι πολύ μεγαλύτερος και οι απαιτήσεις σε αποθηκευτικό χώρο όλο και αυξάνονται με την πάροδο του χρόνου και την εξέλιξη της τεχνολογίας. Τώρα για τις δικές μας απαιτήσεις είναι ένα σύστημα που μπορεί να αντεπεξέλθει στις ανάγκες του εργαστηρίου και για την δουλειά που εμείς το θέλουμε αρκεί όλο το σύστημα να σχεδιαστεί σωστά και να μην υπάρχει πλεονασμός στην χρήση των πόρων. Για να γίνει κάτι τέτοιο όμως πρέπει να μελετήσουμε καλύτερα και τα υπόλοιπα κομμάτια του δικτύου αποθήκευσης καθώς και του δικτύου που θα βγάζει προς τον έξω κόσμο τα δεδομένα μας και τα μηχανήματα.

2.4 Enterprise Directory 140m



Στις παραπάνω εικόνες διακρίνονται το μπροστά και το πίσω μέρος του ed 140m καθώς και τα βασικά στοιχεία που έχει το συγκεκριμένο enterprise directory .

Χρησιμοποιεί την εύρωστη αρχιτεκτονική non-blocking. Πρακτικά αυτό σημαίνει πως κάθε θύρα υποστηρίζει bandwidth 2gb/s διπλής κατεύθυνσης . Είναι ένα director class switch που παρέχει δυναμική

διασύνδεση ανάμεσα σε server και storage device σε ένα san . Είναι εγκατεστημένα στην ec-1500 καμπίνα που ενεργεί ως management server. Η βασική εφαρμογή για την διαχείριση ενός τέτοιου συστήματος είναι το connectrix management, μπορούμε να το διαχειριστούμε τοπικά ή και απομακρυσμένα μέσα από κάποιο απλό δίκτυο.

Όλα τα εξαρτήματα υπάρχουν δύο φορές και αυτό γίνεται γιατί σε περίπτωση βλάβης θα πρέπει το σύστημά μας να συνεχίσει να παίζει, αφού οι εφαρμογές που παρέχουν τέτοια μηχανήματα είναι κρίσιμες και δεν πρέπει να πέφτουν με το παραμικρό.

Τροφοδοσία: Έχει δύο τροφοδοτικά και μοιράζονται τον φόρτο τροφοδοσίας σε περίπτωση βλάβης. Το τροφοδοτικό που απομένει αναλαμβάνει τον όλο φόρτο του συστήματος, είναι αναγκαία η άμεση αντικατάσταση του παρόλο που είναι ορατός ο κίνδυνος της υπερθέρμανσης και της απώλειας και του δεύτερου τροφοδοτικού. (Σε αυτό το σημείο θα ήθελα να τονίσω την σημασία που έχει ένα σύστημα να είναι πάνω 24/μέρα αφού σε ορισμένες εφαρμογές απώλεια έστω και μιας ώρας μέσα σε ένα μήνα μπορεί να σημαίνει μεγάλη οικονομική ζημιά ανάλογα και με τις εφαρμογές που αυτός παρέχει και τοις οικονομικές συμφωνίες που έχει κάνει με τους πελάτες του.).

Μονάδες ανεμιστήρων : Κάθε μια τέτοια μονάδα έχει τρεις ανεμιστήρες. Σε περίπτωση απώλειας κάποιας μονάδας τότε οι υπόλοιπες ανεβάζουν στροφές για να επαναφέρουν το σύστημά μας σε σωστή θερμοκρασία. Πρακτικά αυτό είναι επικίνδυνο αφού υπάρχει μεγάλος κίνδυνος να καούν και αυτή και το σύστημα μας να κλείσει. Άρα σε περίπτωση που χάσουμε κάποια μονάδα πρέπει να την αντικαταστήσουμε άμεσα.

Connectrix manager

Είναι διεπαφή για τον connectrix server , προωθεί εάν σύστημα κεντρικής διαχείρισης για multiple directors ή απλά switches σαν διαχειριστής με την συγκεκριμένη εφαρμογή μπορείς να ελέγξεις ψευδώνυμα και κωδικούς χρηστών καθώς και την πρόσβαση στις πόρτες του συστήματος. Μία από τις βασικότερες λειτουργίες του συγκεκριμένου συστήματος είναι το zoning πράγμα που θα εξηγήσουμε αναλυτικότερα παρακάτω.

Fabric zoning

Το zoning είναι εφικτό μόνο στο fibre channel switched fabric περιβάλλον και αυτό γιατί κάθε συσκευή που συνδέεται στο fibre channel switch λογίζεται σαν κόμβος και μπορεί να ταυτοποιηθεί από το source id, που εκχωρείται όταν εισάγεται στο fibre channel fabric. Το s_id είναι 32-bit διεύθυνση που παράγεται από το domain id και την πόρτα στην οποία είναι συνδεδεμένος ο κόμβος, εκτός από το s_id ο κόμβος έχει και μία μοναδική διεύθυνση 64-bit (wwn). Τα μέλη μιας ζώνης μπορούν να είναι HBA, Storage ports η οποιαδήποτε fibre channel πόρτα.

Η ζώνες χωρίζονται σε δύο κατηγορίες.

Port zoning: Που γίνονται με βάση το domain id και την πόρτα στην οποία είναι συνδεδεμένοι οι κόμβοι

πρακτικά αυτό είναι το s_id.

Wwn zoning: Εδώ οι ζώνες δημιουργούνται με βάση την μοναδική ταυτότητα των κόμβων.

Port zoning: Ζώνες με βάση το domain id του switch και την πόρτα που είναι συνδεδεμένος ο κόμβος όταν χρησιμοποιούμε την συγκεκριμένη τεχνική μόνο οι πόρτες που βρίσκονται στην ίδια ζώνη μπορούν να αλλάξουν δεδομένα μεταξύ τους επίσης οι κόμβοι που βρίσκονται εντός ζώνης δεν μπορούν να στείλουν πακέτα έξω από αυτήν. (Εδώ είναι αναγκαίο να ξεκαθαριστεί πως αναφερόμαστε σε storage network ένας υπολογιστής που συμμετάσχει σε μία τέτοια ζώνη μπορεί να συνδιαλλαγεί μόνο με τα μηχανήματα που έχει στην ζώνη του για αυτό το δίκτυο και να πάρει ή να δώσει δεδομένα σε ή από αυτούς, δεν υπάρχει καμία δέσμευση στο πώς θα χειριστεί αυτά τα δεδομένα είναι ξεκάθαρα θέμα του διαχειριστή.) Κύριο μειονέκτημα της συγκεκριμένης εφαρμογής είναι πως αν θέλουμε να αλλάξουμε ζώνη, κάποιον κόμβο, θα πρέπει να αλλάξουμε την πόρτα στην οποία είναι, αυτό απαιτεί φυσική παρουσία στον χώρο και δεν είναι πάντα το πιο εύκολο πράγμα στον κόσμο.

Επιπλέον χαρακτηριστικά της συγκεκριμένης τεχνικής:

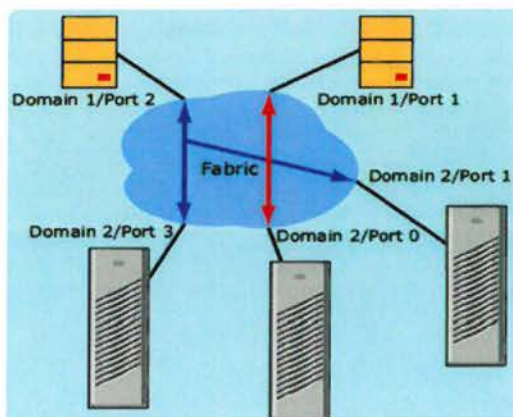
Ασφάλεια: Είναι περισσότερο ασφαλές από το wwn zoning αφού οι αλλαγές στην ζώνη εκτελούνται από το switch. Για να αλλάξουμε κάποια ζώνη χρειάζεται μεταφορά της οπτικής ίνας σε άλλη πόρτα.

Αντικατάσταση HBA: Το hba δεν συμμετέχει ενεργά στην δημιουργία ζώνης πρακτικά αυτό σημαίνει πως δεν χρειάζεται κάποια αλλαγή στις ρυθμίσεις απλά το βάζουμε στην ίδια πόρτα.

- Παρόλα αυτά η EMC δεν προτείνει την συγκεκριμένη λύση αφού παρουσιάζονται τα παρακάτω αρνητικά.

- Οι πόρτες του switch αλλάζουν και χρειάζεται αλλαγή με το χέρι.
- Αν αλλάξει το domain id το zoning σταματά να λειτουργεί.
- Αντικαθιστώντας ένα hba πρέπει να αλλάξεις τις ρυθμίσεις στο storage system, οπότε εδώ δεν προσφέρεται και μεγάλη εξυπηρέτηση.

Port zoning



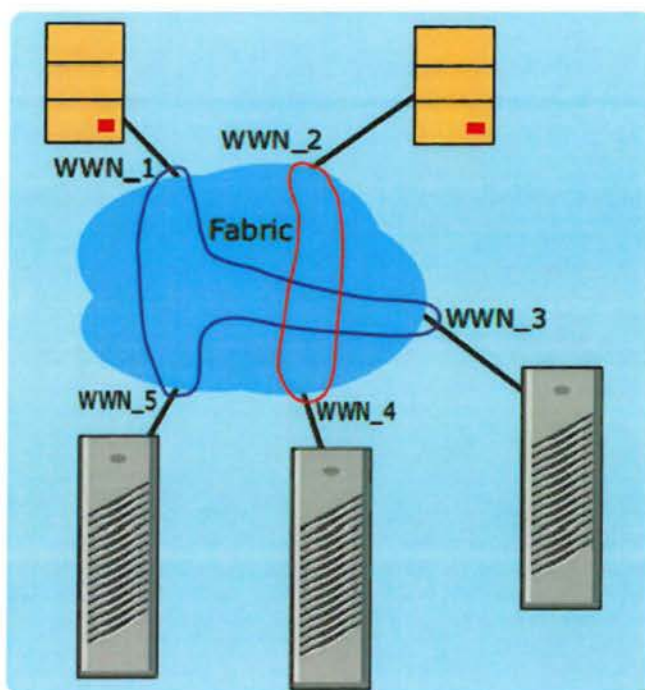
WWN zoning: Δημιουργούνται ζώνες με βάση τα wwn (μία 64-bit μοναδική διεύθυνση που εκχωρείται στους κόμβους ενός fibre channel δικτύου από τον κατασκευαστή κάτι παραπλήσιο με την mac μίας κάρτας δικτύου). Τα wwn καταγράφονται στο switch name server με όλα τα wwn που έχουν εισαχθεί. Το wwn zoning επιτυγχάνεται μέσω μίας NS λειτουργίας. Όταν π.χ. ένα hba εισάγεται το NS δημιουργεί μια λίστα με wwn τα οποία είναι στην ίδια ζώνη από εκεί και πέρα ο κόμβος μας μπορεί να επικοινωνεί μόνο με αυτά.

Δύο είναι τα βασικά πλεονεκτήματα αυτής της τεχνικής.

Ευελιξία: Αν αλλάξουμε τα καλώδια οπτικών ινών δεν χρειάζεται αλλαγή αφού οι ζώνες χτίζονται βάσει των wwn και η πόρτα στην οποία είναι συνδεδεμένα δεν παίζει κανένα ρόλο.

Αναδιαμόρφωση fabric: Αν αλλάξουμε κάτι στο δίκτυό μας χωρίς να πειράζουμε τα domain id τότε δεν χρειάζεται καμία απολύτως αλλαγή αφού όλη η ουσία στην συγκεκριμένη τεχνική κρύβεται στο NS. Επιπλέον υπάρχουν και μερικά αρνητικά στοιχεία σε αυτήν την τεχνική. Είναι δύσκολο να αλλάξει το hba και να αντιστοιχίσεις το σωστό wwn στο νέο hba αφού υπάρχει ο κίνδυνος να δώσεις πρόσβαση σε μη εξουσιοδοτημένο χρήστη. Αν καταστραφεί κάποιο hba πρέπει εκ νέου να σχεδιαστεί η ζώνη αφού προκύπτουν νέα στοιχεία(wwn).

WWN zoning



Port binding: Χρησιμοποιείται στα connectrix m συστήματα. Πρακτικά επιτρέπει στον διαχειριστή να περιορίσει ή να επιτρέψει την πρόσβαση στις πόρτες. Ενεργοποιεί δηλαδή την πόρτα και όταν κάποιο καλώδιο

οπτικών ινών συνδεθεί εκεί πάνω έχει αυτόματα πρόσβαση στην ζώνη που είναι ορισμένη η πόρτα . Η isls είναι μία ειδική περίπτωση της συγκεκριμένης τεχνικής. Υπάρχουν τα wwn που είναι κοινά για όλες τις πόρτες ενός fabric και τα wwn που είναι μοναδικά για κάθε πόρτα. Πρακτικά αυτό που αναλαμβάνει να κάνει ένας διαχειριστής είναι μέσα στο wwn να ενεργοποιήσει κάποια wwn όλα τα υπόλοιπα δεν έχουν το δικαίωμα να αφήσουν πακέτα. **Simple hba zoning:** Εδώ μπορεί να υπάρχει μόνο ένα hba μέσα σε μία ζώνη ενώ από την άλλη Storage-Systems μπορούν να υπάρχουν παραπάνω του ενός. Επιπλέον ένα hba μπορεί να συμμετέχει σε περισσότερες από μία ζώνες.

