ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Ανάπτυξη οδηγού στο Λ/Σ Linux για την υποστήριξη εικονικών δίσκων πάνω από κατανεμημένο σύστημα αποθήκευσης αντικειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Φίλιππος Γιαννάκος

Επιβλέπων: Νεκτάριος Κοζύρης
Αν. Καθηγητής ΕΜΠ

Αθήνα, Ιούνιος 2012

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Ανάπτυξη οδηγού στο Λ/Σ Linux για την υποστήριξη εικονικών δίσκων πάνω από κατανεμημένο σύστημα αποθήκευσης αντικειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Φίλιππος Γιαννάκος

Επιβλέπων: Νεκτάριος Κοζύρης
Αν. Καθηγητής ΕΜΠ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την -εισάγετε ημερομηνία-.

.....	.....	.....
Νεκτάριος Κοζύρης Αν. Καθηγητής ΕΜΠ	Νικόλαος Παπασπύρου Επ. καθηγητής ΕΜΠ	Παναγιώτης Τσανάκας Καθηγητής ΕΜΠ

Αθήνα, Ιούνιος 2012

.....
(Φίλιππος Γιαννάκος)
(Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών ΕΜΠ)

Ευχαριστίες

Η παρούσα διπλωματική εργασία εκπονήθηκε στο Εργαστήριο Υπολογιστικών Συστημάτων της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου υπό την επίβλεψη του Αναπληρωτή Καθηγητή Νεκτάριου Κοζύρη.

Θα ήθελα να ευχαριστήσω θερμά τον αν. καθηγητή κ. Κοζύρη για την ευκαιρία που μου έδωσε με την εκπόνηση της παρούσας διπλωματικής εργασίας να ασχοληθώ με το τομέα της επιστήμης των υπολογιστικών συστημάτων καθώς και για τη θετική συμβολή του καθόλη τη διάρκεια των σπουδών μου.

Ιδιαίτερες ευχαριστίες κατευθύνονται στον διδάκτορα Ευάγγελο Κούκη, και στον υποψήφιο διδάκτορα Γεώργιο Τσουκαλά, για την πολύτιμη συμβολή τους και τις συμβουλές τους κατά την εκπόνηση της διπλωματικής μου εργασίας.

Τέλος ένα μεγάλο ευχαριστώ στους γονείς μου και στην οικογένεια μου, για τη συνεχή τους στήριξη όλο αυτό το διάστημα των σπουδών μου, καθώς και στους φίλους και συμφοιτητές για τη στήριξη και τις ωραίες αναμνήσεις που μου προσέφεραν όλα αυτά τα χρόνια.

Περίληψη

Στη παρούσα διπλωματική εργασία μελετάται το πρόβλημα που παρουσιάζεται κατά τη δημιουργία εικονικών μηχανών, καθώς και κατά τη λήψη στιγμιοτύπων αυτών, στην υπηρεσία okeanos.

Κατά τη δημιουργία νέων εικονικών μηχανών, με βάση μια προρυθμισμένη εικόνα, απαιτείται αναπαραγωγή των δεδομένων στον εικονικό δίσκο της εικονικής μηχανής προκειμένου να έχει αποκλειστική χρήση. Δεδομένου ότι η εικόνα περιέχει τη βασική εγκατάσταση του λειτουργικού συστήματος και σε μεγάλο βαθμό δεδομένα που δε πρόκειται να αλλάξουν κατά τη χρήση του μηχανήματος, η φυσική αντιγραφή των δεδομένων θεωρείται σπατάλη τόσο αποθηκευτικού χώρου, όσο και χρόνου που απαιτείται για την αντιγραφή. Με παρόμοιο τρόπο υποφέρει και η λήψη στιγμιοτύπων μιας εικονικής μηχανής, όπου απαιτείται να παρθεί αντίγραφο του δίσκου τη χρονική στιγμή λήψης του στιγμιοτύπου.

Στη παρούσα διπλωματική εργασία παρουσιάζεται η προτεινόμενη λύση για την αντιμετώπιση του. Επιστρατεύτεται η χρήση της τεχνικής Copy-on-Write (CoW) για την άμεση δημιουργία αντιγράφου και τη φυσική αντιγραφή μόνο των δεδομένων που διαφοροποιούνται. Η λύση αυτή βασίζεται στη δημιουργία ανεξάρτητων υπομονάδων για τη διασύνδεση της εικονικής μηχανής με το δίσκο της, την τήρηση των χαρτών αντιστοίχησης των φυσικών δεδομένων και την πρόσβαση στα φυσικά δεδομένα. Η επικοινωνία των υπομονάδων επιτυγχάνεται μέσω μιας ειδικής βιβλιοθήκης επικοινωνίας, η οποία υποστηρίζει την επικοινωνία διαφορετικών διεργασιών, με τη χρήση ουρών και αιτήσεων δεδομένων.

Κύριο μέρος της διπλωματικής εργασίας αποτελεί η ανάπτυξη δυο υποσυστημάτων, του λογισμικού πρόσβασης στον αποθηκευτικό χώρο (sosd), ο οποίος βρίσκεται σε ένα κατανεμημένο σύστημα αποθήκευσης αντικειμένων και του οδηγού συσκευής block για το πυρήνα του Λ/Σ Linux (xsegbd), ο οποίος δρα ως συνδετικός κρίκος της εικονικής μηχανής με το δίσκο της. Στη συνέχεια ακολουθεί η αξιολόγηση τους. Αναπτύσεται η μεθοδολογία και το λογισμικό αξιολόγησης και παρουσιάζονται και ερμηνέυονται τα αποτελέσματα του, τόσο για το sosd, όσο και για τον οδηγό συσκευής.

Λέξεις κλειδιά

Σύννεφο, Υποδομή ως υπηρεσία, Εικονοποίηση, Κατανεμημένα συστήματα αποθήκευσης, Συστήματα αποθήκευσης βασισμένα σε αντικείμενα, Αντιγραφή κατά την

εγγραφή, Οδηγος συσκευής block, Linux

Abstract

The main objective of this diploma thesis is to study the problem that occurs during VM provisioning and storing a snapshot in GRNET service okeanos.

During VM creation based on a preconfigured disk image, in order to ensure the exclusive usage of the data, replication of data from the disk image to the virtual disk is a necessity. Based on the fact that the image contains the basic installation of the VM's operating system, and in general data with low probability of changing through the life span of the VM, physical replication of the data can be considered a waste of both storage space and copy time. Taking a snapshot of a VM , where we need to have a copy of the virtual disk, suffers in the same manner.

In this diploma thesis, there is a proposal to face this problem. The usage of Copy-On-Write (CoW) technique is proposed to instantly create a copy of the data, where the actual physical data are to be copied only when there is no other option. This solution is based on independent modules to handle the communication of the VM with its virtual disk, to handle the necessary map keeping in order to track the data actually stored in the storage system, and to provide access to the storage system. These independent modules communicate with each other with the aid of a custom communication library which supports communication between processes, providing the necessary queues and requests.

The main part of this diploma thesis is based on the development of two modules, the software to access the storage space (sosd) targeting in a distributed object store as a storage space provider, and the block device driver for the kernel of the operating system GNU/Linux (xsegbd) which acts as a link in the data path between the VM and its virtual disk. Next step is the evaluation of the previously mentioned modules. The methodology of the evaluation is declared along with the matching benchmarking software which was developed. Finally the results of the aforementioned procedure are being presented with an explanation of the system behavior.

Keywords

Cloud, Infrastructure as a service, Virtualization, Distributed storage, Object based storage, Copy on write, Block device driver, Linux

Περιεχόμενα

Κατάλογος σχημάτων

Κατάλογος πινάκων

Κεφάλαιο 1

Εισαγωγή

Η τεχνολογία cloud αναδείχθηκε μέσα από την προσπάθεια της βιομηχανίας, με πρωτοπόρο την εταιρία Amazon, να αξιοποιήσει τις υποδομές στα data centers οι οποίες εκτός αιχμής ήταν σε μεγάλο ποσοστό αδρανείς. Οι υπηρεσίες της Amazon, Elastic Compute Cloud (EC2) και Simple Storage Service (S3) αποτελούν το πρότυπο του cloud computing. Οι υπηρεσίες αυτές χρησιμοποιούνται από προγραμματιστές εφαρμογών web ως εξής: Όταν ο φόρτος της εφαρμογής αυξάνεται, μέσω του EC2, παρέχονται κατ' αίτηση εικονικές μηχανές συγκεκριμένων προδιαγραφών προς εξυπηρέτηση του αυξημένου φόρτου. Η εφαρμογή, που τρέχει στους κόμβους αυτούς, χρησιμοποιεί το S3 για να αποκτά πρόσβαση στα δεδομένα της. Οι μηχανές αυτές είναι αναλώσιμες, και η κατάστασή τους δεν διατηρείται όταν κλείνουν. Η κατάσταση της εφαρμογής καταγράφεται στην υπηρεσία S3. Οι αναλώσιμες εικονικές μηχανές του EC2 δεν εξυπηρετούν ανάγκες για μόνιμους εξυπηρετητές σε ένα περιβάλλον cloud. Η νεότερη υπηρεσία της Amazon, Elastic Block Storage (EBS), δίνει τη δυνατότητα δημιουργίας μόνιμων δίσκων όπου μπορούν με τη σειρά τους να μονιμοποιηθούν και οι εικονικές μηχανές.

Η υπηρεσία ∼okeanos παρέχει εικονικές μηχανές, δίκτυα και αποθηκευτικό χώρο, μέσω webUI για εύκολη αλληλεπίδραση με τους χρήστης αλλά και rest API καταλληλότερο για προγραμματιστικό έλεγχο. Υποστηρίζει τη δυναμική δημιουργία εικονικών μηχανών και δικτύων, βάσει των αναγκών και απαιτήσεων του τελικού χρήστη. Πρόκειται για μόνιμες (persistent) εικονικές μηχανές, στις οποίες ο χρήστης έχει πλήρη έλεγχο.

Η φιλοσοφία ανάπτυξης του συστήματος okeanos, είναι η επαναχρησιμοποίηση ή η ανάπτυξη συγκεκριμένων εργαλείων που υλοποιούν μια εργασία, και ο συντονισμός των επιμέρους εργαλείων στην υλοποίηση ενός ενιαίου συστήματος και στην παροχή μιας υπηρεσίας.

Λόγω της κατανεμημένης φύσης των υποδομών, φυσικοί αποθηκευτικοί και υπολογιστικοί πόροι που πρέπει να λειτουργήσουν μαζί, βρίσκονται σε διαφορετικές μηχανές. Το πώς θα συνδεθεί μια εικονική μηχανή με το δίσκο της είναι ένα σημαντικό ζήτημα στην αιχμή της τεχνολογίας, και κεντρικός άξονας της εργασίας αυτής.

1.1 Αντικείμενο της διπλωματικής

Στον υπηρεσία okeanos, κάθε εικονική μηχανή είναι συνδεδεμένη με τουλάχιστον έναν εικονικό δίσκο για την λειτουργία της. Ο δίσκος αυτός περιέχει κατ' ελάχιστον την εγκατάσταση του λειτουργικού συστήματος καθώς και κάποιες βασικές ρυθμίσεις του συστήματος. Εφόσον πρόκειται για μόνιμες εικονικές μηχανές, ο εικονικός δίσκος είναι αποθηκευμένος στο σύστημα αποθήκευσης της υπηρεσίας έτσι ώστε να διατηρείται ανεξάρτητα από τη κατάσταση της μηχανής.

Κατά τη διαδικασία της δημιουργίας ενός νέου εικονικού μηχανήματος, θα πρέπει να δημιουργηθεί ένας νέος εικονικός δίσκος, αποκλειστικός για αυτό το μηχάνημα. Ο εικονικός δίσκος κατασκευάζεται από κάποια εικόνα δίσκου η οποία θα περιέχει την κατάσταση του μηχανήματος που ο χρήστης επιθυμεί. Η υπηρεσία παρέχει ορισμένες βασικές εικόνες δίσκων που περιέχουν εγκατεστημένα διάφορα λειτουργικά συστήματα για την εύκολη δημιουργία διαφόρων τύπων εικονικών μηχανών, αλλά θα δέχεται επίσης και εικόνες δίσκων που παρέχονται από το χρήστη. Κατά τη διαδικασία της δημιουργίας του εικονικού δίσκου από την εικόνα δίσκου, τα δεδομένα της εικόνας δίσκου θα πρέπει να αναπαραχθούν προκειμένου να δημιουργηθεί ο νέος εικονικός δίσκος ανεξάρτητος από την αρχική εικόνα.

Ακόμη μια από τις δυνατότητες που προσφέρει η υπηρεσία okeanos, είναι η υποστήριξη για τη δημιουργία στιγμιότυπων των εικονικών μηχανών. Αυτό λειτουργεί με την υποστήριξη λήψης στιγμιοτύπων του εικονικού δίσκου και την αποθήκευση της κατάστασης του (στιγμιότυπο), δηλαδή την αποθήκευση των δεδομένων της εικόνας του δίσκου τη χρονική στιγμή λήψης του.

Οι δύο παραπάνω λειτουργίες απαιτούν επιπλέον αποθηκευτικό χώρο για την αναπαραγωγή των ίδιων πληροφοριών (δεδομένα εικόνας δίσκου), αλλά και σημαντικό χρόνο για την αντιγραφή των δεδομένων. Το πρόβλημα που δημιουργείται είναι η αισθητή καθυστέρηση στη δημιουργία νέας εικονικής μηχανής, αλλά και η χρήση πόρων του συστήματος που δεν είναι απαραίτητο, κυρίως αποθηκευτικού χώρου. Το συγκεκριμένο πρόβλημα ενέπνευσε την παρούσα διπλωματική εργασία. Μια διαφορετική προσέγγιση είναι η δημιουργία κλώνων της εικόνας δίσκου, με τη χρήση σημασιολογίας CoW σε αυτούς. Η δημιουργία κλώνων είναι εξαιρετικά γρήγορη λειτουργία, ενώ δεν χρησιμοποιείται περιττός χώρος αποθήκευσης, καθώς εγγράφεται πραγματικά, μόνο ό,τι αλλάζει σε σχέση με τα αρχικά δεδομένα.

1.2 Οργάνωση του κειμένου

Στη συνέχεια παρουσιάζεται η αρχιτεκτονική της λύσης που επιλέχθηκε για την αντιμετώπιση του προβλήματος, η συνεισφορά της διπλωματικής εργασίας στη λύση αυτή και η αξιολόγηση της απόδοσής της.

Πιο συγκεκριμένα, στο κεφάλαιο 2 παρουσιάζεται το απαραίτητο θεωρητικό υπόβαθρο για την καλύτερη κατανόηση τόσο του προβλήματος, όσο και της λύση για την αντιμετώπιση του. Δίνεται έμφαση στον αποθηκευτικό χώρο των υπολογιστικών συστημάτων, τα μοντέλα πρόσβασης σε αυτόν και τις συνήθεις τεχνικές που επισπεύδονται για τη βελτίωση των επιδόσεων τους. Γίνεται λόγος επίσης για τα κατανεμημένα

συστήματα αποθήκευσης και τις απαιτήσεις που οφείλουν να υλοποιούν.

Στο κεφάλαιο 3 παρουσιάζεται αναλυτικότερα το πρόβλημα και γίνεται αναφορά σε πιθανές λύσεις που εξετάστηκαν, ενώ παρουσιάζεται αναλυτικά η αρχιτεκτονική της λύσης που επιλέχθηκε.

Στο κεφάλαιο 4 παρουσιάζεται η συνεισφορά της εργασίας στη λύση αυτή, με συγκεκριμένα εργαλεία λογισμικού, καθώς και η αξιολόγηση τους. Παρατίθεται η μεθοδολογία αξιολόγησης και τα αποτελέσματα που ελήφθησαν μαζί με την ερμηνεία τους.

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

2.1 Τύποι cloud computing

SaaS: Ένας βασικός τύπος υπηρεσιών cloud είναι το μοντέλο του λογισμικού σαν υπηρεσία (Software as a Service, Saas). Σε αυτό το μοντέλο, η εκτέλεση των εφαρμογών τοπικά, αντικαθίσταται από τη φιλοξενία και την εκτέλεση τους (ή με την παροχή εφαρμογών αντίστοιχων δυνατοτήτων) στο σύννεφο. Ο χρήστης αποκτά πρόσβαση στην υπηρεσία μέσω μιας διαδικτυακής πύλης χρησιμοποιώντας το φυλλομετρητή (browser) του, από οποιοδήποτε υπολογιστή οπουδήποτε στον κόσμο. Έτσι οι χρήστες απαλλάσονται από το βάρος της εγκατάστασης και συντήρησης του απαραίτητου λογισμικού για την εργασία τους. Παράδειγμα τέτοιων διαδεδομένων υπηρεσιών αποτελούν τα google docs της google.

PaaS: Προχωρώντας πιο κάτω στην ιεραρχία των μοντέλων χρήσης του cloud computing, βρίσκεται το μοντέλο της πλατφόμας σαν υπηρεσία (Platform as a Service, PaaS). Σε αυτό το μοντέλο, παρέχεται στο χρήστη των υπηρεσιών του συννέφου, μια πλατφόρμα λογισμικού με όλα τα απαραίτητα εργαλεία καθώς και τους απαραίτητους υπολογιστικούς πόρους (κύκλοι cpu, μνήμη, διασύνδεση με το δίκτυο) για να τρέξει τις εφαρμογές του. Παράδειγμα γνωστής τέτοια υπηρεσίας, αποτελεί το google app engine.

IaaS: Τέλος, στο βάθος της ιεραρχίας των μοντέλων χρήσης του cloud computing, βρίσκεται το μοντέλο της υποδομής σαν υπηρεσία (Infrastracture as a Service, IaaS). Σε αυτό το μοντέλο, η υποδομή του σύννεφου, παρέχει στους χρήστες του εικονοποιημένους πόρους (υπολογιστικούς, αποθηκευτικούς, επικοινωνίας) για την καθ' απαίτηση δημιουργία εικονικών μηχανών. Οι εικονικές μηχανές αυτές, έρχονται συνήθως με κάποιο λειτουργικό σύστημα προεγκατεστημένο και ίσως και προρυθμισμένο με βάση τις επιλογές του χρήστη. Ο χρήστης στη συνέχεια έχει πλήρη πρόσβαση στις εικονικές μηχανές για να τις χρησιμοποιήσει σύμφωνα με τις ανάγκες του. Παράδειγμα τέτοιων υπηρεσιών είναι όπως αναφέρθηκε νωρίτερα οι Amazon web services.