Κεφάλαιο 3 - SYNNEFO

Το synnefo είναι ένα cloud λογισμικό, χρησιμοποιείται για να δημιουργήσει μαζικά επεκτάσιμα cloud εικονικών μηχανών και χρησιμοποιεί google ganetti για την διαχείριση των εικονικών μηχανών. Είναι ένα σύστημα με 5 project γραμμένο σε python.

- Astakos
- Pithos
- Cyclades
- Plakton
- Archipelagos

3.1 Astakos

Παρέχει identity management και δημιουργήθηκε για να χρησιμοποιηθεί από το synnefo αλλά είναι opensource .

Είναι το μοναδικό σημείο authentication για το gnet cloud service . Είναι μια πλατφόρμα που επιτρέπει στους χρήστες να διαχειριστούν τους λογαριασμούς τους .

Επεκτείνει την snf-manage command line tool συμπεριλαμβάνοντας εργαλεία για την διαχείριση των λογαριασμών των χρηστών. Είναι φτιαγμένο πάνω στο django και επεκτείνει τους μηχανισμούς ταυτοποίησης .

Ο Astakos συντονίζει την πρόσβαση στους πόρους και πράττει σαν το μοναδικό σημείο εγγραφής και εισόδου στο gnet.

Registration user case:

Invited use : Ένας διαχειριστής στέλνει πρόσκληση σε κάποιον, ο οποίος λαμβάνει ένα e-mail με υπερσύνδεσμο το οποίο οδηγεί στην σελίδα εγγραφής του Astakos ,τότε ο χρήστης είναι έτοιμος να γραφτεί.

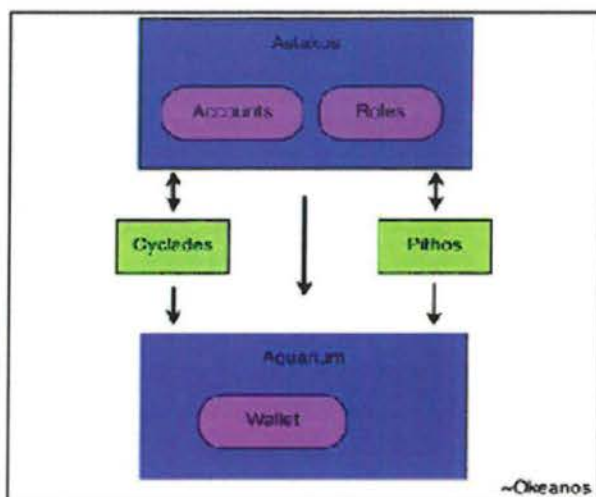
Non invited use : Ένας χρήστης πλοηγείται μέσω τις σελίδας, στην σελίδα εγγραφής και συμπληρώνει τα στοιχεία του, τότε κάποιος διαχειριστής τον εγκρίνει και του στέλνεται ένα e-mail με τα στοιχεία σύνδεσης οπότε είναι έτοιμος να εισέλθει στο σύστημα.

3.2 Authentication use cases

Cloud service user: Κατά την είσοδο μας στο σύστημα δημιουργείται ένα x-auth-token που είναι το διαβατήριό μας για όλες τις εφαρμογές του synnefo. Όταν για παράδειγμα θέλουμε να χρησιμοποιήσουμε κάποια εφαρμογή όπως για παράδειγμα το Pithos ελέγχετε αν το token που προαναφέραμε ανήκει ή όχι σε κάποιον χρήστη και αν ο χρήστης αυτός είναι ενεργός και έχει την δυνατότητα να χρησιμοποιήσει την συγκεκριμένη εφαρμογή.

Registration flow : Η ευθύνη για το χειρισμό των λογαριασμών βρίσκεται στο signupdate, αυτό θα ελέγξει

πότε κάποιο ερώτημα προέρχεται από χρήστη και αν δεν είναι εγγεγραμμένος θα προωθήσει τον χρήστη στην σελίδα από την οποία θα εισαγάγει τα στοιχεία σύνδεσης.



Astakos user and authentication

Ο astakos ενσωματώνει το user authentication του django και το επεκτείνει αφού στο django τα ονόματα χρηστών είναι 30 χαρακτήρες ενώ στον astako η ταυτοποίηση γίνεται από την ηλεκτρονική διεύθυνση του κάθε χρήστη. Οι χρήστες που έχουν εισαχθεί στο σύστημα έχουν την δυνατότητα να κάνουν τις εξής ενέργειες.

- Να αλλάξουν το προφίλ τους : /im/profile
- Να αλλάξουν τον κωδικό τους : /im/password
- Να στείλουν κάποια ανάδραση προς τους μηχανικούς: /im/feedback
- Να αποσυνδεθούν : /im/logout

Ένας χρήστης για να συνδεθεί χρειάζεται ταυτοποίηση μέσα από ένα Authentication token, αν ο χρήστης που προσπαθεί να χρησιμοποιήσει την εφαρμογή δεν έχει προωθείται αυτόματα στο /login.

Επιπλέον πληροφορίες που θα επιστραφούν έπειτα από την σύνδεση με json.

username	Μοναδική ταυτότητα χρήστη
uniq	Ηλεκτρονική διεύθυνση του χρήστη
auth_token	Authentication token
auth_token_expires	Η μέρα που λήγει ο λογαριασμός
auth_token_created	Η μέρα δημιουργίας του λογαριασμού
has_credits	Αν ο χρήστης έχει credits
has_signed_terms	Αν έχει συμφωνήσει στους κανονισμούς

Παράδειγμα στοιχείων

```
{ "userid": "23a0349394304as039",  
  "email": ["sysvlassis@gmail.com"],  
  "name": "apanasego apanasego",  
  "auth_token_created": 1333372365000,  
  "auth_token_expires": 1335964365000,  
  "auth_token": "uiWDLAgtJOGW4mI4q9R/8w==",  
  "has_credits": false }
```

Για να λάβουμε στην ίδια μορφή τοις παρεχόμενες υπηρεσίες πρέπει να επισκεφτούμε τον παρακάτω υπερσύνδεσμο .

/im/get_menu με την μέθοδο get και τα αποτελέσματα που θα πάρουμε θα είναι της μορφής:

```
[{"url": "/", "icon": "home-icon.png", "name": "grnet cloud", "id": "cloud"}]
```

Ενώ το menu από /im/get_menu και θα είναι της μορφής

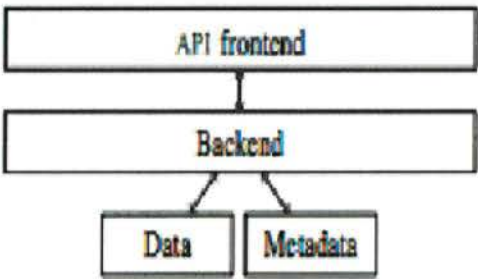
```
[{"url": "/im/profile", "name": "sysvlassis@gmail.com"},  
 {"url": "/im/profile", "name": "view your profile..."}, {"url": "/im/password", "name": "change your  
password..."}, {"url": "/im/feedback", "name": "feedback..."}, {"url": "/im/logout", "name": "logout..."}]
```

3.3 Pithos object storage service

Το Pithos είναι ένα αντικειμενοστραφές storage service βασισμένο στο OpenStack object storage api με διάφορες επεκτάσεις. Χρησιμοποιεί ένα block-base μηχανισμό για να επιτρέπει στους χρήστες να ανεβάσουν και να ανακτήσουν δεδομένα, να κρατάει διαφορετικές εκδόσεις αρχείων. Ακολουθεί ένα leyer modular

implementation.

Pithos desing: Είναι σχεδιασμένο σε επίπεδα. Με τον έξω κόσμο επικοινωνεί μέσω http το οποίο εκτελεί μερικές διεργασίες στο OpenStack object storage api . Το κομμάτι πίσω είναι μια βιβλιοθήκη που επιτρέπει τον καθορισμό διαφορετικού ονόματος για τα μεταδεδομένα, ούτως ώστε το ίδιο αντικείμενο να μπορεί να είναι ορατό από διαφορετικούς χρήστες και να έχουμε εξοικονόμηση χώρου. Τα δεδομένα αποθηκεύονται σε σύστημα αρχείων και τα μεταδιδόμενα σε βάση δεδομένων.



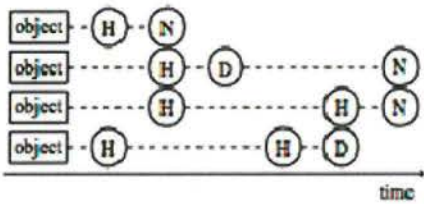
Block base storage: Κάθε αντικείμενο αποθηκεύεται σαν ομάδα από blocks και έτσι μπορούμε να ανταλλάσσουμε κομμάτια του block. Για κάθε έκδοση κρατάει το root Merkle hash, το μέγεθος του αντικειμένου, τον χρόνο της τελευταίας τροποποίησης και το χρήστη που το άλλαξε.

Pithos back end version: Για κάθε έκδοση αντικειμένου κρατάμε το root merkle hash του αντικειμένου, τον χρόνο της τελευταίας τροποποίησης και τον χρήστη που το άλλαξε έχουμε τις τρεις ακόλουθες καταστάσεις.

- Normal, που είναι η τρέχουσα έκδοση.
- History, που είναι η προηγούμενη έκδοση.
- Deleted, που είναι αντικείμενα που έχουν διαγραφεί.

Pithos back end permission: Το Pithos αναγνωρίζει και γράφει δικαιώματα που μπορούν να χορηγηθούν σε χρήστες ή ομάδες χρηστών, οι ομάδες γίνονται σε επίπεδο χρηστών και αυτό γίνεται επειδή δεν μπορεί μία ομάδα να έχει πρόσβαση σε δεδομένα άλλης.

Ένα κρίσιμο σημείο του cloud το openstack compute είναι opensource λογισμικό σχεδιασμένο να παρέχει μεγάλα δίκτυα εικονικών μηχανών δημιουργώντας μια επεκτάσιμη πλατφόρμα.



Pithos developer guide: Το Pithos είναι storage service ανεπτυγμένο από το grnet. Τα δεδομένα αποθηκεύονται σε αντικείμενα , οργανωμένα σε containers , ανήκουν στους χρήστες. Η ιεραρχία του pithos είναι εμπνευσμένη από το openstack object storage api και άλλα παρόμοια cloud files api . Το Pithos ακολουθεί το oos όσο το δυνατόν περισσότερο, μία από τις απαιτήσεις σχεδιασμού είναι να μπορεί ο χρήστης να συνδεθεί με oos client χωρίς αλλαγές. Παρόλα αυτά για να μπορέσουμε να εκμεταλλευτούμε πλήρως το pithos, το λογισμικό του client θα πρέπει να είναι πλήρως ενημερωμένο για τα επιπλέον κομμάτια του Pithos. Τα αντικείμενα του pithos μπορούν να γίνουν update.

Το backend του Pithos είναι block oriented, επιτρέπει την αποτελεσματική μοναδική τοποθέτηση δεδομένων. Η δομή των αντικειμένων εξαρτάται από το api layer, στην ουσία για να στηρίξει το επιπλέον λογισμικό και εφαρμοστούν προηγμένες λειτουργίες διαχείρισης δεδομένων.

Pithos users and authentication: Στο Pithos κάθε χρήστης χαρακτηρίζεται από μοναδικό token. Όλα τα api ζητάνε ένα τέτοιο token το οποίο είναι εσωτερική λύση για μία συμβολοσειρά λογαριασμού.

Το api χρησιμοποιείται για να καθορίσει ποιά αρχεία ανήκουν στους χρήστες και αν η αίτηση είναι τοπική η αν είναι ανάμεσα σε λογαριασμούς.

Το Pithos δεν κρατά βάσεις δεδομένων , για developers και testing ο χρήστης ελέγχεται από το token και τα αρχεία που του αντιστοιχούν μπορούν να ανακτηθούν από εκεί, έχει σχεδιαστεί με ένα ενδιάμεσο λογισμικό ταυτοποίησης και αυτό του επιτρέπει να χειριστεί τις λειτουργίες ταυτοποίησης των χρηστών.

Κάθε client εφαρμογή αν δεν έχει κάποιο token προωθείται στο /login, το οποίο δέχεται τις ακόλουθες παραμέτρους:

- Next: Είναι το επόμενο σημείο που θα οδηγηθεί μετά από την είσοδο στο σύστημα.
- Renew: Η δύναμη του token γιατί όταν τελειώσει η είσοδος στο σύστημα πρέπει να το προωθήσουμε στο σωστό uri και θα πρέπει να ξέρουμε αν έχει δικαίωμα να πάει εκεί.

Το σύστημα που παρέχει την ταυτοποίηση είναι το astakos από το grnet.

Τα uri request έχουν τοις παρακάτω μορφές

- Ανώτερο επίπεδο <http://127.0.0.1/v1>
- Επίπεδο χρήστη <http://127.0.0.1/v1/accout>
- Επίπεδο container <http://127.0.0.1/v1/accout/container>
- Επίπεδο object <http://127.0.0.1/v1/accout/container/object>

Όλα τα ερωτήματα πρέπει να έχουν x-auth-token ως παράμετρο.

Επίσης έχουμε τα παρακάτω σαν κωδικούς λάθους

- 400 Λάθος ερώτημα
- 401 Δεν έχεις πρόσβαση
- 403 Η υπηρεσία έχει πέσει
- 404 Δεν βρέθηκε η σελίδα

- 503 απρόσιτη υπηρεσία

Request parameter namevalue:

- Limit: Ζητείται κάποιος περιορισμός στον αποθηκευτικό χώρο του κάθε χρήστη.
- Marker: Επιστρέφει containers.
- Format: Προαιρετικού τύπου απάντηση (λίστα με χρήστες).
- Name: Το όνομα του λογαριασμού.
- Last_modified: Η τελευταία τροποποίηση.