2.2 Μοντέλα πρόσβασης στον αποθηκευτικό χώρο

Η χρήση του αποθηκευτικού χώρου ενός υπολογιστικού συστήματος απαιτεί τη χρήση ενός μοντέλου πρόσβασης στο φυσικό αποθηκευτικό μέσο για την εγγραφή ή την ανάκτηση δεδομένων. Το μοντέλο αυτό καθορίζει το τρόπο και τη μορφή της επικοινωνίας του υπολογιστικού συστήματος με τη συσκευή που προσφέρει τον αποθηκευτικό χώρο

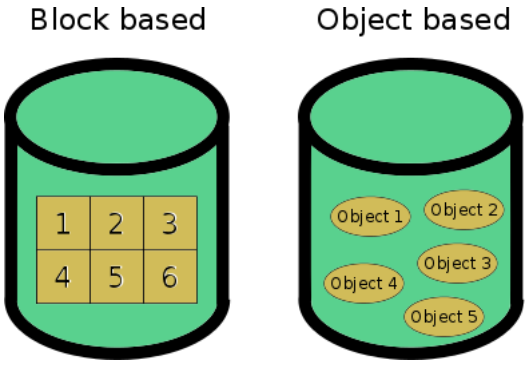
Το μοντέλο πρόσβασης στο φυσικό αποθηκευτικό μέσο, συνήθως κρύβεται από το χρήστη του αποθηκευτικού χώρου μέσω της χρήσης συστημάτων αρχείων ή με απευθείας πρόσβαση στον αποθηκευτικό χώρο των εφαρμογών που τον αξιοποιούν (π.χ. συστήματα διαχείρησης βάσεων δεδομένων).

2.2.1 Block storage

Ο παραδοσιακός τρόπος πρόσβασης σε συσκευές αποθήκευσης ακολουθεί το μοντέλο της αποθήκευσης σε blocks. Ένα block είναι μια σταθερού μεγέθους ακολουθία από bytes και αποτελεί τη βασική μονάδα πληροφορίας κατά την οποία μπορεί να γίνει η μεταφορά των δεδομένων με τη συσκευή. Έτσι η ροή των δεδομένων από και προς τη συσκευή που ακολουθεί αυτό το μοντέλο προσβασης χωρίζεται σε διακριτά κομμάτια πληροφορίας (blocks) σταθερού μεγέθους, τα οποία μεταβιβάζονται με ένα δίκτυο διασύνδεσης το οποίο εξαρτάται από τη συσκευή και τη φυσική της τοποθεσία. Οι συσκευές αυτές κρατούν τα δεδομένα οργανωμένα σαν ένα πίνακα από blocks στον οποίο σε κάθε είσοδο/έξοδο διαβάζεται ή εγγράφεται ένα ολόκληρο block. Παραδείγματα τέτοιων φυσικών συσκευών είναι οι σκληροί δίσκοι, οι οπτικοί δίσκοι και πληθώρα άλλων συσκευών αποθήκευσης. Οι συσκευές αυτές μπορούν να συνδέονται τοπικά πάνω στο ίδιο μηχάνημα στο οποίο προσφέρουν αποθηκευτικό χώρο μέσω κατάλληλων φυσικών διεπαφών (π.χ sas, sata) ή να συνδέονται μέσω ειδικών δικτύων αποθηκευτικού χώρου (storage area networks - SAN) δίνοντας το ίδιο αποθηκευτικό μοντέλο σε blocks μέσω ειδικών προτοκόλλων όπως είναι το iSCSI.

2.2.2 Object based storage

Μια διαφορετική προσέγγιση του μοντέλου πρόσβασης είναι η χρήση αποθηκευτικών συσκευών βασισμένες σε αντικείμενα (object based storage device OSD). Μια συσκευή αποθήκευσης αντικειμένων, είναι μια αποθηκευτική συσκευή, η οποία εν αντιθέσει με το παραδοσιακό προσανατολισμό πρόσβασης επιπέδου block, προσφέρει πρόσβαση σε δεδομένα που βρίσκονται σε διαφόρου μεγέθους αντικείμενα. Κάθε αντικείμενο έχει ένα αναγνωριστικό (Object identifier) με το οποίο είναι δυνατή η ανάκτησή του ή εγγραφή πληροφοριών σε αυτό. Τα αντικείμενα αυτά έχουν τόσο τα πραγματικά δεδομένα, όσο και μετα-δεδομένα, πληροφορίες δηλαδή για τις ιδιότητες του ίδιου του αντικειμένου.



Σχήμα 2.1: Block based storage vs Object based storage

2.3 Τεχνικές βελτίωσης επιδόσεων αποθηκευτικού χώρου

2.3.1 Caching and buffering

Μια από τις πιο διαδεδομένες τεχνικές αύξησης των επιδόσεων κατά την πρόσβαση σε δεδομένα, είναι η τεχνική του caching. Σύμφωνα με αυτή την τεχνική, τα δεδομένα στα οποία εκτελούμε εντολές εισόδου/εξόδου, αντιγράφονται σε μια δεύτερη τοποθεσία μνήμης στην οποία η πρόσβαση για εγγραφή ή ανάγνωση είναι πολύ πιο γρήγορη. Η χρησιμότητα και η επιτυχία της πηγάζει από το γεγονός ότι η πρόσβαση σε δεδομένα ακολουθεί στη πλειοψηφία των περιπτώσεων τις αρχές της τοπικής και της χρονικής τοπικότητας. Σύμφωνα με αυτές, δεδομένα που βρίσκονται κοντά τοπικά (π.χ. σε συνεχόμενες θέσεις μνήμης) με δεδομένα που έχουν χρησιμοποιηθεί πρόσφατα, έχουν αυξημένη πιθανότητα μελλοντικής πρόσβασης. Επίσης τα δεδομένα στα οποία υπήρξε πρόσβαση πρόσφατα, παρουσιάζουν μεγαλη πιθανότητα επανάληψης πρόσβασης. Έτσι κατά την πρώτη πρόσβαση στα δεδομένα, τα δεδομένα αντιγράφονται στη κρυφή μνήμη και κάθε επόμενη πρόσβαση των ίδιων δεδομένων γίνεται πολύ πιο γρήγορα από αυτή, ενώ η διαδικασία οφείλει να είναι διαφανής για το χρήστη. Η χρήση μνήμης cache βρίσκει ιδιαίτερη εφαρμογή στις αναγνώσεις, αλλά η ίδια τεχνική εφαρμόζεται και για τις εγγραφές, όπου η εγγραφή γίνεται πρώτα στην περιοχή της cache, και στη συνέχεια μεταφέρεται στην τοποθεσία όπου προοριζόταν, είτε άμεσα, είτε κατά την εκδίωξή της από την cache, είτε περιοδικά. Η τεχνική αυτή βρίσκει εφαρμογή σε μια πληθώρα από καθημερινές εφαρμογές, από την πρόσβαση στη μνήμη από τον επεξεργαστή, μέχρι τη χρήση κρυφών μνημών και μνημών προσωρινής αποθήκευσης από τις συσκευές φυσικής αποθήκευσης.

Μια επίσης διαδεδομένη τεχνική στο χώρο της επιστήμης των υπολογιστών, είναι η χρήση προσωρινής μνήμης κατά τη μεταφορά των δεδομένων. Τα δεδομένα κατά τη μεταφορά τους μεταξύ λογικών μονάδων ή διαφορετικών επιπέδων, αποθηκεύονται σε προσωρινές μνήμες, είτε λόγω αδυναμίας επεξεργασίας τους από το υλικό, είτε για λόγους βελτίωσης των επιδόσεων. Οι εντολές εισόδου εξόδου αρκετές φορές δημιουργούνται πολύ πιο γρήγορα από ότι μπορεί να εξυπηρετήσει το υλικό με

αποτέλεσμα να συσσωρεύονται σε περιοχές προσωρινής αποθήκευσης. Επίσης, λόγω φυσικών περιορισμών των συσκευών, η αναδιάταξη των αιτήσεων εισόδου εξόδου, μπορεί να επιφέρει σημαντική αύξηση των επιδόσεων προς τον τελικό χρήστη, κάτι που μπορεί να γίνει εύκολα με τη χρήση προσωρινών μνημών αποθήκευσης.

Στην ροή των δεδομένων κατά την αποθήκευση εφαρμόζεται caching και buffering τόσο σε επίπεδο λογισμικού από το λειτουργικό σύστημα, όσο και σε επίπεδο υλικού από τα φυσικά μέσα αποθήκευσης. Με τη φυσική πρόσβαση στους δίσκους να είναι σημαντικά αργή, όλα τα μοντέρνα λειτουργικά συστήματα εφαρμόζουν την τεχνική της χρήσης κρυφής μνήμης σελίδων (page cache) προκειμένου να επιταχύνουν τη πρόσβαση στα δεδομένα. Χρησιμοποιούν διαθέσιμες περιοχές τις μνήμης RAM του συστήματος για την αποθήκευση αντιγράφων των δεδομένων των φυσικών συσκευών, ενώ φροντίζουν να προωθούν τα δεδομένα που έχουν υποστεί αλλαγές προς τις φυσικές αποθηκευτικές συσκευές. Έτσι η page cache λειτουργεί και ως κρυφή μνήμη (cache) και ως προσωρινός χώρος αποθήκευσης των αλλαγών (buffer). Buffering επίσης υπάρχει κατά την μεταβίβαση των εντολών μεταξύ των υποσυστημάτων του λειτουργικού συστήματος για τη βελτίωση των επιδόσεων μέσω ειδικών αλγορίθμων, αλλά και κατά τη διάρκεια μεταβίβασης των εντολών προς τα φυσικά μέσα λόγω της διαφοράς στην ταχύτητα εξυπηρέτησης τους. Αντίστοιχα η χρήση cache και buffering έχει επεκταθεί και στις φυσικές συσκευές, με τη χρήση προσωρινής μνήμης αποθήκευσης, πριν την εγγραφή των δεδομένων στο μόνιμο αποθηκευτικό μέσο, επίσης λόγω της διαφοράς στη ταχύτητα πρόσβασης αλλά και για εφαρμογή έξυπνων αλγορίθμων αναδιάταξης αιτήσεων προς αύξηση των επιδόσεων.

Η χρήση μνήμης cache λόγω της σημαντικότερης επίδοσης στην πρόσβαση που παρουσιάζει, καταφέρνει να επιταχύνει την είσοδο έξοδο από και προς τις φυσικές συσκευές αποθήκευσης. Επειδή είναι όμως διαφανής ως προς το χρήστη, η εγγραφή δηλώνεται επιτυχής όταν τα δεδομένα εγγραφούν επιτυχώς στη cache, και όχι στο μόνιμο φυσικό αποθηκευτικό μέσο. Το παραπάνω δημιουργεί πρόβλημα στην ασφάλεια των δεδομένων, καθώς η μνήμη cache διακρατεί τα δεδομένα μόνο εφόσον βρίσκεται σε λειτουργία, οπότε τυχών αστοχία του συστήματος ή διακοπής της ρευματοδότησης, μπορεί να οδηγήσει σε απώλεια δεδομένων, για τα οποία ο χρήστης έχει διαβεβαιωθεί οτί βρίσκονται στη συσκευή μόνιμης αποθήκευσης. Για την αποφυγή του παραπάνω σεναρίου, ως συμβιβασμός ανάμεσα στην ασφάλεια και στις επιδόσεις, η χρήση των τεχνικών caching και buffering συνεχίζουν να υφίστανται, αλλά υιοθετήθηκε η χρήση της εντολής flush ως εργαλέιο για το χρήστη, ενώ μπορούν να δημιουργηθούν εντολές εγγραφής με σημασιολογία Force Unit Access (FUA). Η εντολή flush χρησιμοποιείται για να εγγραφούν τα περιεχόμενα μιας κρυφής μνήμης που δεν έχουν έχουν ακόμη εγγραφεί στον προορισμό τους, και επιστρέφει αφού ολοκληρωθεί η διαδικασία. Μια εγγραφή με σημασιολογία FUA, όπως προδίδει η ονομασία της, φροντίζει για την εγγραφή των δεδομένων της αίτησης στον τελικό προορισμό τους και επιστρέφει αφού ολοκληρωθεί η εγγραφή των δεδομένων.

2.3.2 Copy-on-Write

Η αντιγραφή κατά την εγγραφή (Copy on Write - CoW), αποτελεί συνηθισμένη πρακτική βελτιστοποίησης της απόδοσης ενός συστήματος, όπου η αντιγραφή δεδομένων είναι κοινό μοτίβο και συνεπώς απαιτείται να είναι εξαιρετικά γρήγορη. Κατά τη διάρκεια της αντιγραφής με χρήση αυτής της τεχνικής, δημιουργείται άμεσα ένα δεύτερο αντικείμενο σε λογικό επίπεδο, το αντίγραφο, χωρίς να έχουν αντιγραφεί πραγματικά σε φυσικό επίπεδο τα δεδομένα του. Το αντίγραφο συνεπώς μοιράζεται τα ίδια δεδομένα σε φυσικό επίπεδο με το αρχικό αντικείμενο. Κατά την εγγραφή σε ένα από τα δυο αρχεία, εφόσον τα δεδομένα στο σημείο της εγγραφής μοιράζονται, δημιουργείται ένα αντίγραφο ενός μέρους των αρχικών δεδομένων στο φυσικό χώρο αποθήκευσης, στο οποίο στη συνέχεια εγγράφονται οι αλλαγές. Το αντικείμενο στο οποίο έγιναν οι αλλαγές, αποκτά αποκλειστική χρήση στο φυσικό αντίγραφο των δεδομένων, ενώ ο αρχικός χώρος αποθήκευσης επιστρέφει στο αρχικό αντικείμενο, είτε με αποκλειστική χρήση, είτε μοιραζόμενη εαν το αντικείμενο έχει περισσότερα αντίγραφα με αυτή τη σημασιολογία. Τα δεδομένα που δεν έχουν υποστεί καμία αλλαγή, συνεχίζονται να μοιράζονται κανονικά (Σχήμα 2.2).

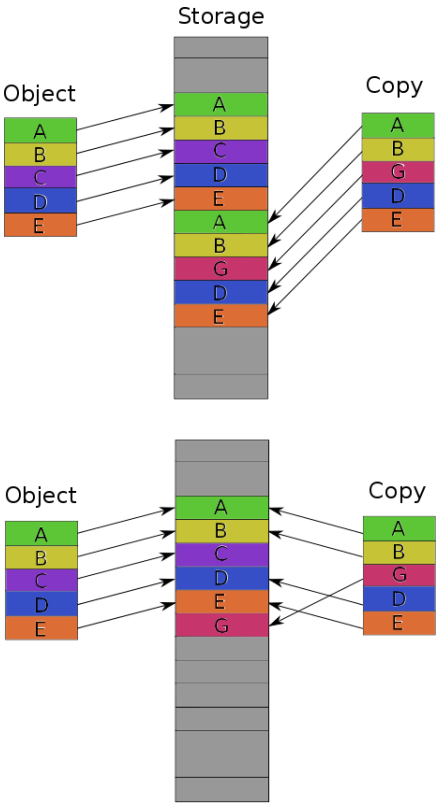
Για την υλοποίηση του CoW, τα αντικείμενα προς αντιγραφή, χωρίζονται νοητά σε σταθερού μεγέθους τεμάχια. Το μέγεθος αυτό αποτελεί την ελάχιστη πληροφορία που θα αντιγραφεί κατά την εγγραφή νέων δεδομένων, σε ένα από τα δύο αντικείμενα (CoW granularity). Συνεπώς όλα τα δεδομένα ενός τέτοιου τεμαχίου λειτουργούν ως μια μονάδα ως προς την ιδιοκτησία τους, δηλαδή είτε όλα μοιράζονται μεταξύ των αντικειμένων, είτε ανήκουν αποκλειστικά σε κάποιο από αυτά. Το αρχικό αντικείμενο αναφέρεται συχνά ως πατέρας, ενώ το αντίγραφο αντικείμενο ως παιδί. Για την εύρεση της τοποθεσίας των δεδομένων ενός αντικειμένου, είναι απαραίτητη η τήρηση χαρτών σχετικά με το ποια τεμάχια έχουν αντιγραφεί και ποια είναι κοινά.

Η παραπάνω τεχνική διανέμει το χρονικό κόστος της δημιουργίας του αντιγράφου στη διάρκεια εγγραφής των επιμέρους αλλαγών και ταυτόχρονα περιορίζει το αποθηκευτικό κόστος του αντιγράφου μόνο στις απαραίτητες αλλαγές. Προσθέτει όμως παραπάνω πολυπλοκότητα στην υλοποίηση, καθώς προκύπτει η ανάγκη τήρησης χαρτών με την τοποθεσία των δεδομένων και την αντιστοίχιση τους σε λογικά αντικείμενα.

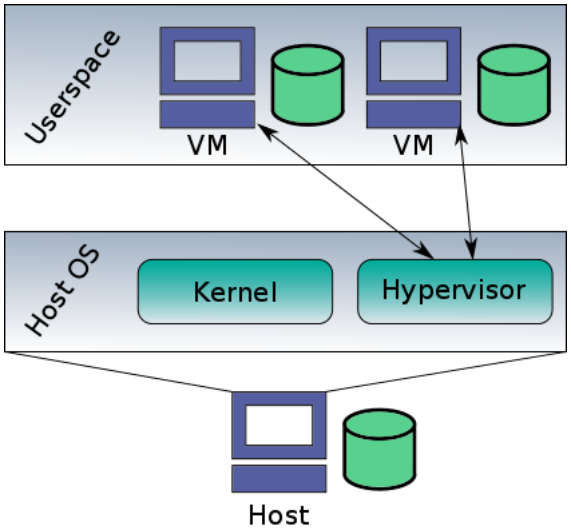
2.4 Virtualization

Ο όρος εικονική μηχανή έχει χρησιμοποιηθεί για να καλύψει διάφορους τομείς όπου απαιτείται η εκτέλεση μιας εφαρμογής σε ελεγχόμενο και απομονωμένο περιβάλλον.

Χρησιμοποιείται σε επίπεδο εφαρμογών χρήστη από ορισμένες γλώσσες προγραμματισμού, ως ένα ενδιάμεσο στάδιο της μεταγλώττισης του προγράμματος, όπου εκτελείται ο ενδιάμεσος κώδικας που παράγεται (bytecode) από την εικονική μηχανή της γλώσσας και το περιβάλλον που αυτή προσφέρει. Η εικονική μηχανή αναλαμβάνει τη μετάφραση του ενδιάμεσου κώδικα σε κώδικα μηχανής όπου εκτελείται η εφαρμογή, καθώς και την επικοινωνία με το περιβαλλον όπου εκτελείται, επιτρέπο-



Σχήμα 2.2: Σύγκριση της κανονικής αντιγραφής με τη τεχνική copy-on-write.



Σχήμα 2.3: Virtualization - type B hypervisor

ντας έτσι την απομονωμένη εκτέλεση των εφαρμογών και την εύκολη υποστήριξη πολλών πλατφόρμων εκτέλεσης (υλικού, λογισμικού).

Χρησιμοποιείται επίσης στη περιγραφή ενός εικονικού περιβάλλοντος υλικού, το οποίο συμπεριφέρεται όπως ένας πραγματικός υπολογιστης και επιτρέπει την εκτέλεση ενός λειτουργικού συστήματος. Το εικονικό περιβάλλον μπορεί να υλοποιηθεί με την εικονοποίηση του συστήματος της εικονικής μηχανής (virtualization) ή με τη προσομοίωση του (emulation).