Pithos+ Back End Nodes

Το pithos είναι οργανωμένο σε δεντρική δομή , με ένα κόμβο ανά μονοπάτι. Ένας κόμβος μπορεί να είναι κάποιος λογαριασμός, container ή και κάποιο αντικείμενο. Ένας χρήστης μπορεί να έχει πολλούς λογαριασμούς που με την σειρά τους μπορούν να έχουν πολλά container που με την σειρά τους μπορούν να έχουν πολλά αντικείμενα τα οποία μπορούν να έχουν πολλές εκδόσεις.

Είναι το υπολογιστικό/δικτυακό κομμάτι του synnefo. Συνδυάζει το google ganeti για την διαχείριση των εικονικών μηχανών με το python/django στο ορατό κομμάτι του χρήστη . Έχει επιλεγεί το ganeti για την διαχείριση των εικονικών μηχανών, αφού είναι μία αξιόπιστη λύση και δεν χρειάζεται να ανακαλύψουμε εκ νέου τον τροχό.

Στηρίζοντας την δουλειά στο ganeti κατάφεραν να στήσουν μία σταθερή εφαρμογή που θα αναλάβει την διαχείριση των εικονικών μηχανών σε χαμηλό επίπεδο, μεταφορά από φυσικό κόμβο σε άλλο όταν αυτό απαιτείται, αντιμετώπιση σε τυχόν αστοχίες κόμβων, έτσι λοιπόν ο σχεδιασμός και η υλοποίηση της διεπαφής του χρήστη γίνεται πιο εύκολη.

Με τις cyclades οι χρήστες έχουν τις εικονικές μηχανές προερχόμενες από kvm,τρέχουν linux και windows σε debian host και χρησιμοποιούν ganeti για την διαχείριση του cluster των εικονικών μηχανών, οι οποίες είναι προσβάσιμες από τον τερματικό χρήστη μέσω web ή μέσω προγραμματισμού (openstack compute api).Οι χρήστες έχουν πλήρη έλεγχο των εικονικών μηχανών τους μπορούν δηλαδή να δημιουργήσουν, να ορίσουν πόσους επεξεργαστές και μνήμη (ram αλλά και δίσκους) θα έχουν, να τις ξεκινήσουν, να τις σβήσουν , να τις επανεκκινήσουν αλλά και να τις καταστρέψουν.

3.4 Network Service (Cyclades)

Οι cyclades αντιλαμβάνονται δύο τύπους εικονικών δικτύων

- Public networks
- Privet networks

Το Public network δημιουργείται από το διαχειριστή μέσω του snf-manager και μπορεί να χρησιμοποιηθεί από όλους τους τερματικούς χρήστες. Ένα Public networks μπορεί να ανήκει μόνο σε ένα Privet networks το Privet networks από την άλλη μπορεί να έχει πολλαπλά Public networks.

Από την σκοπιά της εικονικής μηχανής κάθε nic συνδέεται με ένα συγκεκριμένο δίκτυο.

Όταν μια εικονική μηχανή δημιουργείται ο back end allocator την παραθέτει σε ένα back end δίκτυο έπειτα το synnefo αναλαμβάνει να το δώσει στο πρώτο Public networks που έχει κενές θέσεις

Ένα εικονικό δίκτυο μπορεί να έχει τα παρακάτω στοιχεία.

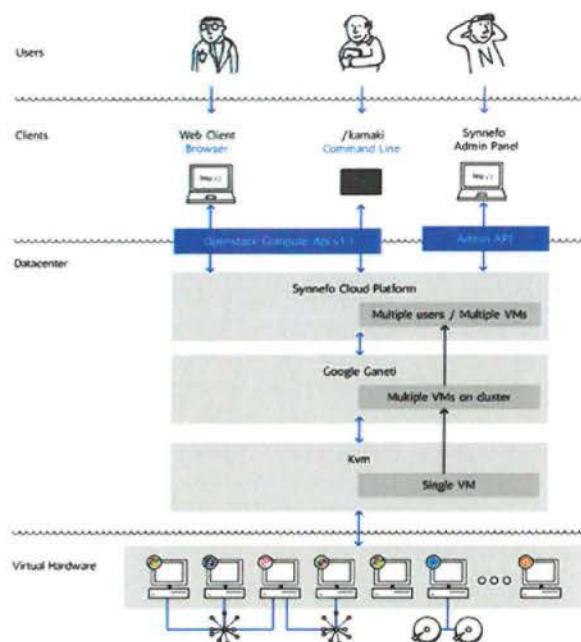
- IPV4 SUBNET
- IPV4 GATEWAY
- IPV6 SUBNET
- IPV6 GATEWAY
- PUBLIC/PRIVET FLAG
- FLAVOR

Για να εξασφαλιστεί το L2 απομόνωση, το Synnefo υποστηρίζει δύο διαφορετικούς μηχανισμούς .

- εκχωρεί ένα φυσικό vlan ανά δίκτυο
- εκχωρεί ένα MAC prefix ανά δίκτυο οπότε κάθε nic που εισάγεται στο δίκτυο έχει prefix.

Έχοντας αυτό κατά νου και για να αποφευχθούν δίπλα vlan και MAC prefix το synnefo έχει δύο ειδών pool.

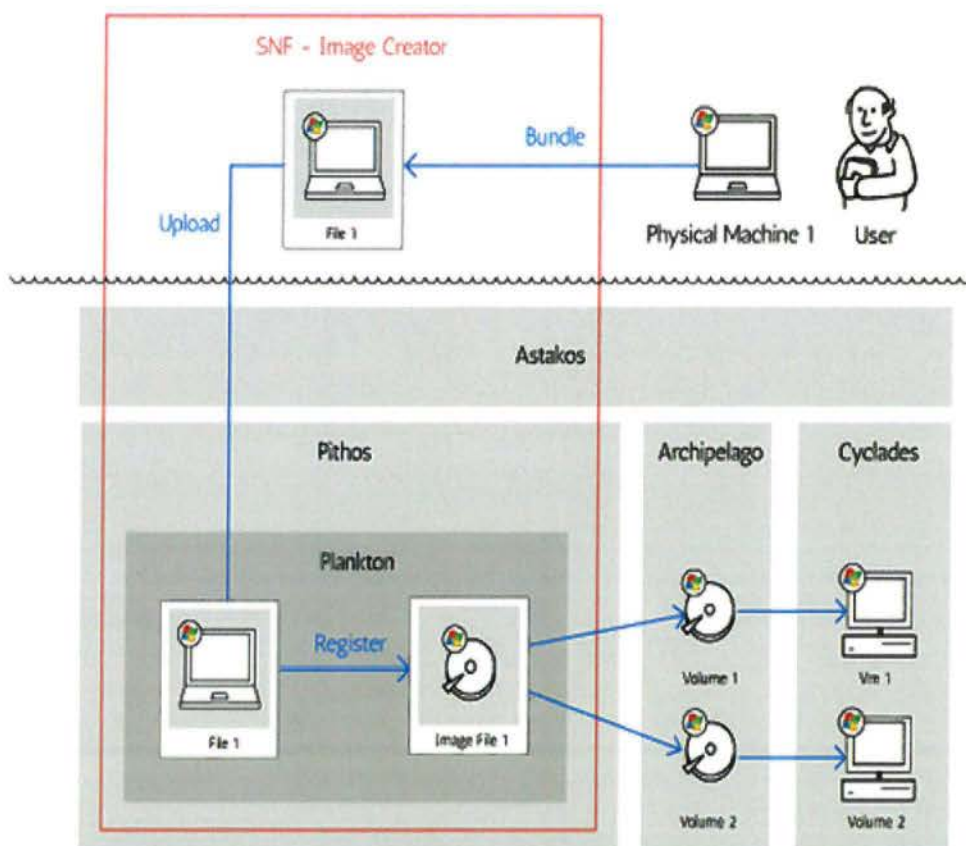
- Bridge Pool
- MAC prefix Pool



Υποστηριζόμενες δικτυακές εφαρμογές

Name	Mode	Link	MAC prefix	Tags
IP_LESS_ROUTED	routed	DEFAULT_ROUTING_TABLE	DEFAULT_MAC_PREFIX	'ip-less-routed'
MAC_FILTERED	bridged	DEFAULT_MAC_FILTERED_BRIDGE	pool	'private'filtered'
PHYSICAL_VLAN	bridged	pool	DEFAULT_MAC_PREFIX	'physical-vlan'
CUSTOM	bridged	DEFAULT_BRIDGE	DEFAULT_MAC_PREFIX	

3.5 Image Registry Service (plankton)



Το Plankton είναι το image registry service. Εφαρμόζεται σαν ένα λεπτό στρώμα στην κορυφή του pithos , κάθε image στο Plankton είναι αρχείο στο Pithos back end . Πρακτικά αυτό που κάνει το Plankton είναι να καταγράφει την κίνηση από και προς το σύστημά μας για τους χρήστες που είχε εξουσιοδοτημένοι. Η παρακάτω εικόνα δείχνει καλύτερα την κατάσταση σε αυτήν την εφαρμογή.

Το plankton ακόμα είναι κλειστού κώδικα αφού είναι σε φάση ανάπτυξης άρα μέχρι να τελειώσει η φάση αυτή δεν μπορούμε να έχουμε επιπλέον πληροφορίες.

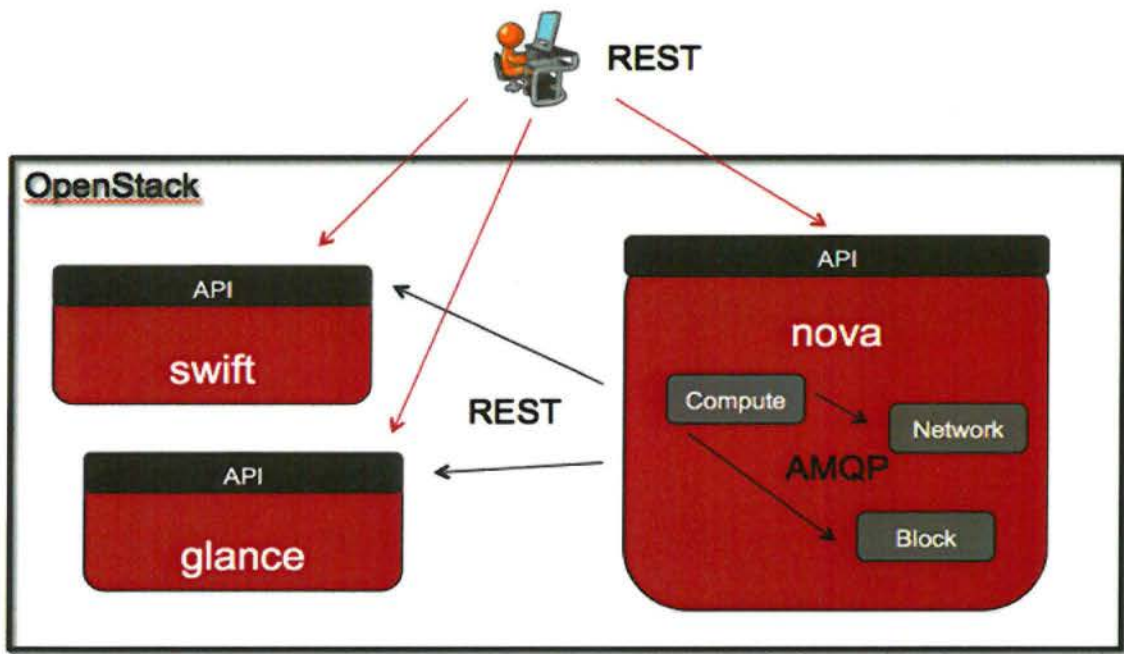
Είναι ένα λεπτό στρώμα πάνω από το storage cluster προσφέρει συγχρονισμό στην χρήση αρχείων , images στο Pithos χωρίς να αντιγραφούν τα ίδια τα αρχεία.

3.6 OpenStack

Το openstack project στοχεύει στην δημιουργία ενός ανοιχτού λογισμικού cloud computing platform , για ιδιωτικά και δημόσια δίκτυα με άμεσο στόχο την επεκτασιμότητα χωρίς να γίνεται πολύ πολύπλοκο. Η εφαρμογή αποτελείται από τρία κομμάτια

- OpenStack compute: Λογισμικό που συγχρονίζει, διαχειρίζεται και παράγει εικονικές μηχανές
- OpenStack object storage: Λογισμικό που αποθηκεύει τα δεδομένα των αντικειμένων.
- OpenStack image service: Λογισμικό που παρέχει ερωτήματα και υπηρεσίες αποθήκευσης για εικονικούς δίσκους

Communication Today



3.7 OpenStack object storage system

Το OpenStack object storage system είναι σχεδιασμένο να το χρησιμοποιούν διαφορετικού τύπου χρήστες. Κάθε χρήστης πρέπει να πιστοποιήσει τον εαυτό του χρησιμοποιώντας ένα authentication system (που στην δική μας περίπτωση είναι ο astakos). Αυτό το σύστημα θα μας παρέχει κάποιες παραμέτρους και μέσα σε αυτές θα είναι και ένα token το οποίο θα πιστοποιεί πως ο χρήστης έχει την δυνατότητα να εισαχθεί στο σύστημα. Αυτό το token πρέπει να χρησιμοποιείται προς όλα τα ερωτήματα στο σύστημα αφού το OOS δεν έχει κανενός τύπου πιστοποίηση χρηστών. Μπορούμε επίσης μέσα από αυτό το σύστημα πιστοποίησης και το token που παράγει να δώσουμε δικαιώματα εγγραφής και ανάγνωσης σε αντικείμενα και λογαριασμούς. Σε γενικές γραμμές κάθε χρήστης έχει πλήρη δικαιώματα στο λογαριασμό του και στα αντικείμενα που του ανήκουν, αλλά μόνο αν έχει εισέλθει στο σύστημα μπορεί να τα πειράξει. Ο μόνος τρόπος να πειράξει δεδομένα από άλλο χρήστη είναι να έχει μοιραστεί ένα session token ή ένα API access key.

Containers and Objects

Το container είναι ένα διαμέρισμα για τα δεδομένα και αποτελεί μία μέθοδο οργάνωσης των δεδομένων μας. Μία καλή παρομοίωση είναι τα folder στα windows ή τα directors στο Linux . Η βασική διαφορά ανάμεσα σε αυτά τα δύο είναι πως στο container δεν μπορούμε να έχουμε εμφωλευμένα container μέσα σε άλλα . Μπορούμε να έχουμε όσα θέλουμε και το ελάχιστο είναι ένα αφού όλα τα δεδομένα μας αποθηκεύονται σε αυτά. Ένας περιορισμός υπάρχει στο όνομα. Δεν μπορεί να ξεκινάει με "/" και πρέπει να είναι λιγότερο από 256 byte. Επίσης πρέπει να τονίσουμε πως μιλάμε για κωδικοποίηση σε μορφή url οπότε ένα container με την μορφή Nikos Vlassis πρακτικά έχει την μορφή Nikos%20vlassis οπότε θα πρέπει στα κενά να υπολογίζουμε επιπλέον χαρακτήρες. Ένα αντικείμενο είναι η βασική μονάδα αποθήκευσης και οποιαδήποτε μεταδομένο που αφορά τα δεδομένα μας στην ουσία περιγράφει ένα αντικείμενο. Ο μόνος περιορισμός στο όνομα είναι πως θα πρέπει να είναι το πολύ 1024 byte στο μέγεθος μετά από url κωδικοποίηση . Το μέγιστο όριο αποθήκευσης είναι 5 GB και το ελάχιστο 0 byte.

Απαιτήσεις συστήματος

Υλικό : Είναι σχεδιασμένο να τρέχει σε εμπορικό υλικό και επίσης δεν απαιτεί κάποιου τύπου raid, παρόλα αυτά προτείνεται η χρήση του για τους λόγους που έχουμε προαναφέρει.

- Λειτουργικό σύστημα : Το oos τρέχει σε Ubuntu σύστημα και αν χρειαστούμε την μεγαλύτερη έκδοση μπορούμε να το τρέξουμε στο Ubuntu XX.XX LTS.
- Δίκτυο : Η προτεινόμενη ταχύτητα είναι 1000 mbps , αυτό συμβαίνει γιατί πρέπει να υπάρχει ένα δίκτυο για να συνδέσει τον έξω κόσμο με των proxy server.
- Βάση δεδομένων : Για το oos θα χρειαστούμε μία sqlite βάση για τους χρήστες και τα containers .

Proxy server

Ο Proxy συνδέει τα υπόλοιπα κομμάτια του cluster, για κάθε ερώτημα που γίνεται κατά το container, την τοποθεσία του χρήστη, ή το αντικείμενο στο ring και δρομολογεί ερώτημα καταλλήλως. Ένας μεγάλος αριθμός από αστοχίες μπορεί να αντιμετωπιστεί από τον Proxy αφού αν παρατηρήσει πολλά ερωτήματα σε κάποιο κομμάτι του cluster θα ρωτήσει το ring πού βρίσκεται η συγκεκριμένη υπηρεσία. Όταν ένα αντικείμενο μοιράζεται προς ή από τον object server , μοιράζεται απευθείας από τον Proxy server.

The rings

Ορίζουν πού θα πρέπει να βρίσκεται μέσα στο cluster. Υπάρχει ένα ξεχωριστό ring για τις βάσεις των λογαριασμών , των containers , των αντικειμένων. Παρόλα αυτά δουλεύουν ακριβώς με τον ίδιο τρόπο. Τα rings χρησιμοποιούν έναν αριθμό από bits κωδικοποιημένο με md5 hash σαν ένα partition index που δείχνει την συσκευή στην οποία βρίσκεται. Η χρήση αυτής της τεχνικής επιτρέπει στα υπόλοιπα κομμάτια του cluster να δουλεύουν με δεσμίδες πράγμα πολύ πιο απλό από το να δούλευαν κάθε στοιχείο ξεχωριστά.

Object Server

Είναι ένας storage server που μπορεί να αποθηκεύσει, ανακτήσει , διαγράψει αντικείμενα που είναι αποθηκευμένα σε τοπικές συσκευές . Τα αντικείμενα αποθηκεύονται σαν binary στο σύστημα αρχείων και τα μεταδεδομένα αποθηκεύονται σαν επεκτάσεις των αρχείων. Τέλος κάθε αντικείμενο χρησιμοποιεί ένα μονοπάτι προερχόμενο από το ίδιο του το όνομα .

Μέθοδος Get και αντικείμενα : Η μέθοδος αυτή χρησιμοποιείται από τον Object Server για να ανακτήσουμε τα δεδομένα του αντικειμένου στο οποίο αναφερόμαστε.

Παράδειγμα χρήσης της get για ανάκτηση δεδομένων.

```
GET /<api version>/<account>/<container>/<object> HTTP/1.1
```

```
Host: storage.swiftdrive.com
```

```
X-Auth-Token: eaaafd18-0fed-4b3a-81b4-663c99ec1cbb
```

Στο παραπάνω παράδειγμα τα δεδομένα που ανακτώνται θα επιστραφούν στο βασικό κορμό της σελίδας και τα μεταδεδομένα σαν ένας HTTP headers αν επιστραφεί η τιμή 200 σημαίνει πως η ανάκτηση έγινε επιτυχώς ενώ με την 404 δεν βρέθηκε το αντικείμενο.

Από την άλλη για εγγραφή ή τροποποίηση ενός αντικειμένου χρησιμοποιείται η put σαν μέθοδος , μπορούμε να εξασφαλίσουμε επιτυχή αποθήκευση των δεδομένων του αντικειμένου προσθέτοντας μια ακολουθία MD5 checksum στο ETag header ούτως ώστε να εξασφαλίσουμε πως δεν θα υπάρξει κανένα σφάλμα κατά την μεταφορά. Επίσης έχουμε την δυνατότητα να ορίσουμε σε ένα αντικείμενο ημερομηνία λήξης χρησιμοποιώντας X-Delete-At η Xdelete-After, όταν το cloud βρει έναν τέτοιο χαρακτήρα στην κεφαλίδα , αυτόματα σταματά να σερβίρει το συγκεκριμένο αντικείμενο και μετά την ημερομηνία το διαγράφει. Επίσης υπάρχει η δυνατότητα να κρατάμε διαφορετικές εκδόσεις του ίδιου αντικειμένου, Είναι εύκολα κατανοητό πως παλιότερες εκδόσεις καλό

θα είναι να τις έχετε σε containers που οι χρήστες δεν θα έχουν πρόσβαση. Επιπλέον υπάρχει η δυνατότητα να αντιγράψουμε κάποιο αντικείμενο από ένα container σε κάποιο άλλο ή και να χρησιμοποιήσουμε αυτήν την τεχνική για να μετονομάσουμε ένα αντικείμενο χωρίς να χρειαστεί να το ανεβάσουμε εκ νέου και αυτό μπορεί να γίνει με δύο τρόπους.

- PUT με “XCopY-From” header.
- Απλά αντιγραφή του αντικειμένου.

Μέθοδος ένα :PUT /<api version>/<account>/<container>/<destobject> HTTP/1.1

Host: <storage URL>

X-Auth-Token: <some-auth-token>

X-CopY-From: /<container>/<sourceobject>

Content-Length: 0

Μέθοδος δύο :COPY /<api version>/<account>/<container>/<sourceobject> HTTP/1.1

Host: <storage URL>

X-Auth-Token: <some-auth-token>

Destination: /<container>/<destobject>

Διαγραφή αντικειμένου: Όπως καταλαβαίνουμε υποστηρίζεται η διαγραφή αντικειμένου από το storage system

Παράδειγμα διαγραφής αντικειμένου.

DELETE /<api version>/<account>/<container>/<object> HTTP/1.1

Host: storage.swiftdrive.com

X-Auth-Token: eaaafd18-0fed-4b3a-81b4-663c99ec1cbb

Επίσης έχουμε την δυνατότητα να ανακτήσουμε τα μεταδεδομένα των αρχείων. Το μόνο που απαιτείται σε αυτήν την περίπτωση είναι το authorization token.

HEAD /<api version>/<account>/<container>/<object> HTTP/1.1

Host: storage.swiftdrive.com

X-Auth-Token: eaaafd18-0fed-4b3a-81b4-663c99ec1cbb

αν επιστραφεί η τιμή 200 έχουμε επιτυχή ανάκτηση δεδομένων και την 404 όταν δεν υπάρχει το αντικείμενο.

Παράδειγμα ανταπόκρισης σε ένα τέτοιο ερώτημα.

HTTP/1.1 200 OK

Date: Thu, 07 Jun 2010 20:59:39 GMT

Server: Apache

Last-Modified: Fri, 12 Jun 2010 13:40:18 GMT

ETag: 8a964ee2a5e88be344f36c22562a6486

Content-Length: 512000

Content-Type: text/plain; charset=UTF-8

X-Object-Meta-Meat: Bacon

X-Object-Meta-Fruit: Bacon

X-Object-Meta-Veggie: Bacon

X-Object-Meta-Dairy: Bacon

Τέλος μπορούμε να κάνουμε αλλαγές στα μεταδεδομένα των αντικειμένων με τον εξής απλό τρόπο.

POST /<api version>/<account>/<container>/<object> HTTP/1.1

Host: storage.swiftdrive.com

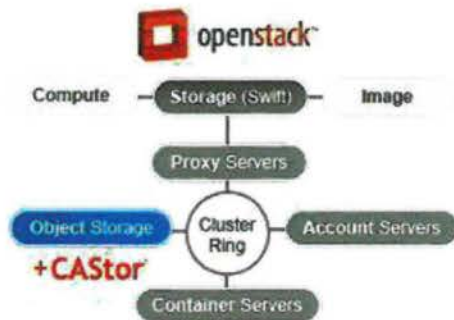
X-Auth-Token: eaaafd18-0fed-4b3a-81b4-663c99ec1cbb

X-Object-Meta-Fruit: Apple

X-Object-Meta-Veggie: Carrot

Container server

Η βασική δουλειά της συγκεκριμένης υπηρεσίας είναι να δημιουργεί λίστες αντικειμένων τα οποία δεν ξέρει πού βρίσκονται, το μόνο που την ενδιαφέρει είναι ποιά αντικείμενα βρίσκονται στο ίδιο container. Οι λίστες αυτές αποθηκεύονται σε sqlite βάσεις , και αντιγράφονται μέσα στο cluster ακριβώς όπως συμβαίνει και με τα αντικείμενα.



Account reaper

Ο Account reaper διαγράφει τα δεδομένα των χρηστών που έχει επιλογή για διαγραφή, πρακτικά όταν θέλουμε να διαγράψουμε έναν χρήστη στέλνουμε ένα αίτημα και στην βάση δεδομένων στο πεδίο

account_stat παίρνει την τιμή deleted, οπότε δρομολογείται και η διαγραφή των δεδομένων , αφού υποδεικνύεται στο σύστημα πως θα πρέπει να διαγραφούν, ο account reaper τώρα αυτό που κάνει πρακτικά είναι να ψάχνει στους server για λογαριασμούς προς διαγραφή, αφού πρώτα βρει το μηχανήμα στο οποίο ο λογαριασμός είναι στον πρωτεύοντα κόμβο έπειτα ανακαλύπτει και τα αντίγραφα ούτως ώστε να τα διαγράψει.

Χρήση της υπηρεσίας μέσω της cURL

Είναι ένα command line εργαλείο που υπάρχει στα περισσότερα Unix-like συστήματα και μπορεί να προστεθεί στα windows , σου επιτρέπει να στέλνεις και να λαμβάνεις http αιτήματα και ταυτόχρονα να αποκρίνεσαι μέσω κάποιου τερματικού ή μέσω κάποιου script. Αυτό πρακτικά δίνει την δυνατότητα να επικοινωνήσουμε με την εφαρμογή χωρίς να χρησιμοποιήσουμε κάποιο client λογισμικό.