Η προσομοίωση χρησιμοποιείται για την προσομοίωση συστημάτων διαφορετικής αρχιτεκτονικής από την αρχιτεκτονική του φυσικού συστήματος, και περιλαμβάνει σημαντική μείωση των επιδόσεων καθώς πρέπει να εξομοιωθεί ολόκληρη η αρχιτεκτονική. Αντίθετα στην εικονοποίηση, οι εφαρμογές που τρέχουν στην εικονική μηχανή, προορίζονται για την αρχιτεκτονική του συστήματος που φιλοξενεί την εικονική μηχανή, και μπορεί να τρέξει σχεδόν φυσικά πάνω στους φυσικούς πόρους. Η τεχνική αυτή ονομάζεται εικονοποίηση υλικού (hardware virtualization).

Η εικονοποίηση υλικού απαιτεί ένα ενδιάμεσο στρώμα λογισμικού με πλήρη πρόσβαση πάνω στους πόρους του υλικού. Το στρώμα αυτό ονομάζεται επόπτης (hypervisor) και δημιουργεί το εικονικό περιβάλλον λειτουργίας των εικονικών μηχανών. Αναλαμβάνει τη χρονοδρομολόγηση των εικονικών μηχανών, καθώς και να πολυπλέξει τις αιτήσεις τους για χρηση των φυσικών πόρων. Εγγυάται επίσης και την απομόνωση της εκτέλεσης των εικονικών μηχανών.

Υπάρχουν δύο τύποι hypervisor

α) εκείνοι που τρέχουν απευθείας πάνω στο υλικό του host

β) εκείνοι που τρέχουν πάνω ή μαζί με ένα λειτουργικό σύστημα.

Οι εικονικές μηχανές συνδέονται με έναν εικονικό δίσκο ο οποίος χρειάζεται αποθηκευτικό χώρο για την αποθήκευση των δεδομένων του. Ο χώρος αυτός ανατίθεται

στον εικονικό δίσκο από τον επόπτη ανάλογα με τις επιλογές του διαχειριστή και μπορεί να είναι μια αποθηκευτική συσκευή του συστήματος στην οποία έχει αποκλειστική χρήση ή μια εικόνα δίσκου η οποία βρίσκεται στο αποθηκευτικό σύστημα, χωρίς όμως να περιορίζεται μόνο σε αυτές τις επιλογές. Μια εικόνα δίσκου (disk image) είναι ένα αρχείο που περιέχει όλα τα δεδομένα ενός αποθηκευτικού μέσου με την ίδια δομή όπως στο αρχικό μέσο.

2.5 Αποθηκευτικός χώρος για το cloud

2.5.1 Βασικές αρχές

Η αποθηκευτική υποδομή θα πρέπει να υπακούει σε κάποιες βασικές αρχές και να έχει ορισμένα επιθυμητά χαρακτηριστικά για τη καλύτερη και ομαλότερη λειτουργία του συννέφου. Οι αρχές αυτές είναι γενικές, αλλά εδώ εξετάζονται από τη σκοπιά του συστήματος αποθήκευσης.

- **Αξιοπιστία (reliability):** Το σύστημα θα πρέπει να είναι αξιόπιστο και να προσφέρει επαρκή ασφάλεια στα δεδομένα που αποθηκεύει.

- **Robustness και ανοχή σε σφάλματα (Robustness and Fault tolerance):** Το σύστημα θα πρέπει να μπορεί να συνεχίζει να λειτουργεί σωστά, ακόμη και σε περιπτώσεις σφαλμάτων ή απρόσμενων συνθηκών. Συνδέεται άμεσα με την αξιοπιστία του συστήματος.

- **Υψηλή διαθεσιμότητα (High availability):** Η διαθεσιμότητα ενός συστήματος αναφέρεται στη δυνατότητα του χρήστη να αποκτήσει πρόσβαση σε αυτό. Ένα cloud σύστημα οφείλει να είναι υψηλής διαθεσιμότητας, δηλαδή διαθέσιμο ει δυνατόν συνέχεια, και την ίδια απαίτηση οφείλει να ικανοποιεί και το σύστημα αποθήκευσης στο οποίο στηρίζεται το σύννεφο. Βασική ιδιότητα ενός συστήματος κατασκευασμένο με βάση τη διαθεσιμότητα κατα νου, είναι η έλλειψη μοναδικού σημείου αποτυχίας single point of failure.

- **Κλιμάκωση (Scalability):** Με τον όρο κλιμάκωση εννοούμε την δυνατότητα αύξησης των διαθέσιμων αποθηκευτικών πόρων, με ανάλογη άυξηση της χωριτικότητας χωρίς η αύξηση αυτή να επηρεάσει την επίδοση του συστήματος αποθήκευσης (χρόνος πρόσβασης (latency), ρυθμός διαμεταγωγής (throughtput). Αποτελεί βασική ανάγκη ενός συστήματος αποθήκευσης για το σύνεφο, καθώς οι αποθηκευτικές ανάγκες αυξάνονται συνεχώς, και η αποθηκευτική υποδομή θα πρέπει να είναι σε θέση να υποστηρίζει ικανοποιητικά αυξανόμενο αποθηκευτικό χώρο.

- **Επιδόσεις (Performance):** Το αποθηκευτικό σύστημα θα πρέπει να προσφέρει τις υπηρεσίες του με ικανοποιητικές επιδόσεις, δηλαδή μικρή καθυστέρηση πρόσβασης στα δεδομένα, και μεγάλο ρυθμό διαμεταγωγής.

2.5.2 Τεχνικές

Τεχνικές που χρησιμοποιούνται για την επίτευξή τους:

- **Χρήση πλεονάζοντων πόρων:** Για τα σημεία εκείνα του συστήματος που θα πρέπει να είναι κοινά και απαραίτητα για τη λειτουργία του συστήματος, τοποθετούνται πλεονάζοντες πόροι (reduntant resources), για την αποφυγή downtime σε περίπτωση αστοχίας του κοινού σημείου, αλλά και για τον καταμερισμό του φόρτου εργασίας. Στα αποθηκευτικά συστήματα, βασικός πόρος όπου χρησιμοποιούνται πλεονάζοντες πόροι είναι τα ίδια τα δεδομένα. Με το πλεονασμό των δεδομένων, γίνεται αναπαραγωγή των δεδομένων σε περισσότερους του ενος κόμβους. Η χρήση πλεοναζόντων πόρων συνεισφέρει στην ασφάλεια και στη διαθεσιμότητα του συστήματος σε περίπτωση που ένας κόμβος δεν είναι διαθέσιμος λόγω κάποιου προβλήματος (διακοπή δικτύου, αστοχία υλικού κλπ), αλλά και στον καλύτερο επιμερισμό του φόρτου εργασίας.

- **Shared nothing architecture:** Πρόκειται για μια αρχιτεκτονική κατανεμημένων συστημάτων όπου κάθε κόμβος είναι ανεξάρτητος. Ο σχεδιασμός συστημάτων με βάση αυτή την αρχιτεκτονική βοηθάει στην αποφυγή μοναδικού σημείου αποτυχίας, καθώς και στην κλιμάκωση του συστήματος. Βρίσκει εφαρμογή στα συστήματα κατανεμημένης αποθήκευσης ως εξής: κάθε κόμβος έχει πρόσβαση και είναι υπεύθυνος μόνο για τα δεδομένα του χωρίς να υπάρχει κάποιου είδους αποθηκευτικός χώρος με κοινή πρόσβαση μεταξύ των κόμβων.

- **Failover:** Αυτόματη μετατόπιση φόρτου εργασίας και ομαλή συνέχιση εργασιών μεταξύ των κόμβων εξαιτίας αποτυχίας από κάποια αστοχία, σε πλεονάζοντες πόρους.

2.5.3 Κατανεμημένα συστήματα αποθήκευσης

Ο αποθηκευτικός χώρος για το σύννεφο, θα πρέπει να είναι προσβάσιμος από όλους τους κόμβους που υποστηρίζουν τη λειτουργία του σύννεφου. Θα πρέπει επίσης όλοι οι κόμβοι να βλέπουν τον ίδιο αποθηκευτικό χώρο, με κοινό χώρο ονομάτων για να απευθύνονται στα ίδια δεδομένα. Σε ένα κατανεμημένο σύστημα αποθήκευσης, τα δεδομένα βρίσκονται διασκορπισμένα σε πολλαπλά διαφορετικά φυσικά μηχανήματα και είναι προσβάσιμα από πολλαπλούς χρήστες οι οποίοι μπορεί να βρίσκονται σε διαφορετικά μηχανήματα. Το σύστημα αυτό φροντίζει για τη συνέπεια και την ασφάλεια των δεδομένων που αποθηκεύει και τελικώς παρέχει στους χρήστες του. Είναι επιθυμητό να προσφέρει καλές επιδόσεις καθώς και τα προαναφερθέντα χαρακτηριστικά.

Μια ειδίκευση του παραπάνω όρου, αποτελεί ένα κατανεμημένο σύστημα αποθήκευσης αντικειμένων (distributed object store). Πρόκειται για ένα cluster από συσκευές αποθήκευσης αντικειμένων (OSDs), οι οποίες συνεργάζονται και προσφέρουν έναν κατανεμημένο αποθηκευτικό χώρο. Ένα παράδειγμα κατανεμημένου συστήματος αποθήκευσης αντικειμένων που απασχολεί τη παρούσα διπλωματική εργασία είναι το RADOS.

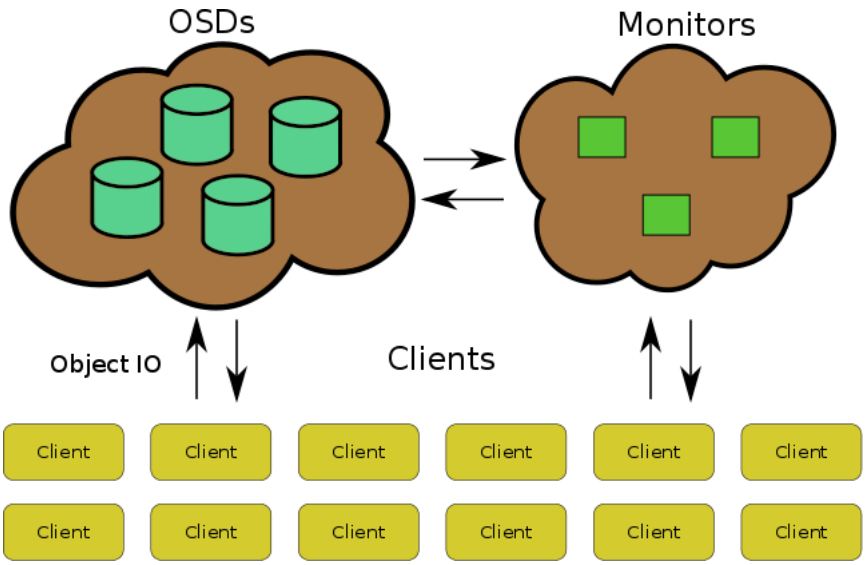
Ένα κατανεμημένο σύστημα αρχείων είναι ένα σύστημα αρχείων το οποίο προσφέρει την αποθήκευση/ανάκτηση δεδομένων σε επίπεδο αρχείων από ένα κατανεμημένο σύστημα αποθήκευσης. Το κατανεμημένο σύστημα αρχείων μπορεί να προσφέρει απευθείας πρόσβαση στο κατανεμημένο σύστημα αποθήκευσης σε επίπεδο block ή να αλληλεπιδρά μαζί του μέσω ενός πρωτοκόλλου πάνω από ένα δίκτυο. Στη πρώτη περίπτωση αντιστοιχούν τα συστήματα αρχείων κοινού δίσκου (shared disk file systems), ενώ στη δεύτερη αναφέρονται απλώς ως κατανεμημένα συστήματα αρχείων (distributed file systems).

Μια υποκατηγορία των κατανεμημένων συστημάτων αρχείων, είναι τα κατανεμημένα συστήματα αρχείων βασισμένα σε αντικείμενα (distribuited object based file systems), τα οποία έχουν σαν αποθηκευτική βάση ένα κατανεμημένο σύστημα αποθήκευσης αντικειμένων. Σε ένα τέτοιο σύστημα αρχείων, συνήθως τα μεταδεδομένα του αρχείου (πληροφορίες σχετικά με το μέγεθος τους, τα permissions τους κλπ) χωρίζονται από τα πραγματικά δεδομένα του αρχείου και διαχειρίζονται από ειδικούς εξυπηρετητές μετα-δεδομένων, ενώ τα πραγματικά δεδομένα αποθηκεύονται ως αντικείμενα σε ένα κατανεμημένο σύστημα αντικειμένων. Ένα παράδειγμα κατανεμημένου συστήματος αρχείων βασισμένο σε αντικείμενα που απασχολεί και τη παρούσα διπλωματική είναι το CEPH.

RADOS

Το RADOS αποτελεί ένα κατανεμημένο συστημα αποθήκευσης αντικειμένων, το οποίο προσπαθεί να προσθέσει επιπλέον νοημοσύνη στους κόμβους του cluster και να την εκμεταλευτεί προκειμένου να επιτύχει καλύτερη κατανομή υπολογιστικού και αποθηκευτικού φόρτου, αλλά και μεγαλύτερη αυτονομία μεταξύ των κόμβων. Το RADOS συνδυάζει αυτόματη αναπαραγωγή δεδομένων μεταξύ των κόμβων, ψευδοτυχαία κατανομή των δεδομένων στους κόμβους και αυτόματη διαδικασία failover σε περίπτωση αστοχίας ενός κόμβου. Επιπλέον κάθε κόμβος δρα σχεδόν αυτόνομα ως προς τα παραπάνω. Μέσω αυτών των δυνατοτήτων του προσφέρει αξιοπιστία (data replication), υψηλή διαθεσιμότητα (data replication, node redundancy, failover), κλιμάκωση (distribution of data placement and replication burden) και επίδοση (direct client IO access, pseudo randomly distribution of objects -> load balancing IOs between the nodes). [1]

Ένας RADOS cluster αποτελείται από τους OSD οι οποίοι αποθηκεύουν τα δεδομένα, και τους monitors οι οποίοι είναι υπεύθυνοι για το κύριο (master) αντίτυπο του χάρτη (cluster map) και συνεπώς μέσω αυτού για όλο τον cluster. Ό χάρτης αυτός αναπαριστά την τρέχουσα δομή του cluster, καθώς και πληροφορίες σχετικά με τη κατανομή των αντικειμένων. Κάθε μέλος του cluster αλλα και κάθε client έχει ένα αντίτυπο του χάρτη, ο οποίος ενημερώνεται από μικρές αυξητικές ενημερώσεις που ανταλλάσονται μεταξύ των κόμβων κατά την επικοινωνία τους όταν αυτή χρειάζεται (lazy map propagation). Κάθε έκδοση του χάρτη έχει ένα μοναδικό αριθμό έκδοσης (map epoch) ο οποίος αυξάνεται σε κάθε έκδοση, δηλαδή όταν αλλάξει η δομή του cluster (OSD up, down, in, out), και επιτρέπει στους κόμβους να έχουν μια συνεπή εικόνα του cluster, για την επικοινωνία τους. Υπεύθυνος για την έκδοση νέου χάρτη είναι οι monitors.



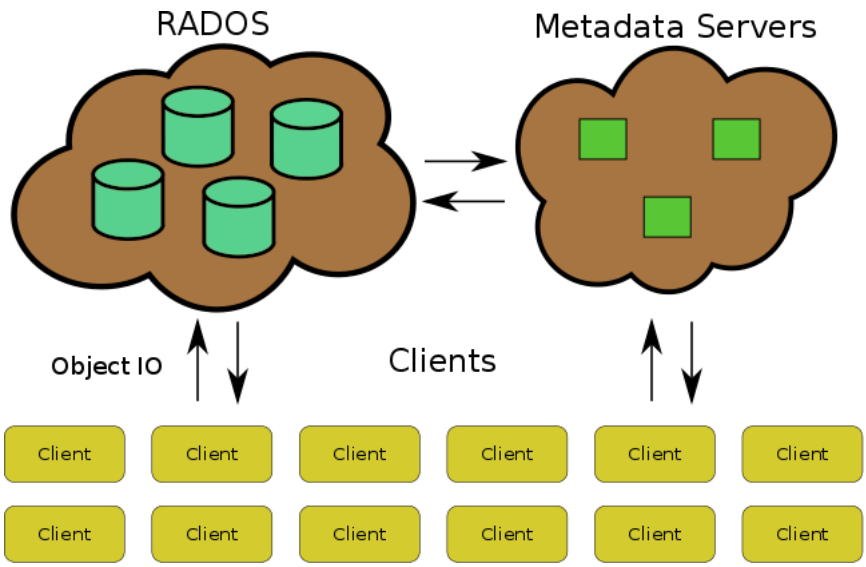
Σχήμα 2.4: Αρχιτεκτονική του RADOS

Κάθε αντικείμενο αντιστοιχίζεται σε ένα placement group. Το placement group είναι ένα λογικό σύνολο αντικειμένων τα οποία αναπαράγονται από τους ίδιους κόμβους. Η αντιστοίχιση γίνεται με βάση το όνομα του αντικειμένου και γίνεται με σταθερό υπολογιστικό τρόπο. Κάθε placement group αντιστοιχίζεται σε r osd nodes (όπου r το replication level) στους οποίους αναπαράγονται τα δεδομένα του placement group, Η αντιστοίχιση αυτή γίνεται μέσω του CRUSH, ενός αλγορίθμου κατανομής αντιγράφων, ο οποίος παράγει μια σταθερή ψευδο-τυχαία αντιστοίχιση. Η αντιστοίχιση αυτή γίνεται πάλι με γνωστό υπολογιστικό τρόπο από όλους τους κόμβους του συστήματος (OSDs, clients) και άρα η εύρεση της τοποθεσίας ενός αντικειμένου μπορει να γίνει υπολογιστικά, εξαλείφοντας έτσι την ανάγκη ύπαρξης κάποιου κεντρικού συστήματος data placement mapping. Η δυνατότητα αυτή, μεταφέρει το φόρτο της εύρεσης των δεδομένων στους κόμβους, απαλείφει την ανάγκη τήρησης μιας κεντρικής αντιστοίχησης για όλο το σύστημα, αφαιρώντας έτσι ένα congestion point και πιθανό bottleneck, δίνοντας έτσι περισσότερες δυνατότητες κλιμάκωσης του συστήματος.

Κάθε PG έχει έναν πρωτεύων κόμβο που είναι υπεύθυνος για αυτό. Ο πρωτεύων κόμβος επικοινωνεί με τους υπόλοιπους του Placement Group και φροντίζει για την αυτόματη αναπαραγωγή των δεδομένων μεταξύ των κόμβων σε κάθε εγγραφή. Επίσης σε περίπτωση κάποιας αστοχίας ενός πρωτεύοντος κόμβου, ένας πλεονάζων αναλαμβάνει το ρόλο του πρωτεύοντος και το σύστημα συνεχίζει την αδιάλειπτη παροχή υπηρεσιών χωρίς απώλεια δεδομένων.