Πιστοποίηση με cURL : Πρώτα απ' όλα για να μπορέσουμε να χρησιμοποιήσουμε κάποια από τις υπηρεσίες του openstack μέσω cURL εντολών θα πρέπει να εισέλθουμε στο σύστημα. Αυτό γίνεται δίνοντας του το token και το όνομα χρήστη που του δίνει και οποιαδήποτε άλλη εφαρμογή.

```
curl -i \  
-H "X-Auth-Key: jdoesecretpassword" \  
-H "X-Auth-User: jdoe" \  
https://auth.api.yourcloud.com/v1.0
```

Επίσης μέσα απο cURL μπορούμε να καθορίσουμε την χρήση για τον χώρο αποθήκευσης : Μπορούμε δηλαδή να δούμε πόσα δεδομένα έχεις στο storage πόσα container και με την χρήση της -i παραμέτρου βγάζουμε το αποτέλεσμα στην οθόνη.

```
curl -X HEAD -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343
```

Για να πάρουμε την λίστα με τα container που μας ανήκουν μπορούμε να χρησιμοποιήσουμε τις παρακάτω εντολές.

```
curl -X GET -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343
```

Ενώ με παραπλήσιο τρόπο μπορούμε να δημιουργήσουμε container.

```
curl -X PUT -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343/george
```

δημιουργία λίστας αντικειμένων

```
curl -X GET -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343/dogs?limit=2
```

Ανάκτηση αντικειμένου

```
curl -X GET
```

```
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF_xer7_343/dogs/JingleRocky.jpg >  
JingleRocky.jpg
```

Αντιγραφή αντικειμένου από την πλευρά του server

```
curl -X PUT -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
-H "X-Copy-From: /dogs/JingleRocky.jpg" \  
-H "Content-Length: 0" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343/elaine/JingleRocky.jpg
```

Δημιουργία μεταδεδομένων για τα αντικείμενα

```
curl -X POST -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
-H "X-Object-Meta-Breed: Terrier pit bull mix" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343/dogs/JingleRocky.jpg
```

Ανάκτηση μεταδεδομένων από τα αντικείμενα.

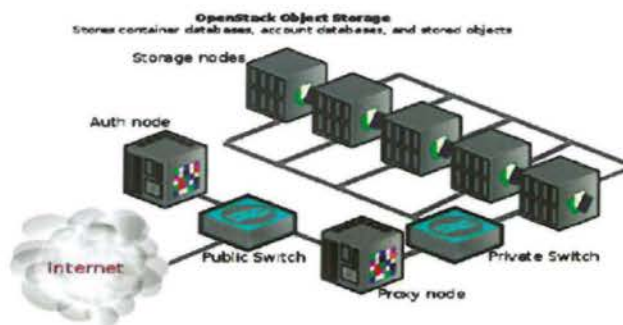
```
curl -X HEAD -i \  
-H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" \  
https://storage.swiftdrive.com/v1/CF\_xer7\_343/dogs/JingleRocky.jpg
```

Αυτές ήταν λίγες εφαρμογές της cURL στο OpenStorage. Η ουσία είναι πως μέσα από αυτό το εργαλείο μπορούμε να αλληλεπιδράμε με την εφαρμογή μας χωρίς να έχουμε μεγάλες απαιτήσεις. Βέβαια αυτό που θα ήθελα να κρατήσουμε για την συνέχεια είναι πως το σύστημα αυτό τρέχει σε ubuntu .

Εγκατάσταση OpenStack Object Storage on Ubuntu

Για να εγκαταστήσουμε το OpenStack πρέπει να έχουμε

- Έναν κόμβο στον οποίο θα εγκαταστήσουμε το swift-proxy-server για να λειτουργεί σαν proxy server.
- Έναν κόμβο που θα τρέχει το swift-auth-server και θα λειτουργεί σαν authentication server.
- Πέντε κόμβους που θα τρέχουν swift-account-server, swift-container-server και swiftobject-server για να ελέγχουν λογαριασμούς αντικείμενα, containers.



Ενα παράδειγμα τέτοιας αρχιτεκτονικής

Αντιμετώπιση προβλημάτων

Για το openstack object storage όλα είναι καταγεγραμμένα στο `/var/log/syslog`. Βέβαια υπάρχουν και παράμετροι που επιτρέπουν την δημιουργία custom log files `log_name, log_facility, log_level`.

Αντιμετώπιση προβλημάτων με σκληρό δίσκο

Αν ο δίσκος μπορεί να αντικατασταθεί απευθείας είναι η καλύτερη επιλογή, αν παρόλα αυτά για τον έναν ή τον άλλο λόγο αυτό δεν είναι εφικτό τότε καλό θα είναι να τον κάνουμε unmounted για να συνεχίσει το σύστημά μας να παίζει με τα αντίγραφα ασφαλείας ωστόσο αντικαταστήσουμε τον δίσκο και δημιουργηθούν τα δεδομένα πάλι.

Αντιμετώπιση προβλημάτων με server

Αν κάποιο μηχάνημα παρουσιάσει κάποιο σφάλμα καλό είναι να ελέγξουμε αν παίζουν οι υπηρεσίες του openstack object storage. Αν παίζει μπορεί η υπηρεσία να δουλεύει όσο θα διορθώνουμε το λάθος, αν όχι καλό είναι να το αντιμετωπίσουμε άμεσα έτσι ώστε να επαναφέρουμε γρήγορα την εφαρμογή μας.

3.8 Google Ganeti

Το ganeti είναι λογισμικό που χρησιμοποιείται για την διαχείριση ενός cluster δημιουργημένο από την google, συνδυάζει visualization και data replication και οι απαιτήσεις του ganeti είναι : python, dr:db, lvm, kvm η lxc.

Οι δυνατότητες του ganeti είναι :

- Διαχείριση cluster φυσικών μηχανών.
- Ανάπτυξη εικονικών μηχανών
- Live
- Ανθεκτικότητα σε αστοχίες
- Cluster blancing
- Ευκολία στην διόρθωση

Κάνοντας το virtualization σε όσο το δυνατό χαμηλότερο επίπεδο κερδίζουμε:

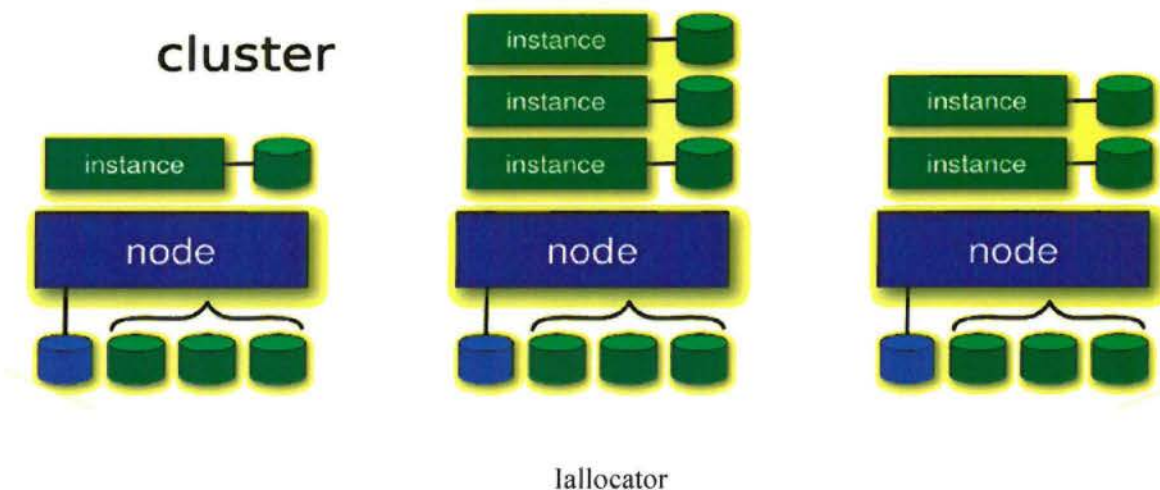
- Ευκολία στην διαχείριση.
- Δεν χρειαζόμαστε εξειδικευμένο υλικό.
- Ταχύτητα

Ganeti terminology

Node : Είναι το κύτταρο ενός cluster, είναι ένα μηχάνημα που παίρνει μέρος σε ένα δίκτυο τέτοιου τύπου.

Υπάρχουν 3 βασικά είδη :

- Master node : Είναι ο κόμβος που ελέγχει το cluster.
- Master candidate node : Μόνο οι κόμβοι αυτού του επιπέδου έχουν την δυνατότητα να γίνουν Master αφού μόνο αυτοί έχουν τα cluster configuration.
- Regular node : Δεν μπορεί να γίνει master γιατί έχει μόνο ένα μικρό κομμάτι από τα configuration.
- Drained node : Οι κόμβοι σε αυτήν την κατηγορία έχουν κάποιο πρόβλημα και επιδιορθώνονται.
- Offline node : Ελέγχει την λειτουργία των κόμβων που είναι εγκατεστημένοι αλλά ο δαίμονας δεν μπορεί να επικοινωνήσει μαζί τους.
- Instance: ένας επισκέπτης σε εικονική μηχανή.



Ένα framework που χρησιμοποιεί εξωτερικά script για να υπολογίσουμε την τοποθέτηση των instances στους κόμβους του cluster. Αυτή η τεχνική εξαλείφει την ανάγκη να τα ορίσουμε με το χέρι . Για να μπορέσει το ganeti να χρησιμοποιήσει τα συγκεκριμένα script πρέπει να τοποθετηθούν στο (lib/ganeti/iallocator)

“Πρωτεύοντες” και “Δευτερεύοντες” κόμβοι

Το instances έχει την δυνατότητα να έχει πρωτεύοντες αλλά και δευτερεύοντες κόμβους, συνήθως τρέχει στον πρωτεύοντα, ενώ τον δευτερεύοντα τον έχει για τα αντίγραφα των δίσκων.

Εντολές διαχείρισης instances

Εκ πρώτης όψεως φαίνεται δύσκολη διαδικασία παρόλα αυτά αν καταλάβεις τις παραμέτρους που χρειάζεται είναι κάτι απλό και δεν θέλει πολύ χρόνο. Οι παράμετροι που χρειάζεται είναι οι παρακάτω.

- Το λειτουργικό σύστημα που θα τρέξει.
- Το template του δίσκου.
- Η χωρητικότητα του δίσκου και η θέση του στο storage.
- Ο κόμβος που θα το τρέξει και σε εναλλακτική το iallocator.
- Και τέλος το όνομά του.

Το λειτουργικό σύστημα θα πρέπει να είναι ορατό με την `gnt-os list`.

Το template του δίσκου ορίζει τον τύπο του storage που χρησιμοποιείται στο backend..

Τέλος ο κόμβος στον οποίο τρέχει μπορεί να βρεθεί με την παράμετρο `-v`.

Ένα παράδειγμα μιας τέτοιας εντολής παραθέτεται παρακάτω.

```
gnt-instance add \  
-n TARGET_NODE:SECONDARY_NODE \  
-o OS_TYPE \  
-t DISK_TEMPLATE -s DISK_SIZE \  
INSTANCE_NAME
```

Επίσης υπάρχουν και άλλοι παράμετροι που είναι προαιρετικές

Το μέγεθος της μνήμης (`-B memory`).

Ο αριθμός των εικονικών cpu (`-B vcpus`).

Διαγραφή instances : `$ gnt-instance remove INSTANCE_NAME`

Έναρξη instances : `$ gnt-instance startup INSTANCE_NAME`

Απενεργοποίηση instances : `$ gnt-instance shutdown INSTANCE_NAME`

Υπάρχουν δύο τρόποι για να αντλήσουμε πληροφορίες από τα instances.

A)Μαζικά παίρνοντας δηλαδή μία λίστα από τα instances που υπάρχουν και κάποιες πληροφορίες για το καθένα

B) Ζητώντας πιο εξειδικευμένες πληροφορίες χρησιμοποιώντας το όνομά του.

- `$ gnt-instance list`
- `$ gnt-instance info INSTANCE`

Μία άλλη δυνατότητα που έχει το ganeti είναι πως μπορεί να κρατήσει στιγμιότυπα, αυτό μπορεί να φανεί χρήσιμο για δύο λόγους

A) backup του instance

B)εξαγωγή του σε άλλο κόμβο.

- **Εξαγωγή στιγμιότυπου:** `$ gnt-backup export -n TARGET_NODE INSTANCE_NAME`

- **Δημιουργία αντίγραφου:** `$ gnt-backup import -n TARGET_NODE \ --src-node=NODE --src-dir=DIR INSTANCE_NAME`

Xen PVM

Με το Xen PVM έχουμε τρεις επιλογές.

Πρώτον μπορούμε να χρησιμοποιήσουμε τον kernel του κόμβου θέτοντάς του διάφορες παραμέτρους.

- `kernel_path` : θα δείχνει ένα έγκυρο αρχείο μέσα στον κόμβο
- `root_path` : Π.χ. `/dev/xvda1`

Εναλλακτικά μπορούμε να χρησιμοποιήσουμε το `pvgrub` αλλά για να συμβεί αυτό θα πρέπει:

- Το `kernel_path` να δείχνει στο `pvgrub` (π.χ. `/usr/lib/xen/boot/pv-grub-x86_32.gz`)
- Το `kernel_args` να δείχνει στο configuration file του grub (π.χ. `(hd0,0)/grub/menu.lst`)
- Το `root_path` να είναι κενό
- Το `bootloader_path` και το `bootloader_args` να είναι επίσης άδεια.

Η βασική ρύθμιση για το `pygrub` είναι το `bootloader_path` να δείχνει στο binary αρχείο (π.χ. `/usr/bin/pygrub`).