CEPH

Το ceph είναι ενα κατανεμημένο σύστημα αποθήκευσης αρχείων σχεδιασμένο να παρέχει αξιοπιστία, καλή κλιμάκωση, διαθεσιμότητα, και όλα αυτά διατηρώντας παράλληλα της επιδόσης του. Βασίζεται στο κατανεμημένο σύστημα αποθήκευσης



Σχήμα 2.5: Αρχιτεκτονική του Ceph

αντικειμένων RADOS, το οποίο αποτελεί το σύστημα αποθήκευσης όλων των δεδομένων, αξιοποιώντας όλα τα πλεονεκτήματα και τις ιδιότητες που αυτό προσφέρει. [2]

Το ceph χωρίζει τα πραγματικά δεδομένα των αρχείων, από τα μεταδεδομένα τους. Τα πραγματικά δεδομένα αντιστοιχίζονται στατικά σε αντικείμενα τα οποία αποθηκεύονται στο RADOS. Οι χρήστες του συστήματος, στη συνέχεια επικοινωνούν κατευθείαν με αυτό για τη πρόσβαση στα δεδομένα των αρχείων. Για τα μεταδεδομένα, το ceph χρησιμοποιεί ειδικούς εξυπηρετητές μεταδεδομένων. Οι εξυπηρετητές αυτές διαφυλάσσουν τα μεταδεδομένα των αρχείων, και χρησιμοποιούν τον αποθηκευτικό χώρο του RADOS για την αποθήκευση και προστασία τους. Το σύστημα αρχείων Ceph υποστηρίζει επίσης τη δυνατότητα λήψης στιγμιοτύπων των αρχείων, τα οποία προορίζονται μόνο για ανάγνωση.

Κεφάλαιο 3

Αντικείμενο της διπλωματικής

Πριν την εξέταση των παρακάτω λύσεων είναι χρήσιμο να ξεκαθαριστούν ορισμένες έννοιες που θα χρησιμοποιηθούν.

- **Εικονικος δισκος (virtual disk):** Αποτελεί το δίσκο της εικονικής μηχανής.
- **Στιγμιότυπο (snapshot)**: Αποτελεί τη κατάσταση ενός εικονικού δίσκου μια συγκεκριμένη χρονική στιγμή και προορίζεται μόνο για ανάγνωση (readonly).
- **Κλώνος (clone)**: Με τον όρο κλώνο εννοούμε ένα αντίγραφο με CoW σημασιολογία το οποίο είναι εγγράψιμο (writable).

Για τον okeanos σημασιολογικά υπάρχουν δυο ειδών εικόνες δίσκων. Οι τόμοι (volume) και εικόνες (image).

- **Τόμοι:** πρόκειται για εγγράψιμες εικόνες δίσκων οι οποίες συνδέονται με τον εικονικό δίσκο μιας εικονικής μηχανής και αποτελούν τον αποθηκευτικό τους χώρο. Δημιουργούνται ως κλώνοι μιας εικόνας (βλέπε παρακάτω).
- **Εικόνες:** πρόκειται για εικόνες δίσκων που προορίζονται μόνο για ανάγνωση. Αποτελούν τη βάση για τη δημιουργία νέων τόμων για τις εικονικές μηχανές, καθώς και το τρόπο αποθήκευσης των στιγμιοτύπων που λαμβάνονται από υπάρχοντες τόμους.

3.1 Περιγραφή

Όπως αναφέρθηκε και στην εισαγωγή εφαλτήριο της παρούσας διπλωματικής αποτέλεσε το πρόβλημα της αναπαραγωγής των ίδιων δεδομένων και της καθυστέρησης που αυτή επιφέρει, κατά τη δημιουργία μιας νέας εικονικής μηχανής ή κατά τη λήψη ενός νέου στιγμιοτύπου.

Κατά τη δημιουργία μιας νέας εικονικής μηχανής, δημιουργείται ένας νέος εικονικός δίσκος δεμένος με έναν τόμο. Ο τόμος δημιουργείται από μια εικόνα, και για τη δημιουργία του, απαιτείται αντιγραφή των δεδομένων της εικόνας στο τόμο,

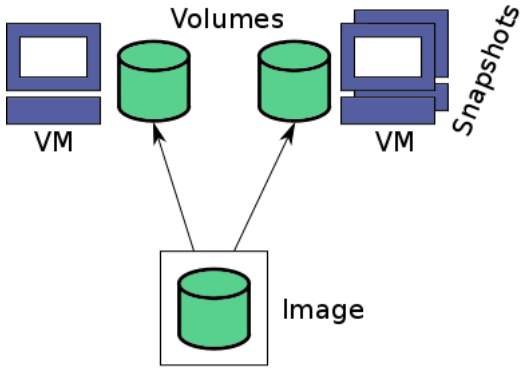
προκειμένου η εικονική μηχανή να έχει αποκλειστική πρόσβαση στα δεδομένα του τόμου τα οποία αρχικοποιούνται με εκείνα της εικόνας δίσκου. Η αντιγραφή αυτή των δεδομένων, απαιτεί αποθηκευτικό χώρο για την αναπαραγωγή των δεδομένων αλλά και χρόνο για τη πραγματοποίηση της. Συνεπώς η δημιουργία νέου τόμου, και κατ' επέκταση μιας νέας εικονικής μηχανής επιβαρύνεται χρονικά από την αντιγραφή των δεδομένων και επιβαρύνει το αποθηκευτικό σύστημα με την αναπαραγωγή των δεδομένων. Εφόσον όμως τα δεδομένα είναι ίδια, και εφόσον πρόκειται για τα δεδομένα της βασικής εγκατάστασης όπου θα παραμείνουν σε μεγάλο βαθμό ίδια, η αναπαραγωγή των δεδομένων είναι σπατάλη τόσο χώρου όσο και χρόνου.

Ακριβώς αντίστοιχη είναι και η περίπτωση της λήψης ενός στιγμιοτύπου. Κατά τη λήψη του στιγμιοτύπου, τα δεδομένα του τόμου τη χρονική στιγμή της λήψης θα πρέπει να αντιγραφούν σε μια νέα εικόνα δίσκου, η οποία αποθηκεύεται ανεξάρτητα από το τόμο. Εφόσον πρόκειται για αντιγραφή των δεδομένων μιας εικόνας δίσκου, απαιτείται τόσο επιπλέον αποθηκευτικός χώρος, όσο και χρόνος προκειμένου να ολοκληρωθεί. Όμως και σε αυτή τη περίπτωση τα δεδομένα είναι στην αρχή ίδια, ενώ οι αλλαγές που θα προκύψουν δε θα είναι σε ολόκληρο το τόμο. Άρα η φυσική αναπαραγωγή όλων των δεδομένων είναι περιττή.

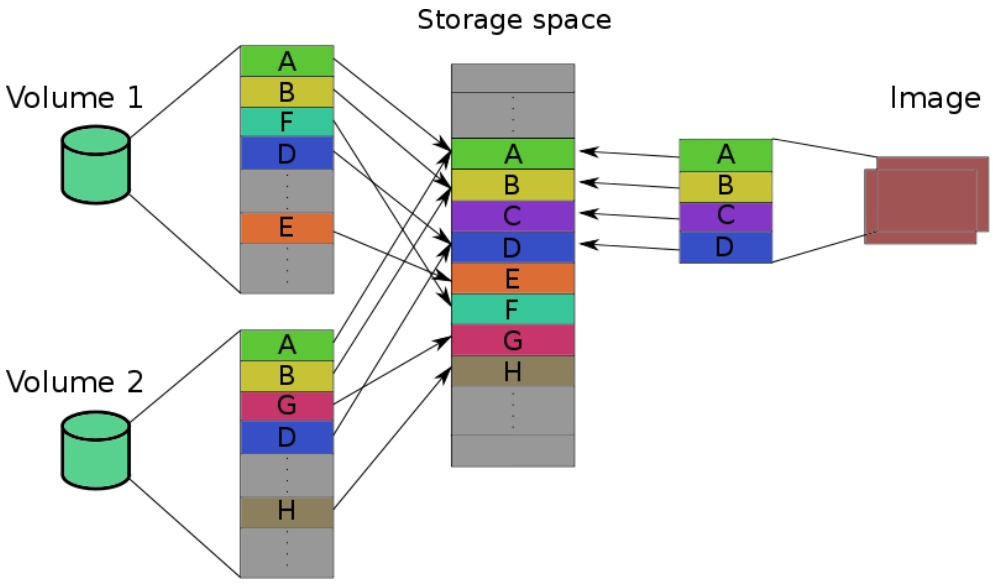
Η σπατάλη χώρου αναφέρεται στο γεγονός ότι αναπαράγονται τα ίδια δεδομένα και καταλαμβάνουν αποθηκευτικό χώρο, ενώ η καθυστέρηση αναφέρεται στο χρόνο που απαιτεί η αναπαραγωγή αυτή, ο οποίος αποτυπώνεται στο χρόνο εκτέλεσης της λειτουργίας. Η διαφορετική προσέγγιση στο πρόβλημα, που υιοθετεί η παρούσα διπλωματική, είναι η αντιγραφή των δεδομένων με τη χρήση CoW σημασιολογίας σε αυτά. Σύμφωνα με αυτή τη προσέγγιση, τα ίδια δεδομένα δε θα αναπαράγονταν στο φυσικό μέσο και πραγματικά παραπάνω αποθηκευτικό χώρο θα απαιτούν μόνο οι αλλαγές στο αντίγραφο (ή στο αρχικό, εξαρτάται ποιο αντιπροσωπεύει το τόμο, μιας και τα στιγμιότυπα προορίζονται μόνο για ανάγνωση), ενώ ο χρόνος για τη δημιουργία των αντιγράφων θα είναι σχεδόν μηδενικός, βελτιώνοντας κατά πολύ τις λειτουργίες της δημιουργίας νέων τόμων και της λήψης στιγμιοτύπων. Συνεπώς στόχος στον okeano είναι, ένας τόμος να αποτελεί κλωνο ενός στιγμιοτύπου, ενώ ένα στιγμιότυπο να είναι ένα read-only CoW αντίγραφο ενός τόμου, σχήμα που επιτρέπει μεγάλη ευελιξία για τους χρήστες της υπηρεσίας.

Ένα παράδειγμα για τα παραπάνω αποτελεί η δημιουργία 1000 εικονικών μηχανών από ίδια βασική εικόνα που περιέχει τη βασική εγκατάσταση του λειτουργικού, μεγέθους 5GB. Για την αντιγραφή των δεδομένων σε 1000 αντίγραφα απαιτείται χώρος περίπου 5TB. Όμως εφόσον πρόκειται για τη βασική εγκατάσταση του λειτουργικού, τα δεδομένα θα μείνουν σε μεγάλο βαθμό ίδια ανάμεσα στις διαφορετικές εγκαταστάσεις. Άρα με τη χρήση της προτεινόμενης τεχνικής, μπορούν τα 1000 εικονικά μηχανήματα να μοιράζονται τα δεδομένα της εγκατάστασης που έχουν κοινά και να αποθηκεύουν μόνο τις αλλαγές ή τα δεδομένα που είναι αποκλειστικά δικά τους. Επίσης για την δημιουργία 1000 αντιγράφων σε ένα αποθηκευτικό σύστημα με ρυθμό διαμεταγωγης 100MB/sec απαιτούνται κατ' ελάχιστον 14 ωρες! Αποφεύγοντας την αντιγραφή των δεδομένων με τη κοινή χρήση των φυσικών δεδομένων, ο χρόνος που απαιτείται για τη δημιουργία των εικονικών μηχανών μειώνεται δραματικά.

Η χρήση σημασιολογίας CoW για τα αντιγραφα των εικόνων δίσκων που δη-



Σχήμα 3.1: Οπτικοποίηση της σχέσης τόμων με τις εικόνες δίσκων στην υπηρεσία okeanos



Σχήμα 3.2: Παράδειγμα χρήσης της τεχνικής CoW για τη δημιουργία εικονικών μηχανών στην υπηρεσία okeanos

μιουργούνται, προϋποθέτει τη τήρηση χαρτών για να είναι δυνατή η εύρεση των δεδομένων. Η τήρηση αυτών των χαρτών σε ένα κατανεμημένο σύστημα όπου θα πρέπει να είναι προσβάσιμοι από όλους προϋποθέτει ορισμένες αρχές για να πραγματοποιηθεί σωστά. Καταρχάς θα πρέπει να είναι τοποθετημένοι σε μόνιμο αποθηκευτικό μέσο για να διασφαλίζεται η πληροφορία, προσβάσιμο παράλληλα από όλους. Επίσης, θα πρέπει να δίνονται εγγυήσεις οτί μια αλλαγή στο χάρτη θα εκτελείται ατομικά, δηλαδή ή θα επιτυγχάνει ή θα αποτυγχάνει και δε θα υπάρχει περίπτωση να αφήσει το σύστημα σε ημιτελή κατάσταση, π.χ. μετά από κάποια αστοχία του συστήματος. Επίσης θα πρέπει να προσφέρει απομόνωση μεταξύ 2 διαφορετικών αλλαγών στα ίδια δεδομένα, προκειμένου να εγγυηθεί τη μετάβαση των δεδομένων σε μια συνεπή κατάσταση.

3.2 Λύσεις που εξετάστηκαν

Στη συνέχεια παρουσιάζονται συνοπτικά οι εναλλακτικές λύσεις που εξετάστηκαν κατά την εκπόνηση της διπλωματικής εργασίας.

3.2.1 Προσθήκη υποστήριξης reflink στο CEPH

Σύμφωνα με το πρότυπο POSIX, τα αρχεία στο σύστημα οργανώνονται σε μια ιεραρχική δομή, όπου κάθε τερματικός κόμβος είναι ένα αρχείο (file), ενώ οι υπόλοιποι μη-τερματικοί κόμβοι είναι κατάλογοι (directories). Ένας κατάλογος είναι ένα αρχείο που περιέχει καταχωρήσεις καταλόγου (directory entries), αντικείμενα δηλαδή που συσχετίζουν ένα όνομα αρχείου με ένα αρχείο. Οι καταχωρήσεις αυτές ονομάζονται και αλλιώς δεσμοί (links). Σύμφωνα με το POSIX υπάρχουν δύο τύποι δεσμών που επιτρέπουν τη συσχέτιση διαφορετικών ονομάτων αρχείων με το ίδιο αρχείο.

- **Symlink:** Πρόκειται για ένα ειδικό τύπο αρχείου, που αποθηκεύει μια συμβολοσειρά που χρησιμοποιείται για να αλλάξει τη διαδικασία εύρεσης του αρχείου με βάση το όνομα μονοπατιού. Σε κάθε πρόσβαση σε ένα αρχείο τέτοιου τυπου, η διαδικασία εύρεσης του αρχείου με βάση το μονοπάτι, ανακατευθείνεται στο μονοπάτι που δείχνει η συμβολοσειρά που αποθηκεύει το αρχείο. Είναι ένα λογικός δείκτης που ανακατευθείνει τη πρόσβαση σε ένα αρχείο σύμφωνα με ένα μονοπάτι στην ιεραρχία του συστήματος αρχείων.

- **Hardlink:** Hardlink είναι η σχέση μεταξύ δυο διαφορετικών καταχωρήσεων καταλόγου που δείχνουν στο ίδιο ακριβώς αρχείο. Αντίθετα με το symlink, ένα hardlink δεν δείχνει σε ένα άλλο μονοπάτι στο επίπεδο της ιεραρχίας των αρχείων, αλλά δείχνει σέ ένα υπάρχον αρχείο ή καλύτερα στην εσωτερική δομή του συστήματος αρχείων που αναπαριστά ένα αρχείο. Η εσωτερική δομή αυτή ονομάζεται συχνά inode και αποθηκεύει μετα-δεδομένα, δηλαδή τις πληροφορίες του συστήματος αρχείων σχετικά με το αρχείο. Έτσι τα hardlinks αποτελούν καταχωρήσεις καταλόγου που δείχνουν στα ίδια inodes, αντικατοπτρίζουν το ίδιο αρχείο και συνεπώς και τα ίδια δεδομενα.

Επιπροσθέτως ορισμένα συστήματα αποθήκευσης αρχείων υλοποιούν και το δεσμό **reflink**. Η ιδέα του δεσμού reflink κατεβαίνει ακόμη ένα επίπεδο στο λογικό μοντέλο των συστημάτων αρχείων όπου πολλαπλά inodes δείχνουν στα ίδια δεδομένα, ενώ επιστρατεύεται η χρήση CoW για τις αλλαγές που μπορεί να προκύψουν σε κάποια από τα αρχεία. Δημιουργούνται έτσι, ξεχωριστές καταχωρίσεις καταλόγου και ξεχωριστά inodes που κρατούν τα μεταδεδομένα των αρχείων, έχοντας σαν αποτέλεσμα τη δημιουργία δύο ξεχωριστών αρχείων που μοιράζονται τα κοινά τους φυσικά δεδομένα.

Μια προσέγγιση στο πρόβλημα που απασχολεί τη παρούσα διπλωματική, και μια από τις προσεγγίσεις που εξετάστηκαν ήταν η χρήση του συστήματος αρχείων Ceph, ως το αποθηκευτικό σύστημα στο οποίο θα βρίσκονταν οι εικόνες δίσκων και οι εικονικοί δίσκοι, με τη παράλληλη προσθήκη υποστήριξης για τη λειτουργία του reflink σε αυτό. Σύμφωνα με αυτή τη προσέγγιση, οι εικόνες οι οποίες θα αποτελούσαν τη βάση ενός τόμου μιας εικονικής μηχανής θα μπορούσαν να είναι αποθηκευμένες στο σύστημα αρχείων Ceph, και να δημιουργουνται οι τόμοι, ως reflink μιας εκ των εικόνων. Αντίστοιχα τα στιγμιότυπα θα μπορούσαν να λαμβάνονται ως reflink του τόμου από τον οποίο προέρχονται, χωρίς δικαιώμα εγγραφής σε αυτά.

Η παραπάνω προσέγγιση κρίθηκε οτί ήταν αρκετά δύσκολη και πολύπλοκη να υλοποιηθεί, αρκετά έξω από τα πλαίσια της παρούσας διπλωματικής εργασίας. Επιπλέον δεν υπάρχουν εγγυήσεις οτί μπορεί να υλοποιηθεί η λειτουργία αυτή στο Ceph, αλλά ούτε τι επιδόσεις θα είχε ή πιθανές παρενέργειες. Τέλος αποτελεί μια ειδική λύση πάνω στο σύστημα αρχείων ceph, και όχι κάποια γενική επαναχρησιμοποιούμενη λύση, κάτι που είναι σχετικά μακρυά από τη φιλοσοφία ανάπτυξης του ωκεανού.

3.2.2 Προσθήκη υποστίριξης για κλώνους εικόνων στη librbd

Rados block device (rbd). Πρόκειται για μια συσκευή block βασισμένη πάνω στο RADOS που λειτουργεί τόσο για το πυρήνα του linux, όσο και για το λογισμικό εικονοποίησης QEMU. Η συσκευή αυτή προσφέρει πρόσβαση στα δεδομένα μιας εικόνας με τη χρήση των rbd images. Πρόκειται για εικόνες δίσκων που προσαρτώνται στη συσκευή rbd και βρίσκονται αποθηκευμένες στο RADOS. Η εικόνα χωρίζεται σε αντικείμενα σταθερού μεγέθους (4MiB) με προβλέψιμο όνομα, τα οποία στη συνέχεια αποθηκεύονται στο RADOS. Υπάρχει επίσης ένα αντικείμενο μοναδικό για τη κάθε rbd εικόνα όπου αποθηκεύονται τα μετα-δεδομένα της εικόνας.

Η librbd είναι μια βιβλιοθήκη που δίνει πρόσβαση στα rbd images από το χώρο χρήστη. Προσφέρει ένα ευρυ API για τη διαχείριση των εικόνων και την πραγματοποίηση IO σε αυτές. Σε αυτή βασίζεται και ο οδηγός για το QEMU. Με βάση τα παραπάνω, μια δεύτερη προσέγγιση στο πρόβλημα μας, ήταν προσθήκη υποστήριξης στη librbd και στο rbd για clonable images. Έτσι κατά τη δημιουργία ενός νέου vm, το βασικό image θα γινόταν απλά clone, με τα αντίστοιχα προτερήματα που προσφέρει αυτό. Σύμφωνα με τη μελέτη που έγινε πάνω σε αυτό, η ιδέα είχε ως εξής: Κάθε εικόνα κλώνος ανάλογα με το μέγεθος της και το μέγεθος κάθε αντικειμένου στα οποία χωριζόταν, είχε ένα συνολικό αριθμό αντικείμενων που την απάρτιζαν. Στα μεταδεδομένα της εικόνας αποθηκευόταν το όνομα της εικόνας που είχε χρησιμοποι-

ηθεί σα βάση αυτής της εικόνας, καθώς και ένας χάρτης bit (bitmap), όπου κάθε bit αντιπροσωπεύει τη πληροφορία για την ύπαρξη ή όχι ενος αντικειμένου της εικόνας. Τα αντικείμενα τα οποια υπήρχαν, είναι αυτά στα οποία έχει επέλθει αλλαγή σχετικά με την αρχική εικόνα και έχουν αντιγραφεί, ενώ τα υπόλοιπα είναι ίδια με αυτά της βασικής εικόνας και θα πρέπει να αναζητηθούν σε αυτή. Η λύση αυτή δεν αντιβαίνει τις παραπάνω αρχές καθώς τα δεδομένα αποθηκευόνται σε κοινό μόνιμο μέσο (το RADOS), δε μπορεί να αφήσει σε ημιτελή κατάσταση το χάρτη, καθώς κάθε αντικείμενο όταν γραφεται αλλάζει ένα μόνο bit που ή αλλαζει ή δεν αλλάζει. Επίσης με τη παραδοχή ότι κάθε εικόνα δίσκου χρησιμοποιείται κάθε στιγμή μόνο από ένα vm, δε χρειάζεται να ληφθεί μέριμνα με κλειδώματα για αλλαγές στα ίδια δεδομένα από διαφορετικούς χρήστες.

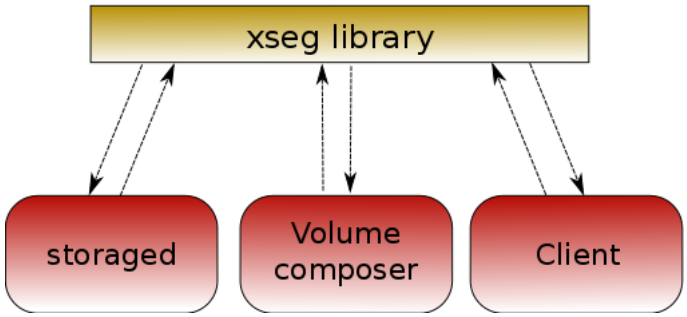
Η προσέγγιση αυτή ήταν ιδιαίτερα κοντά σε αυτό που θέλαμε να πετύχουμε, αλλά τελικά δε προτιμήθηκε διότι αποτελούσε εξειδικευμένη λύση πάνω στο RADOS και τη librb. Επίσης η υποδομή του rbd από πλευράς λογισμικού δεν ήταν ιδιαίτερα μεγάλη, οπότε μπορούσε να εγκαταλειφθεί η ιδέα της τροποποίησης και να αναπτυχθεί ένα νέο στρώμα λογισμικού με την επιθυμητή λειτουργικότητα που θα εξυπηρετούσε καλύτερα τις ανάγκες μας και δε θα ήταν δεσμευτικό από το RADOS.

3.3 Λύση που υιοθετήθηκε

Η λύση που τελικά υιοθετήθηκε, βασίζεται στη προσθήκη ενός ενδιάμεσου στρώματος λογισμικού, το οποίο θα αναλάβει τη διατήρηση των κλώνων και των στιγμιοτύπων των κλώνων σε λογικό επίπεδο, ενω ταυτόχρονα θα διατηρεί τους χάρτες με την αντιστοιχία δεδομένων. Η λύση αυτή λειτουργεί ως εξής: κάθε αίτηση προς τον τόμο ή την εικόνα δίσκου, θα έχει ως προορισμό το ενδιάμεσο αυτό στρώμα, το οποίο μέσω της πληροφορίας που έχει για τη πραγματική τοποθεσία των δεδομένων, θα αναλαμβάνει να ζητήσει τα κατάλληλα δεδομένα από το αποθηκευτικό σύστημα, εφόσον πρόκειται για ανάγνωση, ή να εγγράψει στη κατάλληλη τοποθεσία τα δεδομένα, ανανεώνοντας τις πληροφορίες για το τόμο, εφόσον πρόκειται για εγγραφή σε διαμοιραζόμενα δεδομένα σύμφωνα με τη τεχνική CoW.

Η αρχιτεκτονική της λύσης (σχήμα 3.3) βασίζεται στην ιδέα των διαφορετικών υπομονάδων που λειτουργούν ανεξάρτητα και επικοινωνούν μέσω ενός διαύλου επικοινωνίας. Οι υπομονάδες μπορεί να είναι α) ο volume composer - vlmc, το ενδιάμεσο στρώμα λογισμικού υπεύθυνο για διατήρηση των κλώνων και των στιγμιοτύπων β) υποσύστημα για τη πρόσβαση στον αποθηκευτικό χώρο που βρίσκονται τα δεδομένα (storaged) γ) υποσύστημα - χρήστης το οποίο ζητάει τα δεδομένα. Το τελευταίο θα δρα και ως το σημείο επικοινωνίας του συστήματος αυτού με την εικονική μηχανή.

Η επικοινωνία γίνεται μέσω μιας βιβλιοθήκης που επιτρέπει την επικοινωνία μεταξύ των διαφορετικών υποσυστημάτων, με τη χρήση ουρών, αιτήσεων και μνημών προσωρινής αποθήκευσης για τη μεταφορά των δεδομένων μεταξύ των υποσυστημάτων. Η βιβλιοθήκη αυτή βασίζεται στη χρήση κοινής μνήμης μεταξύ των υποσυστημάτων και στη δυνατότητα ειδοποίησης μεταξύ τους. Διαθέτει οδηγους τόσο για την επικοινωνία μεταξύ υποσυστημάτων που τρέχουν στο χώρο χρήστη, όσο και για την επικοινωνία υποσυστηματων μεταξύ χώρου χρήστη και χώρου πυρήνα, ενώ ο σχεδια-

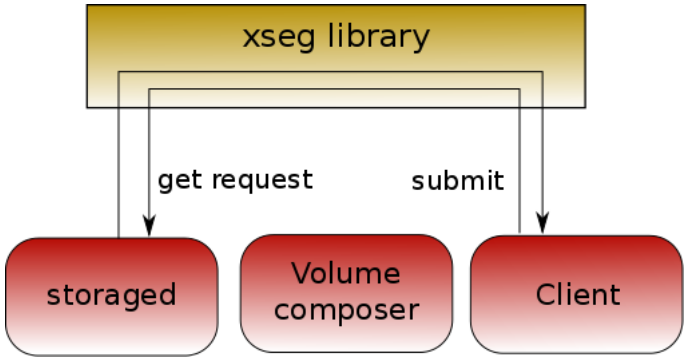


Σχήμα 3.3: Αρχιτεκτονική του συστήματος ως προς την βιβλιοθήκη xseg

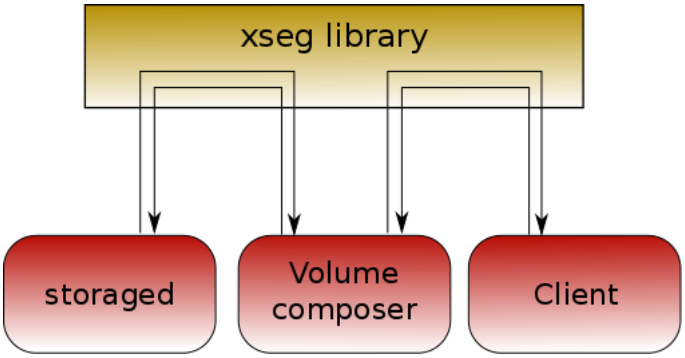
σμός του επιτρέπει την επικοινωνία μεταξύ δυο οποιονδήποτε υποσυστηματων όπου υποστηρίζεται η κοινή χρήση μνήμης και κάποιος τρόπος ειδοποίησης.

Το υποσύστημα - χρήστης ζητάει δεδομένα από έναν τόμο ή μια εικόνα δίσκου με το όνομα τους μέσω αιτήσεων της βιβλιοθήκης επικοινωνίας. Το υποσύστημα πρόσβασης στον αποθηκευτικό χώρο, είναι σχεδιασμένο να δέχεται αιτήσεις της βιβλιοθήκης επικοινωνίας για την εξυπηρέτηση συγκεκριμένων δεδομένων από ένα αντικείμενο ή αρχείο, το οποίο διαχωρίζεται με το όνομα του, ή για την εκτέλεση συγκεκριμένων λειτουργιών, ανάλογα με το τύπο της αίτησης. Ανάμεσα στην επικοινωνία των δυο υποσυστημάτων, μπορεί να παρεμβάλεται ο volume composer, ο οποίος ”συνθέτει” τόμους, και μεταβάλλει την αίτηση έτσι ώστε να αντικατοπτρίζει τα κατάλληλα δεδομένα.

Το αντικείμενο ή το αρχείο προς εξυπηρέτηση που φτάνει στο υποσύστημα πρόσβασης μπορεί να είναι μια ολόκληρη εικόνα δίσκου ή ένα κομμάτι της. Εαν δεν παρεμβάλλεται ο volume composer στην επικοινωνία με το υποσύστημα χρήστης, τότε το υποσύστημα πρόσβασης εξυπηρετεί δεδομένα κατευθείαν από την εικόνα δίσκου ή το τόμο (σχήμα 3.4). Αντίθετα εαν παρεμβάλλεται ο volume composer, τότε προκειμένου να υλοποιηθεί η CoW σημασιολογία, ο κάθε εικονικός δίσκος ή τόμος χωρίζεται σε τεμάχια σταθερού μεγέθους, και στο υποσύστημα του αποθηκευτικού χώρου, φτάνουν αιτήσεις που αφορούν δεδομένα από ένα τεμάχιο μιας εικόνας ή τόμου (σχήμα 3.5).



Σχήμα 3.4: Μονοπάτι χωρίς volume composer



Σχήμα 3.5: Μονοπάτι με volume composer

Κεφάλαιο 4

Συνεισφορά της εργασίας

Στο πλαίσιο της διπλωματικής εργασίας αναπτύχθηκαν τα λογισμικά sosd, για τη διασύνδεση RADOS και Volume και xsegbd, οδηγός για τη διασύνδεση Kernel και VM.

4.1 Λογισμικο sosd

Βασικό σημείο του μονοπατιού για τη χρήση των εικονικών μηχανών είναι η διασύνδεση των τόμων με το αποθηκευτικό σύστημα στο οποίο βρίσκονται. Λόγω των απαιτήσεων της υπηρεσίας και της φύσεως των υποδομών, είναι απαραίτητο οι τόμοι αυτοί να βρίσκονται σε ένα κατανεμημένο σύστημα αποθήκευσης. Σύμφωνα με την αρχιτεκτονική της λύσης που επιλέχθηκε, είναι απαραίτητη η πρόσβαση στο αποθηκευτικό σύστημα μέσω της υποδομής επικοινωνίας που προσφέρει η βιβλιοθήκη xseg. Ως λύση στο παραπάνω πρόβλημα, αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας λογισμικό για τη πρόσβαση στο κατανεμημένο σύστημα αποθήκευσης αντικειμένων RADOS. Το λογισμικό αυτό επικοινωνεί με το χρήστη του μέσω αιτήσεων που του παρέχονται από το χώρο επικοινωνίας που προσφέρει η xseg βιβλιοθήκη.

Στα πλαίσια υλοποίσης του λογισμικού, ορίστηκε ένα API, το Simple Object Store API, για την αποσύζευξη του κώδικα που υλοποιεί τη πρόσβαση στο κατανεμημένο σύστημα αποθήκευσης αντικειμένων, με τον κώδικα ειδικά για τη πρόσβαση στο RADOS.

Αποτελείται από έναν ορισμό για τις αιτήσεις που δέχεται, καθώς και ένα απλό σετ τριών συναρτήσεων (αρχικοποίηση, τερματισμός και υποβολή αιτημάτων) για την αλληλεπίδραση με τα αντικείμενα. Κάθε αίτηση προς τα αντικείμενα, συντάσσεται σύμφωνα με τη προκαθορισμένη μορφή της, και υποβάλλεται μέσω κλήσης της αντίστοιχης συνάρτησης (sos_submit). Ακολουθεί το μοντέλο της ασύγχρονης ειδοποίησης για την ολοκλήρωση μιας αίτησης, μέσω κλήσης μιας προκαθορισμένης συνάρτησης (callback function), η οποία δίνεται κατά την αρχικοποίηση του προγράμματος. Στη συνέχεια όπως είναι φυσικό, υλοποιήθηκε μια βιβλιοθήκη πρόσβασης στο RADOS μέσω κλήσεων του SOS API για τη χρήση της από το υπόλοιπο

```
struct sos_request {
        unsigned long id;               /* request id                   */
        char *name;                     /* target object name           */
        uint32_t namesize;              /* target object name length    */
        uint64_t offset;                /* target object offset         */
        uint64_t size;                  /* requested size of data       */
        char *data;                     /* data pointer                 */
        uint32_t flags;                 /* request flags                */
        volatile unsigned long state;   /* state of request             */
        int retval;                     /* operation's return value     */
        uint32_t op;                    /* operation to be performed    */
        void *priv;                     /* private data                 */
};

typedef int (*sos_cb_t)(struct sos_request *req, \
                        unsigned long event_flags);

struct sos_handle;
typedef struct sos_handle *sos_handle_t;

sos_handle_t sos_init(sos_cb_t cb);
void sos_shut(sos_handle_t sos);
int sos_submit(sos_handle_t sos, struct sos_request *req);
```

Code 1: SOS API

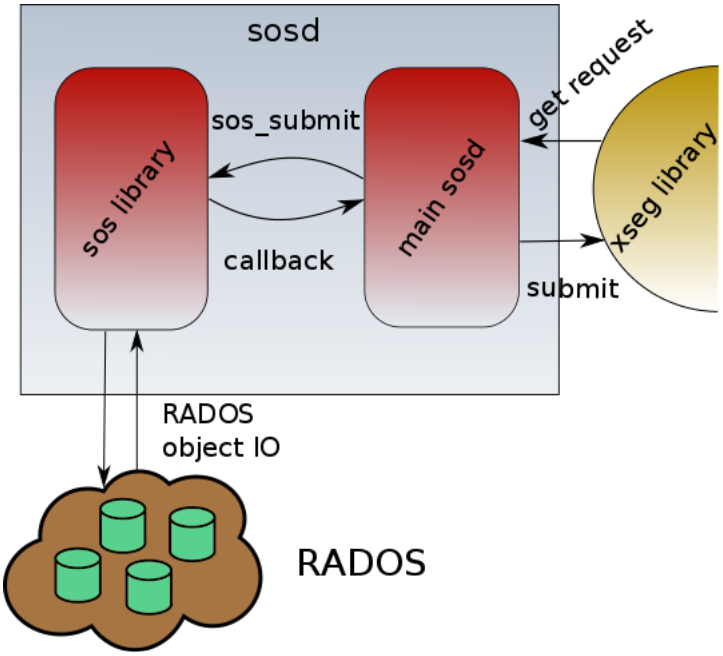
λογισμικό πρόσβασης στο κατανεμημένο σύστημα αποθήκευσης.

Το λογισμικό πρόσβασης που αναπτύχθηκε, λαμβάνει αιτήσεις της μορφής που ορίζονται από τη βιβλιοθήκη επικοινωνίας xseg και τις εξυπηρετεί, εκτελώντας εργασίες εισόδου/εξόδου από το κατανεμημένο σύστημα αποθήκευσης αντικειμένων μέσω της διεπαφής sos API (σχήμα 4.1).

4.2 xsegbd

Πέρα από τη σύνδεση του volume με το αποθηκευτικό χώρο, απαραίτητο βήμα αποτελεί και η σύνδεση της εικονικής μηχανής με το τόμο. Η σύνδεση μιας εικονικής μηχανής με τα δεδομένα του εικονικού της δίσκου πραγματοποιείται με τη βοήθεια το επόπτη, ο οποίος αναλαμβάνει να μετατρέψει τις αιτήσεις από το εικονοποιημένο Λ/Σ προς τον εικονικό δίσκο για block δεδομένων, σε αιτήσεις προς το πραγματικό αποθηκευτικό μέσο των δεδομένων. Η διαδικασία αυτή πραγματοποιείται συνήθως είτε με αιτήσεις προς το σύστημα αρχείων όπου βρίσκεται η εικόνα δίσκου αποθηκευμένη ως αρχείο, είτε με τη χρήση ενός οδηγού συσκευής block που δίνει πρόσβαση στον αποθηκευτικό χώρο. Ο οδηγός αυτός μπορεί να δίνει πρόσβαση στη πραγματική συσκευή των δεδομένων, π.χ σκληρό δίσκο - sda ή να εκθέτει μια νέα συσκευή στο λειτουργικό σύστημα η οποία να δινει πρόσβαση σε μια εικόνα δίσκου, αποθηκευμένη σε κάποιο άλλο αποθηκευτικό σύστημα π.χ. RADOS - rbd). Κοινό σημείο είναι η δημιουργία μιας συσκευής στο λειτουργικό σύστημα, η οποία δίνει πρόσβαση στα δεδομένα μέσω του block layer του linux.