Αλλαγή του πρωτεύοντα κόμβου

Αλλαγή του πρωτεύοντα και του δευτερεύοντα κόμβου μπορεί να γίνει για διάφορους λόγους. Ο κυριότερος από αυτούς είναι πως ο πρωτεύων κόμβος συνήθως αποτυγχάνει και το `instances` τρέχει πάντα από το δευτερεύοντα. Για να αλλάξει αυτό λοιπόν είναι πολύ εύκολο, η παρακάτω εντολή αλλάζει τον δευτερεύοντα κόμβο και τον κάνει πρωτεύοντα.

```
$ gnt-instance failover INSTANCE_NAME
```

Επίσης αν το disk template είναι `rbd` μπορούμε να ορίσουμε τον κόμβο που θα παίζει πλέον.

```
$ gnt-instance failover -n TARGET_NODE INSTANCE_NAME
```

Live migration : αν το `instances` τρέχει άρα δεν θέλουμε να το σταματήσουμε έτσι ώστε να κάνουμε migration. Μπορούμε αυτήν την διαδικασία να την κάνουμε μέσα από το δευτερεύοντα κόμβο.

```
$ gnt-instance migrate INSTANCE_NAME
```

Εντολές για την διαχείριση του δίσκου

Όπως σε κάθε σύστημα έτσι και εδώ ο σκληρός δίσκος είναι ένα κρίσιμο κομμάτι για την ακεραιότητα της εφαρμογής μας, για αυτό τον λόγο το `ganeti` παρέχει την δυνατότητα διαχείρισης καταστάσεων ούτως ώστε το σύστημά μας να μην πέφτει σε περίπτωση που κάτι πάει στραβά με κάποιο δίσκο.

Στο σημείο αυτό πρέπει να πούμε πως για να μπορέσει το `ganeti` να διαχειριστεί τις παραπάνω καταστάσεις θα πρέπει να έχει εγκατεστημένο το LVM και δεν θα πρέπει αυτό να επιτρέπει κάποιο λάθος.

Λογικά αφού εγκαταστήσουμε το LVM η έξοδος της εντολής θα είναι η παρακάτω.

```
$ vgs
```

```
/dev/sdb1: read failed after 0 of 4096 at 0: Input/output error
```

```
/dev/sdb1: read failed after 0 of 4096 at 750153695232: Input/output error
```

```
/dev/sdb1: read failed after 0 of 4096 at 0: Input/output error
```

```
Couldn't find device with uuid 't30jmN-4Rcf-Fr5e-CURS-pawt-z0jU-mITgeJ'.
```

```
Couldn't find all physical volumes for volume group xenvg.
```

Παρατηρούμε πως δεν εκτελείται σωστά. Για να διορθωθεί αυτό πληκτρολογούμε την ακόλουθη εντολή.

```
$ vgreduce --removemissing VOLUME_GROUP
```

Αν και μετά από αυτό ο χαλασμένος δίσκος συνεχίσει να φαίνεται τότε πρέπει να τον απενεργοποιήσουμε χρησιμοποιώντας την παρακάτω εντολή.

```
$ pvs -x n /dev/DISK
```

Αποκατάσταση σε DRBD-based instances

Ανάλογα με τον κόμβο που έχει πέσει έχουμε τρεις επιλογές για να χειριστούμε την κατάσταση αυτή:

Αν έχει πέσει ο πρωτεύων κόμβος τότε πρέπει να δημιουργήσουμε ξανά τον δίσκο.

Αν έχει πέσει ο δευτερεύων τότε έχουμε δύο επιλογές: ή να ξαναδημιουργήσουμε τον δίσκο ή να αλλάξουμε κόμβο και να δημιουργήσουμε εκεί τον δίσκο εκ νέου.

- Δημιουργία του δίσκου στον πρωτεύοντα κόμβο : `$ gnt-instance replace-disks -p INSTANCE_NAME`
- Δημιουργία του δίσκου στο δευτερεύοντα κόμβο : `$ gnt-instance replace-disks -s INSTANCE_NAME`
- Αλλαγή δευτερεύοντα κόμβου με το χέρι : `$ gnt-instance replace-disks -n NODE INSTANCE_NAME`
- Αλλαγή δευτερεύοντα κόμβου με το iallocator scrip: `$ gnt-instance replace-disks -I SCRIPT INSTANCE_NAME`
- Αυτόματη διόρθωση (από την έκδοση 2.1 και έπειτα): `$ gnt-instance replace-disks -a INSTANCE_NAME`
-

Πρόσβαση σε instance δίσκους

Για να έχουμε πρόσβαση σε δίσκους ή google έχουμε φροντίσει να υπάρχει γραμμή εντολών. Καλό είναι να χρησιμοποιείται αυτή γιατί, αν για παράδειγμα προσπαθήσουμε να κάνουμε mount τον δίσκο με το χέρι υπάρχει κίνδυνος να σπάσουμε την σύνδεση με τα αντίγραφα και τα δεδομένα μας να παρουσιάζουν ασυνέπεια.

```
$ gnt-instance activate-disks instance1
```

```
node3:disk/0:...
```

```
$ ssh node3
```

```
$ kpartx -l /dev/...
```

```
$ kpartx -a /dev/...
```

```
$ mount /dev/mapper/... /mnt/
```

```
$ umount /mnt/  
$ kpartx -d /dev/...  
$ exit
```

Εντολές διαχείρισης instance

Για να τρέξουμε την κονσόλα ενώ το σύστημά μας τρέχει πρέπει να δώσουμε την παρακάτω εντολή.

```
$ gnt-instance console INSTANCE_NAME
```

Επιπλέον παρέχεται εργαλείο για να διαγνώσουμε τυχόν λάθη στο λειτουργικό σύστημα .

```
$ gnt-os diagnose
```

Για να μεταφέρουμε ένα instance και από πρωτεύον αλλά και από δευτερεύον σε άλλους πρέπει να το κάνουμε σταδιακά.

```
$ gnt-instance replace -n nodeC instance1
```

```
$ gnt-instance migrate instance1
```

```
$ gnt-instance replace -n nodeD instance1
```

Για αρχή βρίσκεται στον κόμβο A, B στο πρώτο βήμα απλά το μεταφέρουμε στο A.C. Κάνουμε τον C πρωτεύοντα κόμβο και μετά απλά αντικαθιστούμε τον A με τον D.

Εντολές διαχείρισης κόμβων

Είναι αρκετά εύκολο να επεκτείνουμε το cluster μας προσθέτοντας και άλλους κόμβους, αρκεί μία εντολή.

```
$ gnt-node add NEW_NODE
```

Από την άλλη, αν αυτός ο κόμβος υπήρχε και για κάποιο λόγο σταμάτησε να υπάρχει τότε μπορούμε να τον αναδημιουργήσουμε μέσα από το αντίγραφό του.

```
$ gnt-node add --readd EXISTING_NODE
```

Όπως είδαμε και πιο πάνω οι κόμβοι χωρίζονται σε ομάδες (master, Master candidate node, regular node κ.α.) και έχουμε την δυνατότητα να τους αλλάζουμε αρμοδιότητα:

Αν θέλουμε να κάνουμε έναν κόμβο master :

```
$ gnt-cluster master-failover
```

Αν θέλουμε να κάνουμε έναν κόμβο υποψήφιο master :

```
$ gnt-node modify -C yes NODE
```

Αν θέλουμε να βγάλουμε εκτός λειτουργίας έναν κόμβο :

```
$ gnt-node modify -O yes NODE
```

Αν θέλουμε να κάνουμε έναν κόμβο regular :

```
$ gnt-node modify -O no -D no -C no NODE
```

Εδώ πρέπει να τονίσουμε πως ανά πάσα ώρα και στιγμή πρέπει να υπάρχει υποψήφιος master σε περίπτωση που αυτός πέσει, άρα ενδεχόμενη αλλαγή ενός τέτοιου κόμβου μπορεί και να μην είναι εφικτή.

Διαγραφή κόμβου:

```
$ gnt-node remove NODE_NAME
```

Αποσφαλμάτωση του storage-System

Όταν μιλάμε για μεγάλα cloud με πολλούς κόμβους γίνεται κουραστική η διαδικασία να βρεις τα λάθη με το χέρι και να τα διορθώσεις. Η google για να κάνει την ζωή μας πιο όμορφη έχει φτιάξει μερικές εντολές που

κάνουν αυτή την λειτουργία αυτόματα.

Logical volume : Είναι μία εντολή για την διαχείριση του LVM . Απαριθμεί τους λογικούς τόμους σε ένα κόμβο και την ένωσή τους.

```
$ gnt-node volumes
```

Node	PhysDev	VG	Name	Size	Instance
node1	/dev/sdb1	xenvg	e61fbc97-....disk0	512M	instance17
node1	/dev/sdb1	xenvg	ebd1a7d1-....disk0	512M	instance19
node2	/dev/sdb1	xenvg	0af08a3d-....disk0	512M	instance20
node2	/dev/sdb1	xenvg	cc012285-....disk0	512M	instance16
node2	/dev/sdb1	xenvg	f0fac192-....disk0	512M	instance18

Γενικευμένη αποσφαλμάτωση του storage system

Ξεκινώντας από το ganeti 2.1 προστέθηκε ένα νέο framework το οποίο βοηθά στην αποσφαλμάτωση του storage.

Για αρχή μπορεί να καταγράψει το υλικό αποθήκευσης και την κατάσταση στην οποία βρίσκονται (αποθηκευτικές δυνατότητες , ελεύθερος χώρος και χώρος που ήδη χρησιμοποιείται).

```
$ gnt-node list-storage
```

Node	Name	Size	Used	Free
node1	/dev/sda7	673.8G	0M	673.8G
node1	/dev/sdb1	698.6G	1.5G	697.1G
node2	/dev/sda7	673.8G	0M	673.8G
node2	/dev/sdb1	698.6G	1.0G	697.6G

Επίσης μπορεί να επιστρέψει και λίστες με LVM ομάδες :

```
$ gnt-node list-storage -t lvm-vg
```

Node	Name	Size
node1	xenvg	1.3T
node2	xenvg	1.3T

Εντολές διαχείρισης cluster

Μία από τις πιο βασικές εντολές για την διαχείριση του cluster είναι αυτή που μας επιστρέφει τον master κόμβο.

```
$ gnt-cluster getmaster
```

```
node1.example.com
```

Είναι επίσης εφικτό να ζητήσουμε αλλά και να τροποποιήσουμε βασικά χαρακτηριστικά του cluster.

```
$ gnt-cluster info
```

Cluster name: cluster.example.com
Cluster UUID: 07805e6f-f0af-4310-95f1-572862ee939c
Creation time: 2009-09-25 05:04:15
Modification time: 2009-10-18 22:11:47
Master node: node1.example.com
Architecture (this node): 64bit (x86_64)
Tags: foo
Default hypervisor: xen-pvm
Enabled hypervisors: xen-pvm
Hypervisor parameters:
- xen-pvm:
 root_path: /dev/sda1
Cluster parameters:
- candidate pool size: 10
Default instance parameters:
- default:
 memory: 128
Default nic parameters:
- default:
 link: xen-br0

Για να τροποποιήσουμε τώρα κάποια από αυτά τα χαρακτηριστικά μπορούμε να χρησιμοποιήσουμε τις παρακάτω παραμέτρους.

- **Hypervisor:** modify -H xen-pvm:root_path=...,
- **Προεπιλεγμένες παραμέτρους :** modify -Bparameter=value...
- **Ενώ οι παράμετροι του cluster αλλάζουν ξεχωριστές επιλογές της modify :** Π.χ. --candidate-pool-size.

Επίσης υπάρχει η δυνατότητα να τρέξουμε εντολές του λειτουργικού συστήματος με την χρήση του dsh πράγμα που είναι χρήσιμο ούτως ώστε να αναβαθμίσουμε διάφορα script άρα και την διαχείριση ένα τέτοιο παράδειγμα παρατίθεται παρακάτω:

```
$ gnt-cluster command ls -l /path/to/file
```

Επαλήθευση του cluster

Υπάρχουν τρεις βασικοί τρόποι για να επαληθεύσουμε την ορθή λειτουργία ενός cluster: ο πρώτος από αυτούς είναι η verify που δίνει μία εικόνα για την κατάσταση του cluster τονίζοντας τυχόν προβλήματα. Κάτω από φυσιολογικές συνθήκες η εντολή αυτή θα έχει τα παρακάτω αποτελέσματα:

```
$ gnt-cluster verify
```

```
Sun May 15 23:08:58 2013 * Verifying global settings
```

```
Sun May 15 23:08:58 2013 * Gathering data (2 nodes)
```

```
Sun May 15 23:09:00 2013 * Verifying node status
```

```
Sun May 15 23:09:00 2013 * Verifying instance status
```

```
Sun May 15 23:09:00 2013 * Verifying orphan volumes
```

```
Sun May 15 23:09:00 2013 * Verifying remaining instances
```

```
Sun May 15 23:09:00 2013 * Verifying N+1 Memory redundancy
```

```
Sun May 15 23:09:00 2013 * Other Notes
```

```
Sun May 15 23:09:00 2013 - NOTICE: 5 non-redundant instance(s) found.
```

```
Sun May 15 23:09:00 2013 * Hooks Results
```

Η δεύτερη εντολή είναι η `verify-disks` που ελέγχει πως οι δίσκοι τρέχουν χωρίς κανένα πρόβλημα.

```
$ gnt-cluster verify-disks
```

Εδώ να σημειώσουμε πως αυτή η εντολή δεν έχει έξοδο αν ο δίσκος δεν αντιμετωπίζει κανένα σφάλμα.

Αν η εντολή `verify` καταγγείλει αναντιστοιχία αρχείων ανάμεσα στον master και σε άλλους κόμβους, λόγω κάποιου λάθους από χειροκίνητη διαχείριση ή κάποιας αστοχίας στο υλικό, μπορούμε να προωθήσουμε τις παραμέτρους του master μέσω της `redis-conf` εντολής.

```
$ gnt-cluster redis-conf
```

Αλλαγή ονόματος στο cluster

```
$ gnt-cluster rename cluster.example.com
```

Αυτή η εντολή θα μετονομάσει το cluster σε `cluster.example.com`

Ουρά εργασιών: Από το `ganeti 2` και έπειτα υποστηρίζεται ουρά εργασιών. Η διαχείριση γίνεται με την εντολή `queue` και μερικά παραδείγματα παρατίθενται παρακάτω.

```
$ gnt-cluster queue info
```

```
The drain flag is unset
```

```
$ gnt-cluster queue drain
```

```
$ gnt-instance stop instance1
```

```
Failed to submit job for instance1: Job queue is drained, refusing job
```

```
$ gnt-cluster queue info
```

```
The drain flag is set
```

```
$ gnt-cluster queue undrain
```

Ganeti-watcher

Το `ganeti-watcher` είναι μια εφαρμογή που συνήθως προγραμματίζεται μέσω `cron` και κρατάει τις κινήσεις του cluster. Η απενεργοποίηση του από `cron` job δεν είναι και τόσο καλή ιδέα αφού από εκεί και έπειτα δεν θα

έχουμε εικόνα για το τι γίνεται στο σύστημά μας. Μερικά παραδείγματα εντολών του `ganeti-watcher`:

```
$ gnt-cluster watcher info : Επιστρέφει πληροφορίες για την τρέχουσα κατάσταση του.
```

```
$ gnt-cluster watcher pause 1h :Διακόπτει την λειτουργία του.
```

Εγκατάσταση ενός συστήματος με google ganeti

Μετά από μερικά παραδείγματα στην διαχείριση ενός cluster με την εφαρμογή google ganeti ήρθε η ώρα να εγκαταστήσουμε ένα cluster.

Για αρχή πρέπει να εγκαταστήσουμε ένα cluster και να του δώσουμε ip και όνομα.

```
gnt-cluster init -s 192.0.2.1 --enabled-hypervisors=xen-pvm example-cluster
```

Αν η εντολή εκτελεστεί επιτυχώς τότε ελέγχουμε τους κόμβους για να δούμε πως έχει εγκατασταθεί σωστά ο πρώτος κόμβος.

```
$ gnt-node list
```

```
Node DTotal DFree MTotal MNode MFree Pinst Sinst
```

```
node1 1.3T 1.3T 32.0G 1.0G 30.5G 0 0
```

```
$ gnt-cluster verify
```

```
Mon May 16 02:08:51 2013 * Verifying global settings
```

```
Mon May 16 02:08:51 2013 * Gathering data (1 nodes)
```

```
Mon May 16 02:08:52 2013 * Verifying node status
```

```
Mon May 16 02:08:52 2013 * Verifying instance status
```

```
Mon May 16 02:08:52 2013 * Verifying orphan volumes
```

```
Mon May 16 02:08:52 2013 * Verifying remaining instances
```

```
Mon May 16 02:08:52 2013 * Verifying N+1 Memory redundancy
```

```
Mon May 16 02:08:52 2013 * Other Notes
```

```
Mon May 16 02:08:52 2013 * Hooks Results
```

Στην συνέχεια μπορούμε να εγκαταστήσουμε επιπλέον δύο κόμβους.

```
$ gnt-node add -s 192.0.2.2 node2
```

```
-- WARNING --
```

Performing this operation is going to replace the ssh daemon keypair on the target machine (node2) with the ones of the current one and grant full intra-cluster ssh root access to/from it

Unable to verify hostkey of host xen-devi-5.fra.corp.google.com:

```
f7:.... Do you want to accept it?
```

```
y/[n]/?: y
```

```
Mon May 16 02:11:53 2009 Authentication to node2 via public key failed, trying password
```

root password:

Mon May 16 02:11:54 2009 - INFO: Node will be a master candidate

\$ gnt-node add -s 192.0.2.3 node3

-- WARNING --

Performing this operation is going to replace the ssh daemon keypair on the target machine (node3) with the ones of the current one and grant full intra-cluster ssh root access to/from it

Mon May 16 02:12:43 2009 - INFO: Node will be a master candidate

Σε αυτό το σημείο ήρθε η ώρα να εγκαταστήσουμε τα instances .

\$ gnt-instance add -t drbd -o debootstrap -s 256m -n node1:node2 instance3

Mon Oct 26 04:06:52 2009 - INFO: Selected nodes for instance instance1 via iallocator hail: node2, node3

Mon May 16 04:06:53 2013 * creating instance disks...

Mon May 16 04:06:57 2013 adding instance instance1 to cluster config

Mon May 16 04:06:57 2013 - INFO: Waiting for instance instance1 to sync disks.

Mon May 16 04:06:57 2013 - INFO: - device disk/0: 20.00% done, 4 estimated seconds remaining

Mon May 16 04:07:01 2013 - INFO: Instance instance1's disks are in sync.

Mon May 16 04:07:01 2013 creating os for instance instance1 on node node2

Mon May 16 04:07:01 2013 * running the instance OS create scripts...

Mon May 16 04:07:14 2013 * starting instance...

\$ gnt-instance add -t drbd -o debootstrap -s 256m -n node1:node2 instance2

Mon May 16 04:11:37 2013 * creating instance disks...

Mon May 16 04:11:40 2013 adding instance instance2 to cluster config

Mon May 16 04:11:41 2013 - INFO: Waiting for instance instance2 to sync disks.

Mon May 16 04:11:41 2013 - INFO: - device disk/0: 35.40% done, 1 estimated seconds remaining

Mon May 16 04:11:42 2013 - INFO: - device disk/0: 58.50% done, 1 estimated seconds remaining

Mon May 16 04:11:43 2013 - INFO: - device disk/0: 86.20% done, 0 estimated seconds remaining

Mon May 16 04:11:44 2013 - INFO: - device disk/0: 92.40% done, 0 estimated seconds remaining

Mon May 16 04:11:44 2013 - INFO: - device disk/0: 97.00% done, 0 estimated seconds remaining

Mon May 16 04:11:44 2013 - INFO: Instance instance2's disks are in sync.

Mon May 16 04:11:44 2013 creating os for instance instance2 on node node1

Mon May 16 04:11:44 2013 * running the instance OS create scripts...

Mon May 16 04:11:57 2013 * starting instance...

Για να συνδεθούμε σε ένα instances πρέπει να πληκτρολογήσουμε την παρακάτω εντολή.

```
$ gnt-instance console instance2
```

```
[ 0.000000] Bootdata ok (command line is root=/dev/sda1 ro)
```

```
[ 0.000000] Linux version 2.6...
```

```
[ 0.000000] BIOS-provided physical RAM map:
```

```
[ 0.000000] Xen: 0000000000000000 - 0000000008800000 (usable)
```

```
[13138176.018071] Built 1 zonelists. Total pages: 34816
```

```
[13138176.018074] Kernel command line: root=/dev/sda1 ro
```

```
[13138176.018694] Initializing CPU#0
```

```
...
```

```
Checking file systems...fsck 1.41.3 (12-Oct-2008)
```

```
done.
```

```
Setting kernel variables (/etc/sysctl.conf)...done.
```

```
Mounting local filesystems...done.
```

```
Activating swapfile swap...done.
```

```
Setting up networking....
```

```
Configuring network interfaces...done.
```

```
Setting console screen modes and fonts.
```

```
INIT: Entering runlevel: 2
```

```
Starting enhanced syslogd: rsyslogd.
```

```
Starting periodic command scheduler: crond.
```

```
Debian GNU/Linux 5.0 instance2 tty1
```

```
instance2 login:
```

Πλέον είμαστε έτοιμοι να κάνουμε login στο instance να το συνδέσουμε στο δίκτυο και να ελέγξουμε πως όντως αποκρίνεται.

Κεφάλαιο 4 – Υλοποίηση

4.1 Συμπεράσματα θεωρίας

Με βάση το λογισμικό και το υλικό που έχουμε καταλαβαίνουμε πως ακόμα και η εγκατάσταση στα συστήματα που υπάρχουν στο υπόγειο(Sun Enterprise 420R) είναι πρακτικά ανέφικτη αφού δεν υποστηρίζουν visualization. Εδώ πρέπει να σημειώσουμε πως από την στιγμή που όλο το backend κομμάτι του openstack έχει ανάγκη για visualization σαν εύκολα συμπεραίνονται τα παραπάνω. Κάνοντας μια γρήγορη ανασκόπηση των όσων υπάρχουν στο εργαστήριο θα έλεγα πως τα μηχανήματα είναι υπεραρκετά για να εγκατασταθεί ένα πολύ γρήγορο σύστημα πάνω στο οποίο θα τρέχει ένα storage network ικανό να αντεπεξέλθει στις ανάγκες που θα προκύψουν στο εργαστήριο από το συγκεκριμένο κομμάτι τουλάχιστον. Το βασικό πρόβλημα που έχουμε είναι πως δεν υπάρχουν τα μηχανήματα που θα εκμεταλλευτούν αυτό το storage network. Τουλάχιστον όχι για την εφαρμογή που τα θέλουμε αφού τα μηχανήματα της Sun μπορεί να είναι παλιά αλλά μπορούν εκεί πέρα να στηθούν κάποια services που θα μπορέσουν να αξιοποιήσουν οι σπουδαστές σε διάφορες δραστηριότητες. Μια πολύ καλή πρόταση ίσως είναι ένας web server στον οποίο θα τρέχουν διάφορες web υπηρεσίες που θα μπορούσαν να γίνουν σαν πτυχιακές σπουδαστών. Την υποδομή την έχουν αφού έχουν HBA .

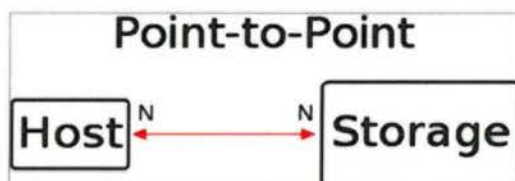
4.2 Προτεινόμενες τοπολογίες για τη storage area network

Για αρχή θα ήθελα να τονίσω την ανάγκη να τοποθετηθεί τουλάχιστον ένα switch gigabit στο υπόγειο αφού αυτή την στιγμή που μιλάμε, το δίκτυο που μοιράζεται με το συγκεκριμένο είναι 100 Mbps άρα οι χρήστες της υπηρεσίας θα μοιράζονται ένα ήδη πολύ αργό δίκτυο οπότε, ότι και να φτιάξουμε από εκεί και πέρα θα είναι ανούσιο, αφού θα παρατηρείται bottleneck στο δικτυακό κομμάτι. Προχωρώντας την μικρή αλλά και χρήσιμη παρένθεση οι προτάσεις για την τοπολογία που θα συνέφερε να χρησιμοποιήσουμε είναι δύο α) σε περίπτωση που απλά βρούμε ένα server και βάλουμε εκεί όλες τις υπηρεσίες που μας παρέχει το openstack τότε είναι περιττό να χρησιμοποιήσουμε το ed 140m, αφού πρακτικά δεν θα διαπράττει καμία λειτουργία. Βέβαια μία τέτοια λύση μπορεί να φαντάζει φθηνή και περισσότερο εύκολη στην υλοποίηση, έχει όμως πολλά αρνητικά. Πρώτα και κύρια είναι οι μειωμένες αποδόσεις σε ταχύτητα αλλά και ευπάθεια σε τυχόν σφάλματα. Το βασικότερο πλεονέκτημα σε cloud εικονικών μηχανών με την υποστήριξη san και cloud computing είναι πως αν πέσει κάποια εικονική μηχανή αυτόματα υπάρχει η δυνατότητα να σηκωθεί κάπου αλλού. Το είδαμε στο OpenStack εκεί που τα instance είχαν και δευτερεύοντα κόμβο τον οποίο χρησιμοποιούσαν αν ο πρωτεύων για κάποιο λόγο έπεφτε. Το κάνει πολύ καλά η ESX αφού το σύστημα αρχείων που έχει αναπτύξει είναι σχεδιασμένο ώστε, αν πέσει για τον άλφα ή βήτα λόγο μία εικονική μηχανή, τότε αυτομάτως ανεβαίνει σε άλλο μηχανήμα. Επιπλέον χάνουμε την δυνατότητα να δημιουργήσουμε ζώνες και θεωρητικά όλοι θα μπορούν να έχουν πρόσβαση σε όλα τα δεδομένα, ενώ πρακτικά την πιστοποίηση την αφήνουμε όλη σε επίπεδο εφαρμογών

κάτι που δεν προτείνεται για υπολογιστικά συστήματα σχεδιασμένα να δεχτούν αρκετό φόρτο . Για τους παραπάνω λόγους είναι εύκολα κατανοητό πως παρά την προσιτή τιμή η παραπάνω λύση δεν είναι η πιο ενδεδειγμένη αφού χάνουμε μερικά από τα πιο σημαντικά πλεονεκτήματα που προσφέρει το cloud computing(αν χρειαζόταν να παρομοιάσω το σύστημά μας με σύνδεση σημείου προς σημείο θα έλεγα πως είναι ίδιες διαφορές με την παράλληλη και ψευδοπαράλληλη επεξεργασία και μεταφορά). Δεύτερη και ποιοτικότερη σε ασφάλεια και ταχύτητα λύση είναι η δημιουργία μιας fabric τοπολογίας χρησιμοποιώντας σαν switch το ed-140m, ενώ παράλληλα μπορούμε να εκμεταλλευτούμε διάφορες ευκολίες που παρέχει, όπως ασφάλεια μέσα από τεχνικές που εμείς θα επιλέξουμε .Στην ουσία θα φτιάξουμε ένα cloud υπολογιστών οι οποίοι θα επωφελούνται τον αποθηκευτικό χώρο που θα τους παρέχει το storage network του οποίου τα βασικά στοιχεία θα είναι οι hosts (μηχανήματα που θα μπορούν να τρέξουν kvm),το ed-140m(enterprise directory-switch) και το cx700 (storage array).