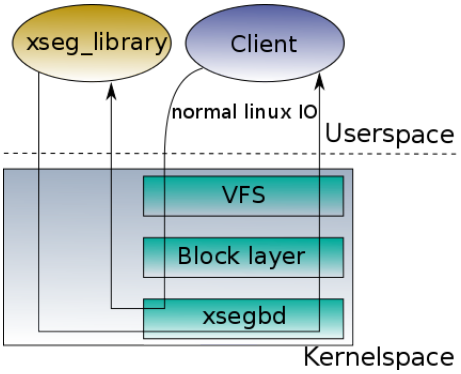
Για τη σύνδεση της εικονικής μηχανής με το τόμο της, αναπτύχθηκε στα πλαίσια



Σχήμα 4.1: Περιγραφή sosd

της παρούσας διπλωματικής εργασίας ένας οδηγός συσκευής για το λειτουργικό σύστημα linux, ο οποίος παρέχει πρόσβαση σε έναν τόμο, χρησιμοποιώντας τη βιβλιοθήκη επικοινωνίας xseg lib. Η βιβλιοθήκη επικοινωνίας αναλαμβάνει την επικοινωνία των δυο υποσυστημάτων, του πελάτη - block device και του λογισμικού πρόσβασης στο αποθηκευτικό σύστημα. Επειδή το λογισμικό πρόσβασης στο αποθηκευτικό σύστημα πρόκειται να τρέχει σε χώρο χρήστη, είναι απαραίτητη η επικοινωνία και η ανταλλαγή δεδομένων μεταξύ του χώρου πυρήνα από τη συσκευή block και του χώρου χρηστη. Τη λειτουργία αυτή την αναλαμβάνει η βιβλιοθήκη. Επιπλέον η συσκευή αυτή εξυπηρετεί και τη πρόσβαση σε έναν τόμο αποθηκευμένο σε ένα σύστημα αποθήκευσης για το οποίο υπάρχει πρόσβαση μέσω της βιβλιοθήκης xseg, όχι μόνο για την εικονική μηχανή που μπορεί να τη χρησιμοποιήσει, αλλά και για οποιονδήποτε άλλο πιθανό χρήστη.

Ο οδηγός συσκευής block που αναπτύχθηκε υλοποιεί τη σύμβαση επικοινωνίας του block layer του πυρήνα με τις συσκευές. Κάθε συσκευή block στο πυρήνα δέχεται αιτήσεις, τις οποίες εξυπηρετεί και ολοκληρώνει. Οι αιτήσεις αυτές συνήθως (αλλά όχι απαραίτητα) τοποθετούνται σε μια ουρά, από την οποία τις παίρνει η συσκευή. Για την υποστήριξη block συσκευών στον πυρήνα, υπάρχει μια ολόκληρη υποδομή block layer υπεύθυνη για τη διασύνδεση των συσκευών με το υπόλοιπο σύστημα. Η υποδομή αυτή προσθέτει επιπλέον λειτουργίες, κοινές για όλες τις συσκευές. Υποστηρίζει την ενοποίηση αιτήσεων που αναφέρονται σε διαδοχικά δεδομένα σε μια μεγαλύτερη, την αναδιάταξη της ουράς των αιτήσεων με στόχο την αποδοτικότερη εκτέλεση τους απο τη συσκευή κλπ. Επίσης κάθε block συσκεύη διαθέτει και δηλώνει ένα μέγεθος block, το οποίο αποτελεί την ελάχιστη μονάδα εισόδου εξόδου από τη



Σχήμα 4.2: Μονοπάτι δεδομένων με τον xsegbd

συσκευή. Μπορεί επίσης να δηλώσει επιπλέον πληροφορίες σχετικά με τις δυνατότητες της συσκευής προκειμένου οι βελτιστοποιήσεις της υποδομής block του πυρήνα να λειτουργήσουν πιο αποδοτικά. Η βασική πληροφορία που ενδιαφέρει τη παρούσα συσκευή είναι το μέγιστο μέγεθος μιας αίτησης που μπορεί να εξυπηρετήσει. Το μέγεθος αυτό θέτεται σε συνάρτηση με το μέγεθος μιας αίτησης στη βιβλιοθήκη xseg, καθώς αυτό αποτελεί το βασικό περιορισμό της συσκευής. Έτσι κάθε εντολή εισόδου εξόδου προς τη συσκευή ακολουθεί τη πεπατημένη διαδρομή της κλήσης συστήματος, linux vfs, block layer, block device driver. Το προηγούμενο μονοπάτι είναι το απλούστερο, χωρίς τη χρήση της page cache.

4.3 Αξιολόγηση

Για την αξιολόγηση των λογισμικών έγινε η εγκατάσταση του κατανεμημένου συστήματος αποθήκευσης αντικειμένων RADOS σε 4 φυσικά μηχανήματα. Κάθε φυσικό μηχάνημα λειτουργούσε ως ένας OSD ενώ παράλληλα 3 από αυτά λειτουργούσαν και ως RADOS monitor. Κάθε OSD είχε το journal του σε ξεχωριστό δίσκο όπου είχε πρόσβαση απευθείας στο block device και το χώρο των δεδομένων του σε ξεχωριστό δίσκο με το σύστημα αρχείων ext4. Διέθεται επίσης 16 επεξεργαστικούς πυρήνες και 48GB RAM. Οι OSDs ήταν ρυθμισμένοι σε βαθμό αναπαραγωγής δεδομένων 2, δηλαδή διατηρούσαν 2 φυσικά αντίγραφα κάθε αντικειμένου σε 2 διαφορετικούς OSD ενώ υπήρχαν 2 ενεργά threads σε κάθε OSD για την εξυπηρέτηση των αιτήσεων. Η διασύνδεση ανάμεσα στα φυσικά μηχανήματα και στο μηχάνημα πελάτης γινόταν με δίκτυο Ethernet 1Gbps.

Το λογισμικό sosd που αναπτύχθηκε, οφείλει να εξυπηρετεί αιτήσεις δεδομένων από ένα offset σε ένα αντικείμενο που του δίνεται με το όνομά του. Εφόσον δεν παρεμβάλλεται ο volume composer στην αξιολόγηση, το όνομα που του παρέχεται είναι το όνομα του τόμου που εξυπηρετεί. Για να μη καταλήξει ολόκληρος ο τόμος σε ένα αντικείμενο, το οποίο αντιστοιχίζεται πάντα στον ίδιο OSD, ενσωματώθηκε στο sosd, η ελάχιστη λειτουργικότητα από τον volume composer για τη διάσπαση της εικόνας δίσκου ή του τόμου σε σταθερού μεγέθους αντικείμενα. Το μέγεθος επιλέχθηκε να

είναι 4MB. Έτσι το λογισμικό sosd λαμβάνει μια αίτηση για δεδομένα σε συγκεκριμένο offset σε ένα τόμο, και με βάση το offset της αίτησης δημιουργεί μια νέα αίτηση (ή και παραπάνω αν τα ζητούμενα δεδομένα επεκτείνονται και σε άλλο αντικείμενο) για ένα συγκεκριμένο αντικείμενο που περιέχει τα δεδομένα του τόμου ή της εικόνας στο ζητούμενο offset προς το κατανεμημένο σύστημα αποθήκευσης αντικείμενων. Με αυτό το τρόπο τα δεδομένα μιας εικονας ή ενός τόμου μοιράζονται σε περισσότερους του ενός OSDs και επιμερίζεται ο φόρτος εξυπηρέτησης των αιτήσεων.

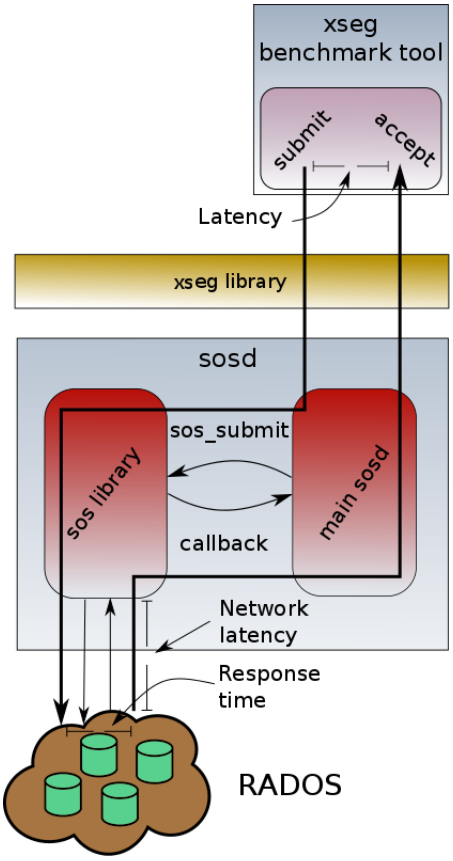
4.3.1 Αξιολόγηση sosd

Για την αξιολόγηση του sosd αναπτύχθηκε ειδικό λογισμικό μέτρησεων, το οποίο επικοινωνεί με τον sosd μέσω της βιβλιοθήκης xseg στέλνοντας έναν ορισμένο αριθμό αιτήσεων σύμφωνα με τις ρυθμίσεις της μέτρησης. Διαθέτει επιλογές για τον ορισμό δυο βασικών παραμέτρων της διαδικασίας μέτρησης, του μεγέθους της αίτησης και των παράλληλων αιτήσεων που θα είναι ενεργές. Το λογισμικό λαμβάνει τις μετρήσεις και παρουσιάζει το μέσο χρόνο εξυπηρέτησης μιας αίτησης κατά τη διάρκεια του ελέγχου, καθώς και του συνολικού ρυθμού διαμεταγωγής (throughput) που επιτεύχθηκε. Ο μέσος χρόνο καθυστέρησης (latency) μιας αίτησης ορίζεται ως το χρόνο από τη στιγμή που εκδίδεται εως ώτου ληφθεί επιτυχώς η απάντηση. Ο χρόνος αυτός περιέχει τόσο το χρόνο μεταφοράς των δεδομένων στον OSD όσο και το χρόνο εξυπηρέτησης της από το αποθηκευτικό σύστημα. Ο ρυθμός διαμεταγωγής αντίστοιχα ορίζεται ως το συνολικό μέγεθος των δεδομένων όλων των αιτήσεων που εκδόθηκαν προς τη συνολική χρονική διάρκεια της δοκιμής, το χρόνο δηλαδή από την υποβολή της πρώτης αίτησης, μέχρι τη λήψη απάντησης για την τελευταία.

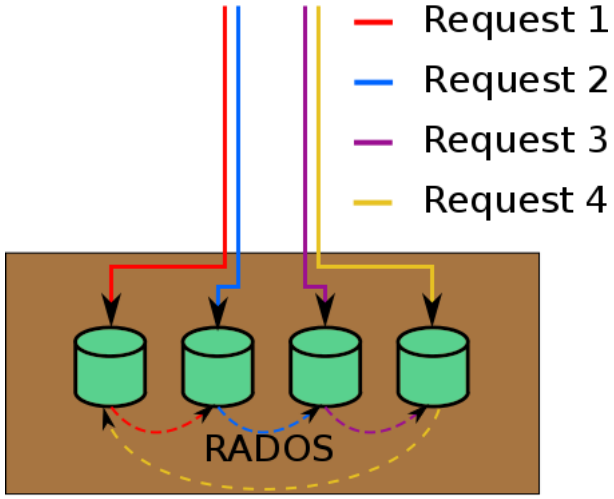
Το λογισμικό αξιολόγησης παράγει αιτήσεις εισόδου/εξόδου της βιβλιοθήκης xseg και φροντίζει να υπάρχουν όσες ενέργες παράλληλες αιτήσεις έχει ζητήσει ο χρήστης. Τα δεδομένα για κάθε αίτηση εγγραφής δημιουργούνται ψευδοτυχαία από το λογισμικό των μετρήσεων για τη μετέπειτα επιβεβαίωση των εγγραφών εφόσον αυτό ζητηθεί από το χρήστη, ή χρησιμοποιώντας έτοιμα δεδομένα για την αποφυγή του κόστους παραγωγής τυχαίων δεδομένων κατά τη λήψη των μετρήσεων. Οι αναγνώσεις αντίστοιχα χρησιμοποιούν τα δεδομένα που έχουν ήδη εγγραφεί στο αποθηκευτικό σύστημα κατά τη διάρκεια λήψης των μετρήσεων εγγραφών. Ο χρήστης μπορεί να οριοθετήσει τόσο τη μέγιστη διάρκεια της μέτρησης, όσο και το μέγιστο αριθμό διαφορετικών αιτήσεων που υποβάλλονται κατά τη διάρκεια της μέτρησης. Το μονοπάτι δεδομένων που ακολουθείται κατά την εκτέλεση των μετρήσεων φαίνεται στο σχήμα 4.3.

Το λογισμικό μετρήσεων υποστηρίζει επίσης δύο διαφορετικούς τύπους μοντέλων πρόσβασης (access paterns):

- **Κάθε αίτηση να γίνεται σε διαφορετικό τόμο (Σχήμα 4.4):** Κατά τη διάρκεια αυτού του τεστ, οι αιτήσεις καταλήγουν πάντα σε διαφορετικά αντικείμενα, εφόσον κάθε φορά πρόκειται για διαφορετικό τόμο. Το σενάριο αυτό πρόσβασης παρουσιάζεται κατά την πρόσβαση στο αποθηκευτικό σύστημα, από περισσότερες απο μία εικονικές μηχανές στο ίδιο φυσικό μηχάνημα. Κάθε εικονική μηχανή θα έχει το δικό της τόμο, και συνεπώς οι αίτησεις δε θα



Σχήμα 4.3: Μονοπάτι δεδομένων με τη χρήση του λογισμικού μετρήσεων

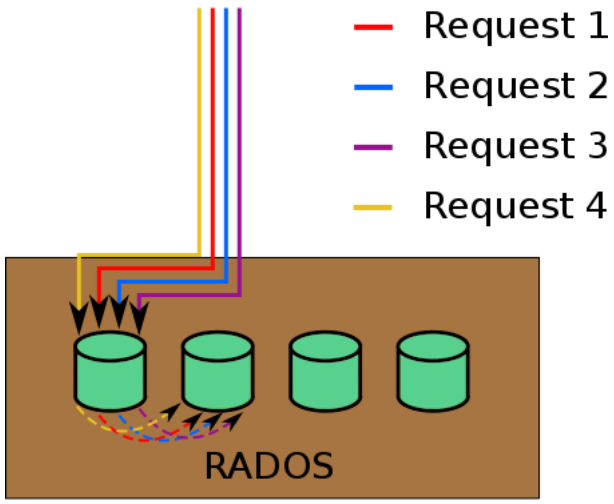


Σχήμα 4.4: Καταμερισμός αιτήσεων στο 1ο σενάριο αξιολόγησης

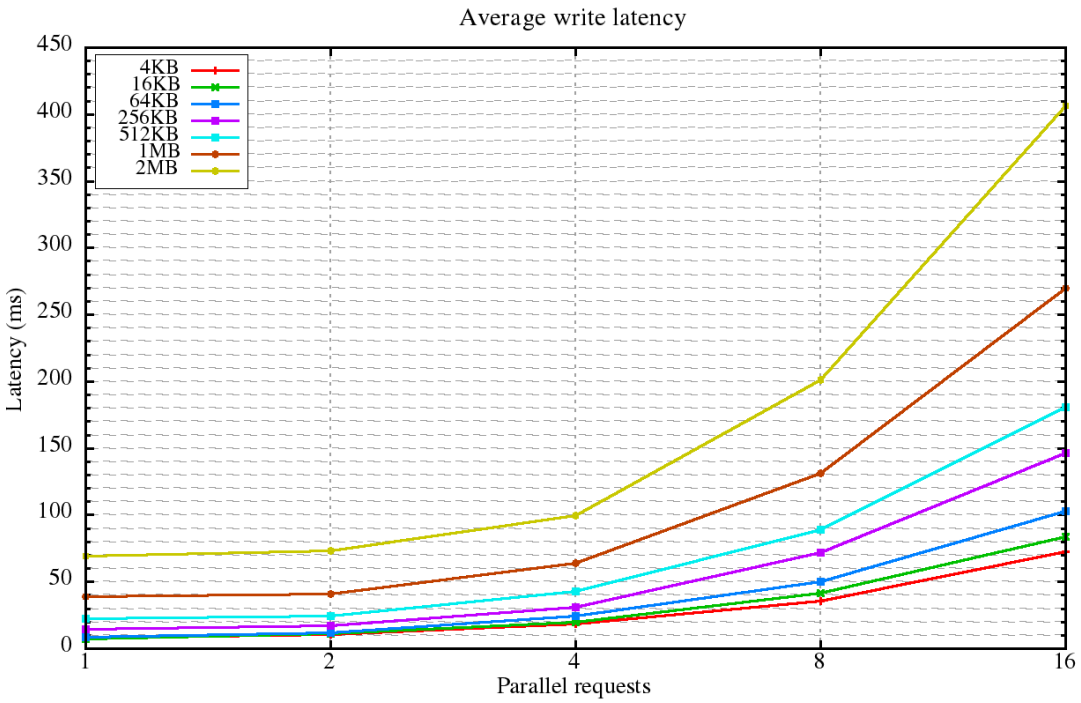
καταλήγουν στο ίδιο αντικείμενο. Αυτου του είδους η πρόσβαση μπορεί να παρουσιαστεί και από μια εικονική μηχανή με ένα τόμο, όπου γίνονται πολλές διαφορετικές αιτήσεις σε διαφορετικά offset του τόμου (πχ εγγραφή σε διαφορετικά αρχεία στο σύστημα αρχείων που υπάρχει στο τόμο, οι οποίες καταλήγουν σε διαφορετικά offset στο τόμο).

- **Κάθε αίτηση να αποτελεί συνέχεια της προηγούμενης στον ίδιο τόμο (Σχήμα 4.5):** Με αυτό το τρόπο, κάθε αίτηση καταλήγει στο ίδιο αντικείμενο με τις προηγούμενες τις, εως ώτου το offset στο τόμο ξεπεράσει το μέγεθος του τρέχοντος αντικειμένου (πχ για αιτήσεις μεγέθους 256KB και μέγεθος αντικειμένου 4MB, οι 16 συνεχόμενες αιτήσεις θα αναφέρονται στο ίδιο αντικείμενο στο κατανεμημένο σύστημα αποθήκευσης αντικειμένων). Το σενάριο αυτής της πρόσβασης βασίζεται στη συνεχόμενη πρόσβαση για εγγραφή ή για ανάγνωση δεδομένων στο τόμο (πχ εγγραφή ή ανάγνωση ενός μεγάλου αρχείου στο σύστημα αρχείων που υπάρχει στο τόμο).