Point-to-point topology

Η point-to-point τοπολογία συνδέει δύο συσκευές που στην περίπτωσή μας είναι ο server και το storage array. Όπως προαναφέραμε η συγκεκριμένη τοπολογία για δίκτυα αυτού του τύπου δεν είναι και η πιο αξιόπιστη . Έχει όμως αρκετά πλεονεκτήματα σε σχέση με το scsi πρωτόκολλο. Ένα βασικό είναι η απόσταση που μπορεί να καλύψει αφού το scsi έχει περιορισμούς ενώ οι οπτικές ίνες μπορούν να υπερβούν τα 10 χιλιόμετρα. Ένα δεύτερο και εξίσου σημαντικό πλεονέκτημα για τις αποδόσεις του δικτύου μας είναι η ανοχή στον ηλεκτρομαγνητικό θόρυβο αφού δεν μεταφέρεται ηλεκτρικό σήμα αλλά δέσμη φωτός.



Υλοποίηση της τοπολογίας : Όπως προανέφερα η υλοποίηση της συγκεκριμένης τοπολογίας είναι αρκετά πιο απλή σε σχέση με την δεύτερη επιλογή. Για να μπορέσουμε να υλοποιήσουμε την συγκεκριμένη τεχνική θα πρέπει να έχουμε έναν υπολογιστή – server με ένα host bus adapter εγκατεστημένο σε αυτόν. Από εκεί και πέρα για να ολοκληρωθεί η συνδεσμολογία του υλικού στο δίκτυο αποθήκευσης απλά πρέπει να συνδέσουμε την οπτική ίνα. Στην συνέχεια πρέπει να σχεδιάσουμε τα lun που θα χρησιμοποιήσουμε και όλα μαζί (lun, server) να μπουν σε ένα κοινό storage group ούτως ώστε να μπορούν να ανταλλάσσουν πακέτα.

LUNs: Η πρόταση που έχω με βάση αυτό και με βάση το πώς αργότερα ίσως μπορέσει το σύστημα να αναβαθμιστεί είναι να δημιουργήσουμε 3 LUNs και να παραμετροποιήσουμε το σύστημα να χρησιμοποιεί το καθένα για διαφορετικό σκοπό.

Δημιουργία Raid group

Unisphere

Πρώτα και κύρια από την στιγμή που θέλουμε να έχουμε την χρήση κάποιου Raid (raid 5 είναι η δική μου πρόταση) πρέπει να δημιουργήσουμε ένα raid group για να κάνουμε αυτό θα πρέπει.

1. Εισερχόμαστε στο σύστημα navisphere για να πλοηγηθούμε στο storage-system που θα δημιουργήσουμε τα raid group και τα LUN.
2. Πατάμε δεξί κλικ στο μενού raid group και έπειτα δημιουργία νέου raid group.
3. Διαλέγουμε το ID του raid group.
4. Έπειτα διαλέγουμε αυτόματη επιλογή δίσκων.
5. Διαλέγουμε τον αριθμό των δίσκων που θα χρησιμοποιήσουμε και πατάμε **Apply** για να τερματίσουμε την διαδικασία

navisphere cli

```
⑩ naviseccli -h 10.101.1.126 createrg 1 0_0_5 0_0_6 0_0_7 0_0_8 0_0_9 0_0_10 0_0_11 0_0_12 0_0_13 0_0_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 2 0_1_0 0_1_1 0_1_2 0_1_3 0_1_4 0_1_5 0_1_6 0_1_7 0_1_8 0_1_9 0_1_10 0_1_11 0_1_12 0_1_13 0_1_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 3 0_2_0 0_2_1 0_2_2 0_2_3 0_2_4 0_2_5 0_2_6 0_2_7 0_2_8 0_2_9 0_2_10 0_2_11 0_2_12 0_2_13 0_2_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 3 0_3_0 0_3_1 0_3_2 0_3_3 0_3_4 0_3_5 0_3_6 0_3_7 0_3_8 0_3_9 0_3_10 0_3_11 0_3_12 0_3_13 0_3_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 4 1_0_0 1_0_1 1_1_2 1_1_3 1_1_4 1_1_5 1_1_6 1_1_7 1_1_8 1_1_9 1_1_10 1_1_11 1_1_12 1_1_13 1_1_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 5 1_1_0 1_1_1 1_1_2 1_1_3 1_1_4 1_1_5 1_1_6 1_1_7 1_1_8 1_1_9 1_1_10 1_1_11 1_1_12 1_1_13 0_1_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 6 1_2_0 1_2_1 1_2_2 1_2_3 1_2_4 1_2_5 1_2_6 1_2_7 1_2_8 1_2_9 1_2_10 1_2_11 1_2_12 1_2_13 1_2_14 -pri high
```

```
⑩ naviseccli -h 10.101.1.126 createrg 7 1_3_0 1_3_1 1_3_2 1_3_3 1_3_4 1_3_5 1_3_6 1_3_7 1_3_8 1_3_9 1_3_10 1_3_11 1_3_12 1_3_13 1_3_14 -pri high
```

Δημιουργία LUN σε Raid group

Όπως γίνεται εύκολα αντιληπτό από τα παραπάνω δημιουργήσαμε 7 raid group τα οποία θα έχουν και τους δίσκους που περιγράφονται μέσα στην εντολή .

Στην συνέχεια θα πρέπει αυτά τα raid group να τα δεσμεύσουμε με κάποια lun .

Unisphere

1. Αφού πλοηγηθούμε στο storage-system που δημιουργήσαμε το raid group.
2. Έπειτα πατάμε δεξί κλικ και Bind LUN.
3. Διαλέγουμε raid type (το raid group που μόλις δημιουργήσαμε).
4. Δίνουμε τις παραμέτρους που είχαμε προαναφέρει στην θεωρία (Lun size, rebuild priority, enable read, write cache κ.α.)

navisphere cli

```
❶ navicli -h 10.101.1.16 bind r5 1 -rg 1 -rc 1 -wc 1 -sq mb -cap 3400
❷ navicli -h 10.101.1.16 bind r5 2 -rg 2 -rc 1 -wc 1 -sq mb -cap 3400
❸ navicli -h 10.101.1.16 bind r5 3 -rg 3 -rc 1 -wc 1 -sq mb -cap 3400
❹ navicli -h 10.101.1.16 bind r5 4 -rg 4 -rc 1 -wc 1 -sq mb -cap 3400
❺ navicli -h 10.101.1.16 bind r5 5 -rg 5 -rc 1 -wc 1 -sq mb -cap 3400
❻ navicli -h 10.101.1.16 bind r5 6 -rg 6 -rc 1 -wc 1 -sq mb -cap 3400
❼ navicli -h 10.101.1.16 bind r5 7 -rg 7 -rc 1 -wc 1 -sq mb -cap 3400
```

rc : read cash (1 = enable 0 = disable)

wc : write cash (1 = enable 0 = disable)

Storage group

Στην συνέχεια για να μπορέσουμε να διαμοιράσουμε τους δίσκους θα πρέπει να δημιουργήσουμε storage group όπου και θα προσθέσουμε και τους εικονικούς δίσκους αλλά και τον υπολογιστή που θα χρησιμοποιήσουν τους δίσκους .

UnisphereUnisphere

Δημιουργία storage group

Αφού πλοηγηθούμε στο storage-system που θέλουμε να δημιουργήσουμε το storage group πατάμε δεξί κλικ και δημιουργία storage group.

Μετά δίνουμε το όνομα του και πατάμε apply

Προσθήκη LUN στο storage group

Για να μπορέσουμε να συνδέσουμε τα αποθηκευτικά μέσα με τους host, που θα είναι συνδεδεμένοι στο σύστημά μας, πρέπει να τους έχουμε στο ίδιο storage group προς το παρόν και επειδή ακριβώς η τοπολογία μας είναι point-to-point θα βάλουμε και τα τρία lun σε ένα.

- Από το storage tab του enterprise storage διαλέγουμε το εικονίδιο του storage group που θέλουμε να προσθέσουμε τα Lun και πατάμε επιλογή Lun.

- Έπειτα διαλέγουμε τα τρία Lun που θέλουμε να προσθέσουμε.
- Πατάμε ok ή apply

Προσθήκη Host στο storage group

- Από το storage tab του enterprise storage διαλέγουμε το εικονίδιο του storage group που θέλουμε να προσθέσουμε τον Host και πατάμε επιλογή Connect Host.

- Επιλέγουμε τον Host ή τους Hosts και τους μεταφέρουμε στην λίστα με τους συνδεδεμένους Hosts.
- Έπειτα πατάμε ok.

navisphere cli

Δημιουργία storage group

```
navicli -h fc4700 storagegroup -create -gname storage_1
```

Προσθήκη LUN στο storage group

- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 1 -alu 1
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 2 -alu 2
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 3 -alu 3
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 4 -alu 4
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 5 -alu 5
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 6 -alu 6
```
- ⑩

```
navicli -h fc4700 storagegroup -addhlu -gname storage_1 -hlu 7 -alu 7
```

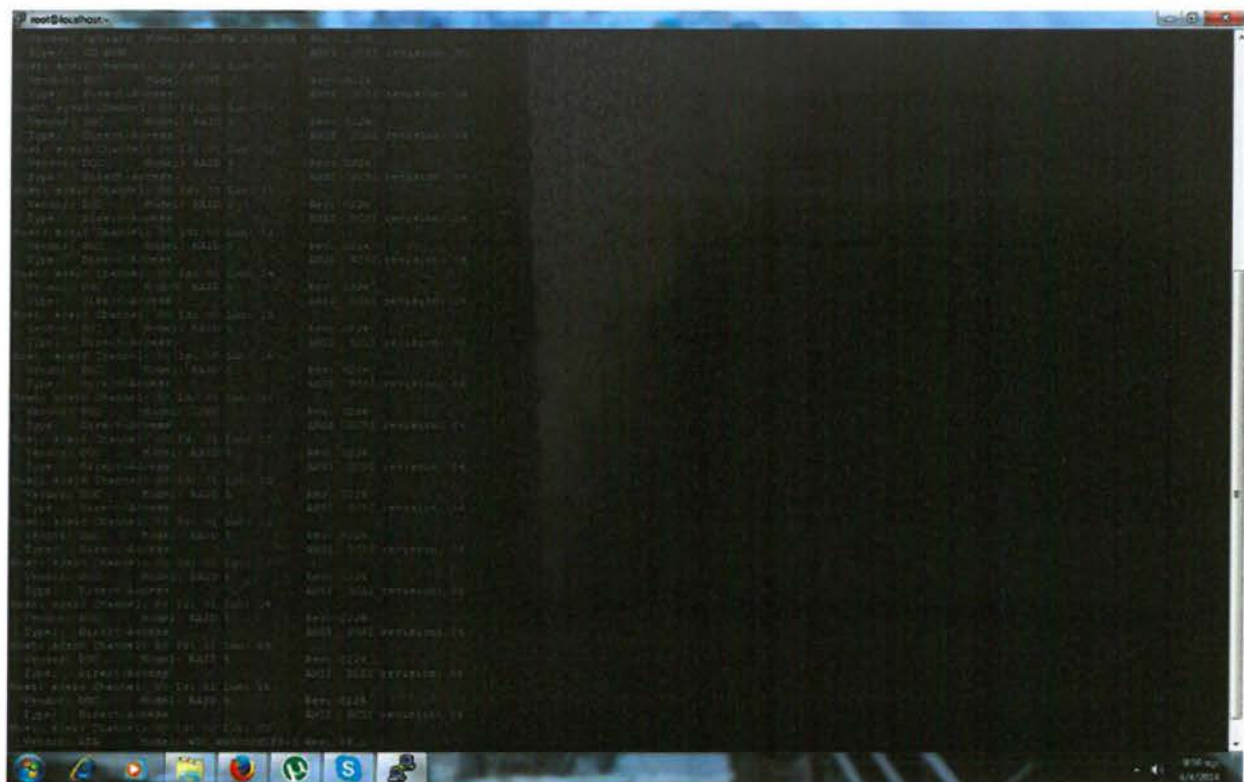
Προσθήκη Host στο storage group

- ⑩

```
navicli -h 10.101.1.16 HOST_HBA register
```
- ⑩

```
navicli -h fc4700 storagegroup -connecthost -host HOST_HBA -gname storage_1
```

Στην συνέχεια για να μπορέσουμε να έχουμε πρόσβαση στην υπολογιστή μας θα πρέπει να φορμάρουμε τα και να τους δώσουμε σύστημα αρχείων. Για να μπορέσουμε όμως να το κάνουμε αυτό θα πρέπει να χρησιμοποιήσουμε μηχανισμούς που έχει το Linux , για να βρούμε τα LUN και να τα επεξεργαστούμε .



A) από την χωρητικότητα των δίσκων χρησιμοποιώντας την εντολή fdisk -l

Βιβλιογραφία

[1] <http://sptecnologikaepitevgmata1.wikispaces.com>

[2] http://en.wikipedia.org/wiki/Optical_fiber_connector

[3] Basics and Application of Fibre Channel SAN, NAS, iSCSI, InfiniBand and FCoE, Second Edition Ulf Troppens, Wolfgang Muller-Friedt, Rainer Wolafka, IBM Storage Software Development, Mainz, Germany