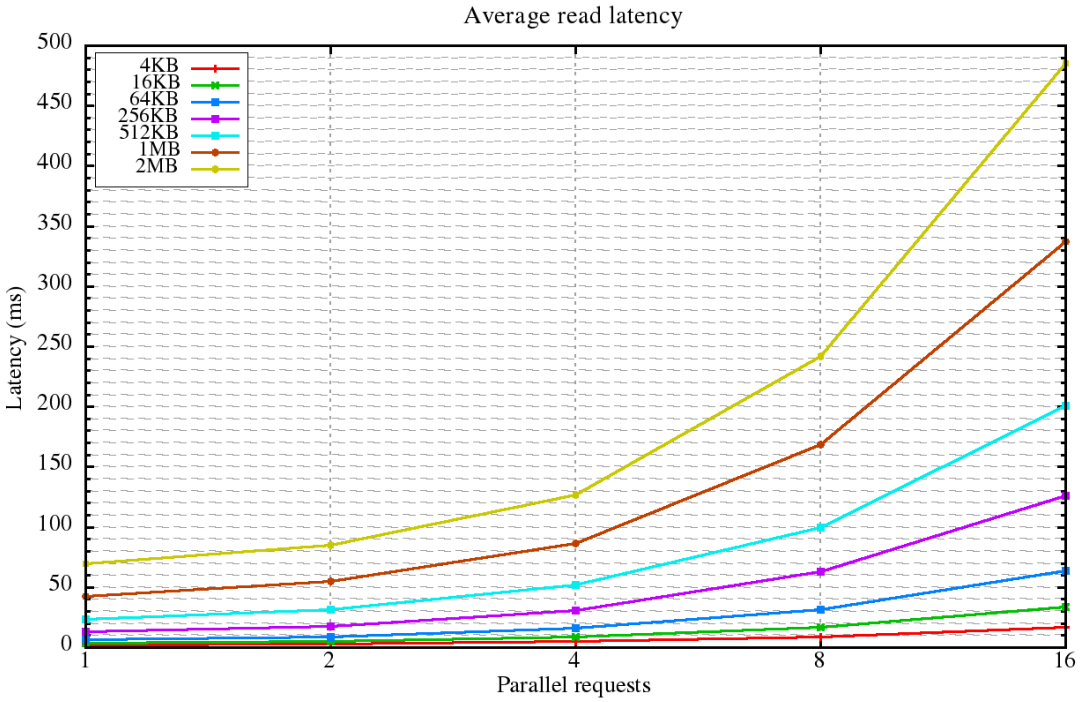
Μετρήθηκε η συμπεριφορά του sosd στα παραπάνω σενάρια για τα μεγέθη αιτήσεων 4KB, 16KB, 64KB, 256KB, 512KB, 1MB, 2MB και για 1, 2, 4, 8, 16 παράλληλες αιτήσεις. Η χρονική διάρκεια κάθε μέτρησης εγγραφής, ορίστηκε στα 150 δευτερόλεπτα συνεχούς λειτουργίας, ενώ για τη μέτρηση των αναγνώσεων χρησιμοποιήθηκε ως όριο ο μέγιστος αριθμός αιτήσεων που ολοκληρώθηκαν κατά το στάδιο μέτρησης εγγραφών. Δίνονται τα αποτελέσματα στους πίνακες Α.1 εώς Α.6) και με μορφή διαγραμμάτων (Διαγράμματα 1 εως 6) για το πρώτο σενάριο, ενώ στους πίνακες Β.1 έως Β.6 και στα αντίστοιχα διαγράμματα 7 έως 12 παρουσιάζονται τα αποτελέσματα για το δεύτερο σενάριο.



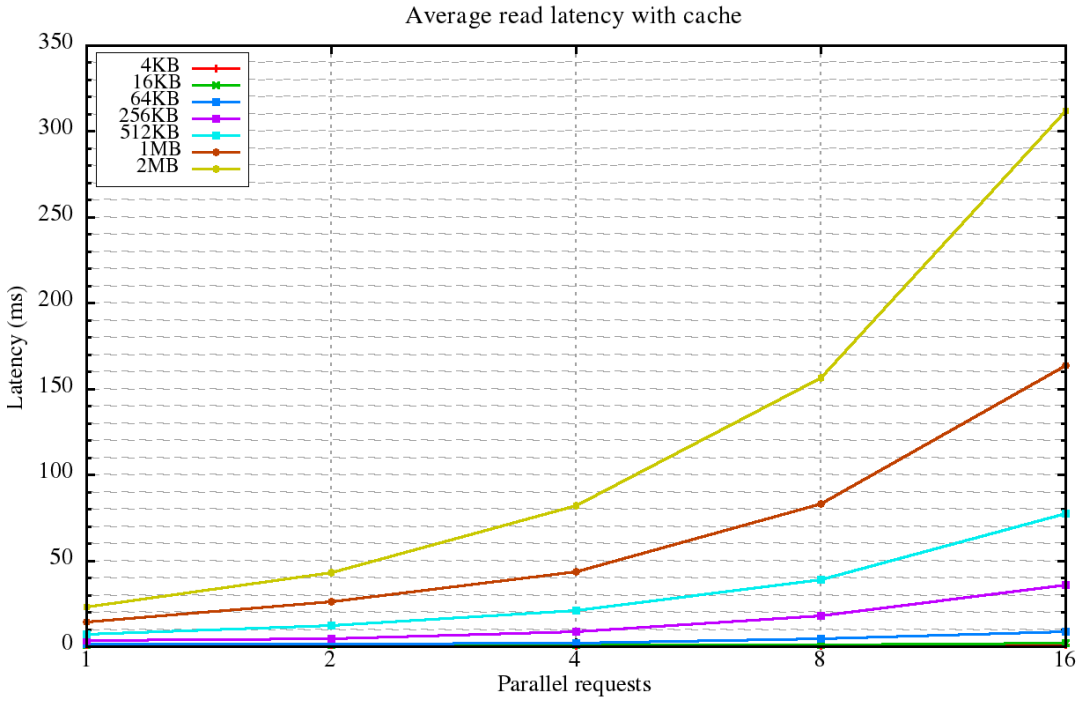
Σχήμα 4.5: Καταμερισμός αιτήσεων στο 2ο σενάριο αξιολόγησης



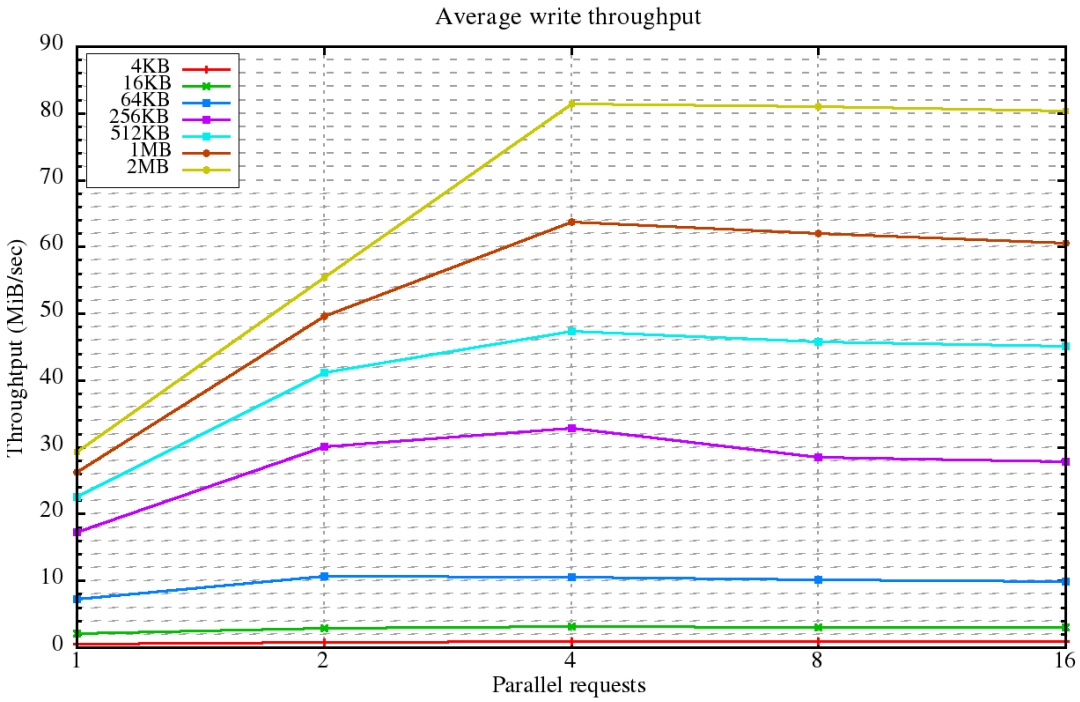
Διάγραμμα 1: Μεσος χρόνος εξυπηρέτησης αίτησης εγγραφής



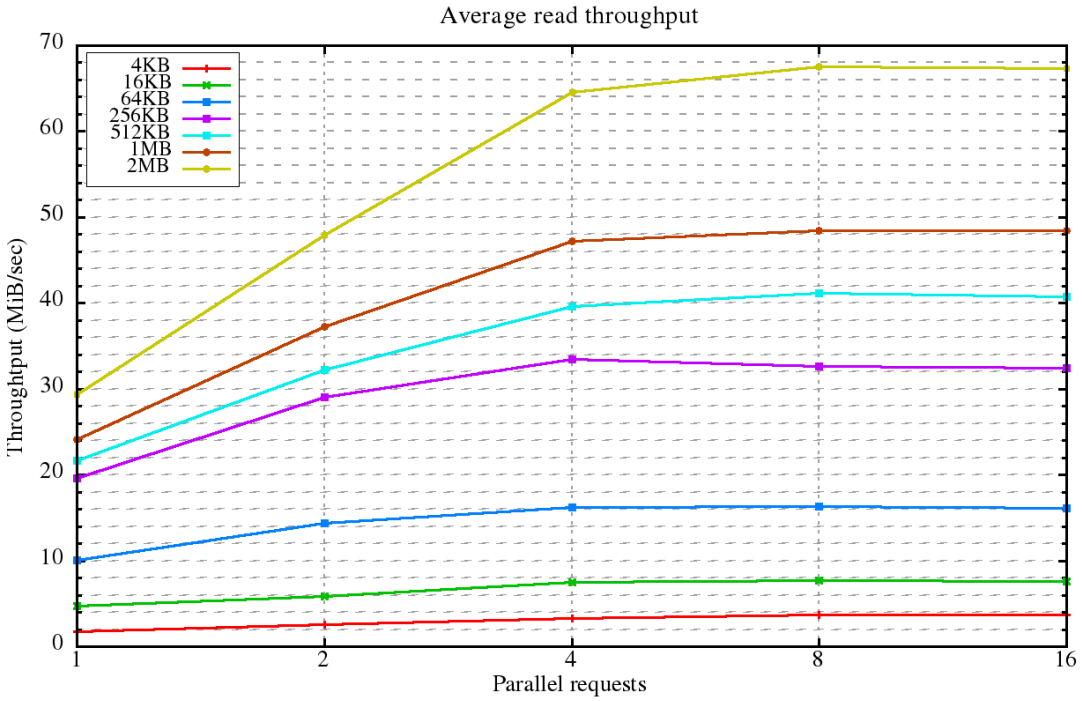
Διάγραμμα 2: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης



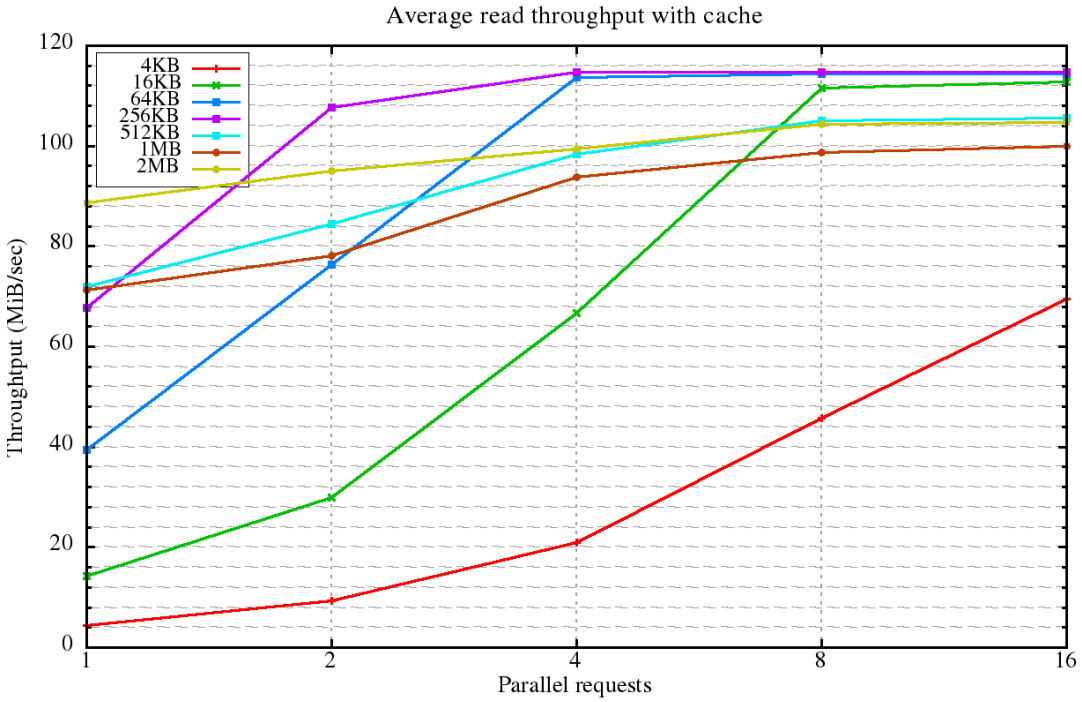
Διάγραμμα 3: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης με χρήση cache στους OSDs



Διάγραμμα 4: Συνολικός ρυθμός διαμεταγωγής για εγγραφές



Διάγραμμα 5: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις



Διάγραμμα 6: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις με χρήση cache στους OSDs

Στο πρώτο σενάριο μετρήσεων κάθε αίτηση απευθύνεται σε διαφορετικά αντικείμενα στον αποθηκευτικό χώρο. Αυτό αντιστοιχεί στη πρόσβαση στον αποθηκευτικό χώρο από διαφορετικές εικονικές μηχανές οι οποίες έχουν διαφορετικούς εικονικούς δίσκους ή και από μια εικονική μηχανή η οποία προσπελαύνει περιοχές του δίσκου που αποθηκεύονται σε διαφορετικά αντικείμενα. Εφόσον οι αιτήσεις αφορούν διαφορετικά αντικείμενα και η εσωτερική κατανομή των αντικείμενων γίνεται με ικανοποιητικό τρόπο, δυο διαδοχικές αιτήσεις είναι πολύ πιθανό να καταλήξουν σε διαφορετικό OSD προς εξυπηρέτηση. Επίσης η εσωτερική αναπαραγωγή των δεδομένων που προκύπτει κατά τη διάρκεια των εγγραφών είναι αρκετά πιθανό να καταλήξει σε διαφορετικούς OSD από τους δυο που δέχτηκαν τις αιτήσεις.

Η πρώτη παρατήρηση η οποία αποτελεί και γενικό κανόνα στις μετρήσεις που λήφθηκαν είναι ότι η αύξηση του μεγέθους των αιτήσεων έχει ως αποτέλεσμα την αύξηση του συνολικού throughput. Αυτό οφείλεται στο γεγονός ότι οι φυσικοί δίσκοι παρουσιάζουν μεγαλύτερες επιδόσεις όταν πρόκειται να αποθηκεύσουν μεγαλύτερο όγκο δεδομένων σε διαδοχικά σημεία του δίσκου.

Η φύση του σεναρίου στην εγκατάσταση όπου λαμβάνονται οι μετρήσεις έχει ώς συνέπεια ότι κατά τη διάρκεια των εγγραφών, στις 2 παράλληλες αιτήσεις είναι αρκετά πιθανό να χρησιμοποιούνται όλοι οι OSD και καθένας να έχει μια εγγραφή να πραγματοποιήσει (είτε κανονική, είτε αναπαραγωγής) χρησιμοποιώντας μόνο το 1 thread στους OSDs. Στις 4 παράλληλες αιτήσεις με βάση μια καλή κατανομή του φόρτου, κάθε OSD θα είναι πλήρης, δηλαδή και τα δυο του threads θα είναι απασχολημένα με μια εγγραφή. Επειδή όμως οι εγγραφές των δυο διαφορετικών threads άνα OSD καταλήγουν στον ίδιο δίσκο, η αύξηση των επιδόσεων δεν είναι ίδια με αυτή του επιμερισμού των εγγραφών σε διαφορετικούς OSD. Σύμφωνα με τις μετρήσεις οι μικρού μεγέθους εγγραφές δεν επωφελούνται σχεδόν καθόλου από τα παράλληλα threads ενώ οι μεγαλύτερου μεγέθους αιτήσεις παρουσιάζουν αύξηση των επιδόσεων. Αυτό συμβαίνει διότι οι φυσικοί δίσκοι παρουσιάζουν καλύτερες επιδόσεις με τις εγγραφές μεγαλύτερου μεγέθους, ενώ ρόλο έχει και το γεγονός οτί οι εγγραφές γίνονται στο journal πρώτα και συνεπώς πρόκειται για διαδοχικές εγγραφές στον δίσκο όπου υπάρχουν περιθώρια βελτιστοποίησης. Ρόλο επίσης διαδραματίζει, το γεγονός ότι όσο μεγαλώνει το μέγεθος μιας αίτησης, αυξάνεται αντίστοιχα και ο χρόνος μετάδοσης των δεδομένων πάνω από το δίκτυο και καταλαμβάνει σημαντικό κομμάτι του χρόνου ολοκλήρωσης μιας αίτησης (π.χ. στις αιτήσεις 2MB ο χρόνος αυτός 16ms). Έτσι, με την αύξηση των παράλληλων αιτήσεων ακόμη και αν δεν υπάρχει καλή κατανομή του φόρτου, υπάρχει επαλληλία της μετάδοσης μιας αίτησης με την επεξεργασία της από τον OSD και άρα επιτυγχάνεται μεγαλύτερος ρυθμός διαμεταγωγής.

Στην περαιτέρω αύξηση των παράλληλων αιτήσεων δεν υπάρχει καμία αύξηση των επιδόσεων, εφόσον δεν υπάρχουν διαθέσιμα threads στους OSDs να τις εξυπηρετήσουν.

Παράλληλα όμως με την αύξηση του throughput, η αύξηση του μεγέθους μιας αίτησης αυξάνει και την καθυστέρηση ολοκλήρωσης της. Επίσης όπως είναι φυσικό από το σημείο όπου η αύξηση των παράλληλων αιτήσεων δεν φέρνει αύξηση του ρυθμού διαμεταγωγής, η μέση καθυστέρηση ολοκλήρωσης κάθε αίτησης διπλασιάζεται αφού υπάρχει διπλάσιος αριθμός αιτήσεων ενεργών, οι οποίες όμως εξυπηρετούνται

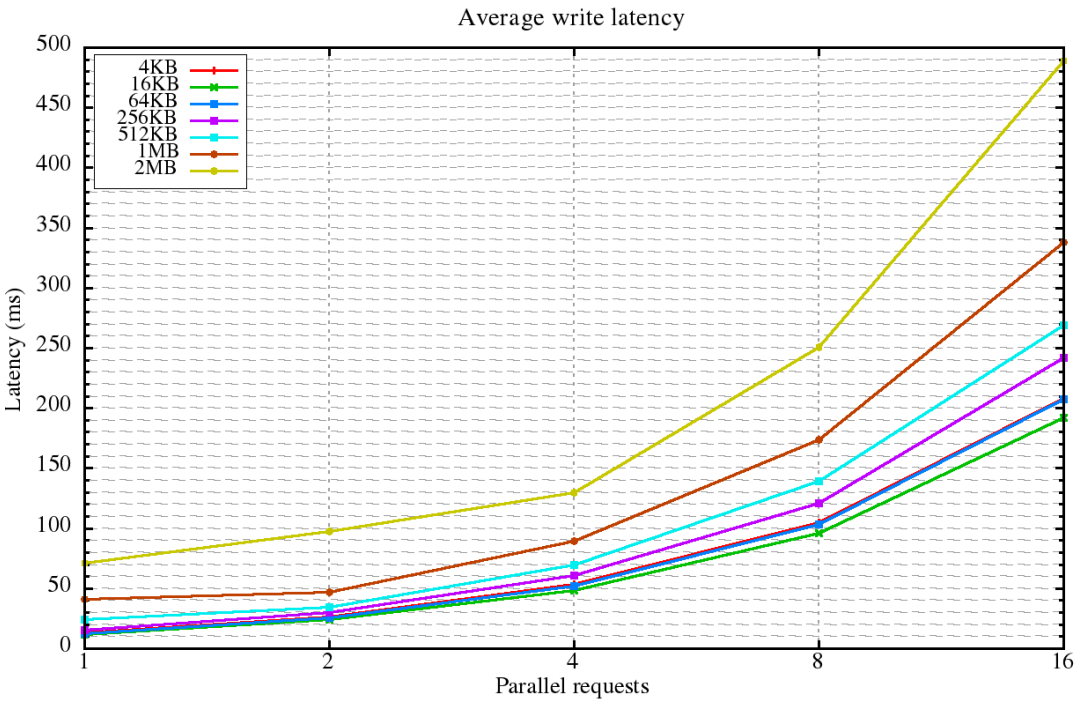
με τον ίδιο ρυθμό.

Στις αναγνώσεις χωρίς χρήση μνήμης page cache, παρουσιάζεται ανάλογη συμπεριφορά με τις εγγραφές, με την εξαίρεση ότι εφόσον δεν υπάρχει αναπαραγωγή δεδομένων, με βάση μια καλή κατανομή των αιτήσεων, μέχρι 4 παράλληλες αιτήσεις κάθε OSD έχει να διεκπεραιώσει μόνο μια αίτηση και συνεπώς ο ρυθμός διαμεταγωγής αυξάνεται με την αύξηση των παράλληλων αιτήσεων. Μετά δε παρουσιάζεται κάποια αύξηση. Ακόμη και με την χρήση του δεύτερου thread του OSD, οι αναγνώσεις απευθύνονται στον ίδιο δίσκο χωρίς κάποια συνοχή και επομένως δεν υπάρχουν μεγάλα περιθώρια βελτίωσης της επίδοσης των αναγνώσεων.

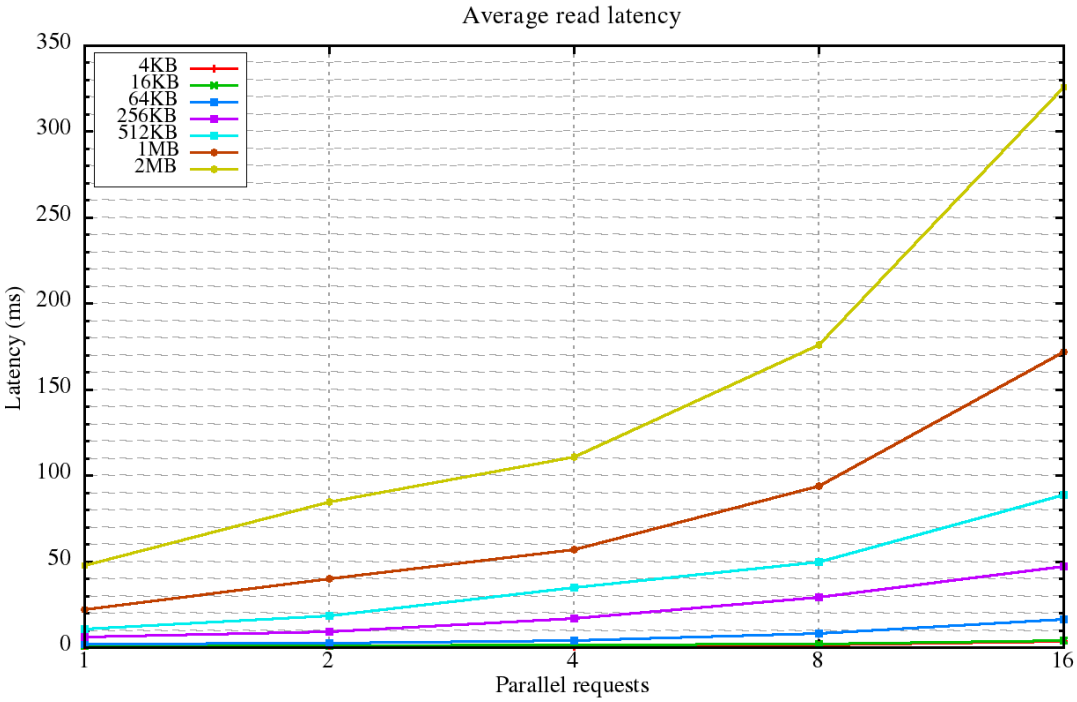
Οι αιτήσεις ανάγνωσης παρουσιάζουν μικρότερη μέση καθυστέρηση ολοκλήρωσης από τις αντίστοιχες αιτήσεις εγγραφών, ενώ όπως και πριν, μετά την επίτευξη του μέγιστου ρυθμού διαμεταγωγής, η μέση καθυστέρηση ολοκλήρωσης απλά διπλασιάζεται με κάθε διπλασιασμό των ενεργών παράλληλων αιτήσεων

Στις αναγνώσεις με τη χρήση pagecache στους OSDs, τα δεδομένα των αντικειμένων προς ανάγνωση, είχαν προσπελαστεί στο παρελθόν και επομένως βρίσκονται στην page cache των OSDs οπότε και η ανάκτηση τους δε γίνεται από το δίσκο. Εξαιτίας αυτού του γεγονότος παρουσιάζουν τις μεγαλύτερες επιδόσεις. Οι αιτήσεις γρήγορα κατακλύζουν το δίκτυο επικοινωνίας και φτάνουν το μέγιστο όριο του δικτύου διασύνδεσης του πελάτη. Οι μεγαλύτερου μεγέθους αιτήσεις το επιτυγχάνουν πιο γρήγορα ενώ καταφέρνουν να το επιτύχουν και οι αιτήσεις με μέγεθος 16K. Στις αναγνώσεις αυτές παρουσιάζεται αύξηση των επιδόσεων ακόμη και με τις 8 παράλληλες αιτήσεις εφόσον τα 2 threads των OSD δεν συναγωνίζονται για τη χρήση του δίσκου, ενώ αύξηση παρουσιάζουν ακόμη και οι 16 παράλληλες ενεργές αιτήσεις, γεγονός που δείχνει ότι με τη ταχύτητα εξυπηρέτησης των αιτήσεων από τους OSDs ο χρόνος για τη δημιουργία νέας αίτησης από το πελάτη είναι συγκρίσιμος με το χρόνο εξυπηρέτησης τους από τους OSDs.

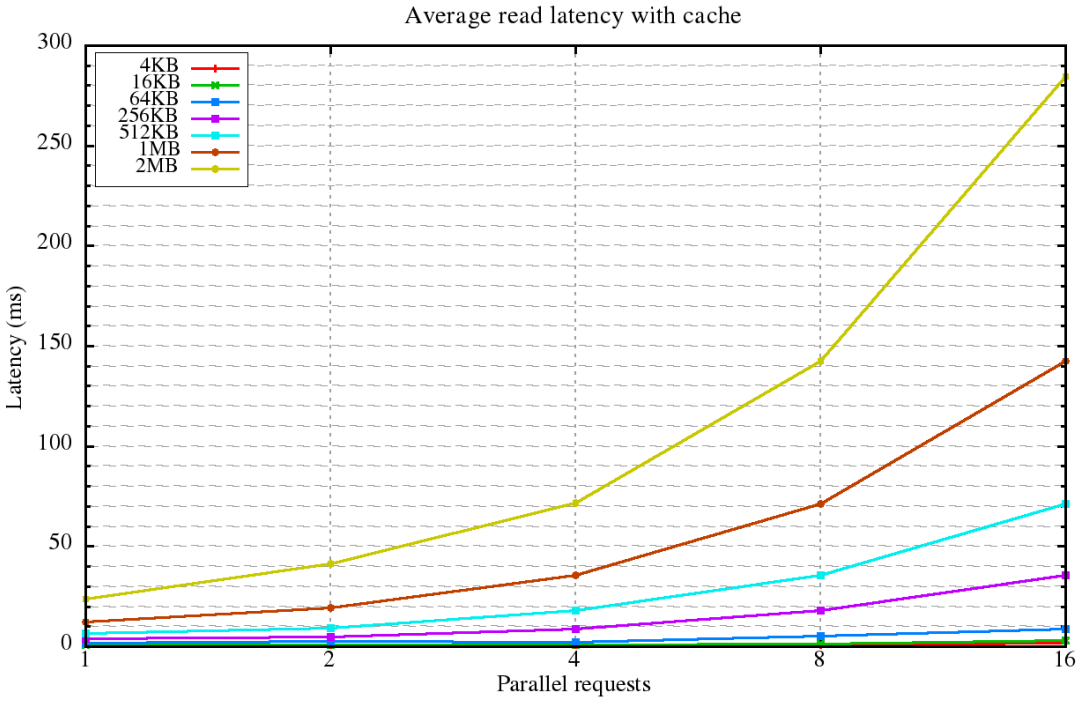
Όπως είναι φυσικό, με την επίτευξη του μέγιστου ρυθμού διαμεταγωγής ο χρόνος ολοκλήρωσης των αιτήσεων διπλασιάζεται, ενώ όλοι οι χρόνοι ολοκλήρωσης σε σχέση με τις αναγνώσεις χωρίς pagecache είναι αρκετά μικρότεροι. Παρατηρείται επίσης οτί για τα αντικείμενα μεγέθους 4K όπου ο χρόνος μεταφοράς των δεδομένων από το πελάτη στους OSDs μέσω του δικτύου επικοινωνίας είναι σχεδόν αμελητέος, απαιτείται ένας ελάχιστος χρόνος περίπου 1ms για την εξυπηρέτηση μιας αίτησης.



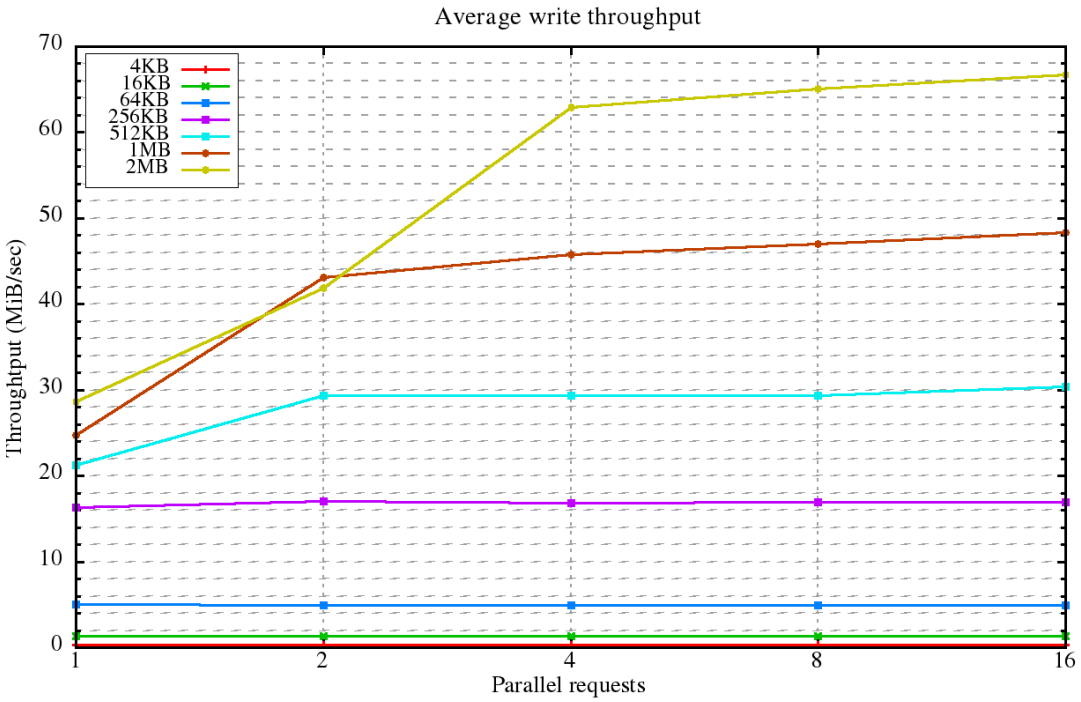
Διάγραμμα 7: Μεσος χρόνος εξυπηρέτησης αίτησης εγγραφής



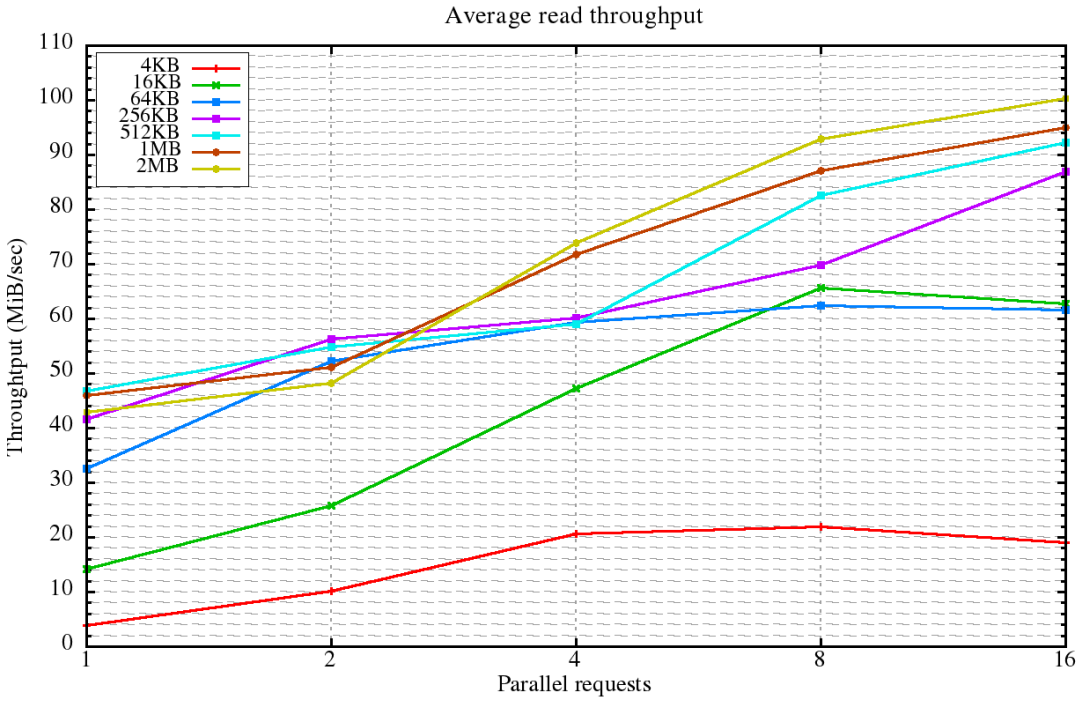
Διάγραμμα 8: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης



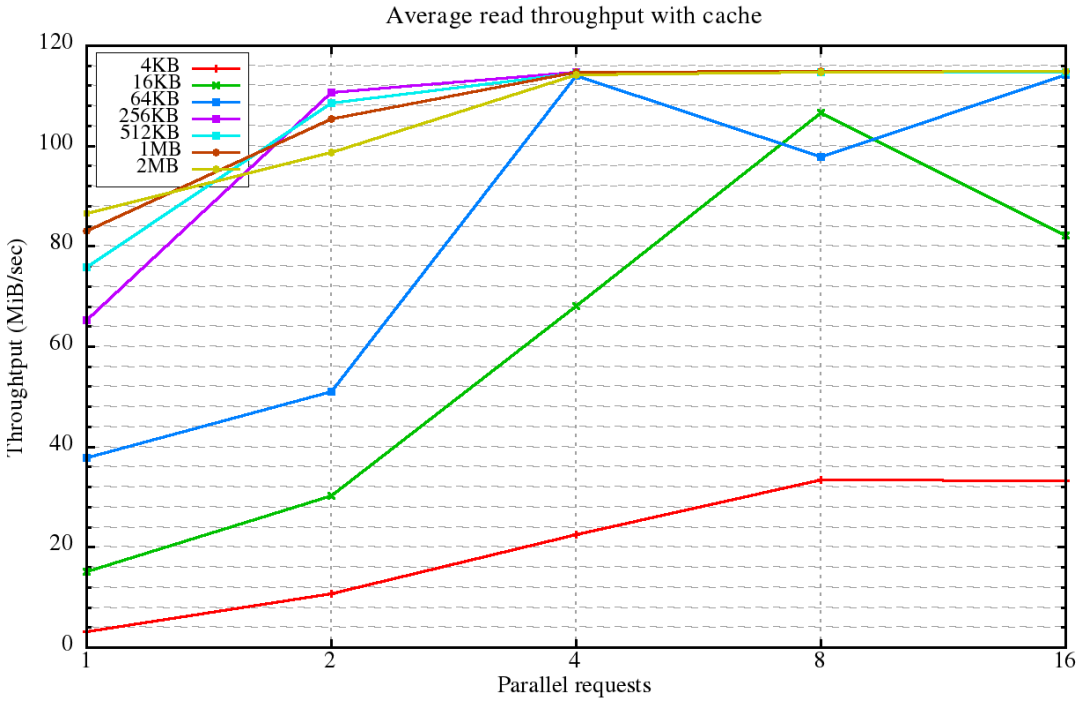
Διάγραμμα 9: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης με χρήση cache στους OSDs



Διάγραμμα 10: Συνολικός ρυθμός διαμεταγωγής για εγγραφές



Διάγραμμα 11: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις



Διάγραμμα 12: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις με χρήση cache στος OSDs

Στο δεύτερο σενάριο μετρήσεων υπάρχει διαδοχική πρόσβαση σε κάθε αντικείμενο έως ότου προσπελαστεί ολόκληρο και αντιστοιχεί στη διαδοχική πρόσβαση στον αποθηκευτικό χώρο από μια εικονική μηχανή. Αυτό σημαίνει πως κάθε αίτηση αποτελεί συνέχεια της προηγούμενης της στο ίδιο αντικείμενο και συνεπώς καταλήγει στον ίδιο OSD με την προηγούμενη έως ότου υπάρξει αλλαγή αντικειμένου. Η αλλαγή αντικεμένου συσχετίζεται άμεσα με το μέγεθος των αιτήσεων. Ο αριθμός των αιτήσεων που απαιτούνται για να προσπελαστεί ολόκληρο είναι ίσος με το μέγεθος του αντικειμένου προς το μέγεθος των αιτήσεων. Έτσι οι αιτήσεις μικρότερου μεγέθους μετακινούνται πιο αργά από ένα αντικείμενο στο επόμενο. Συγκεκριμένα για τις αιτήσεις μεγέθους 4K απαιτούνται 1024 αιτήσεις για την προσπέλαση του αντικειμένουν, ενώ για τις αιτήσεις μεγέθους 2MB μόνο 2. Το παραπάνω έχει ώς συνέπεια τη κατανομή του φόρτου ασύμμετρα, καθώς ο φορτος της εξυπηρέτησης των αιτήσεων κάθε στιγμή είναι σε έναν OSD ενώ οι υπόλοιποι παραμένουν αδρανείς, ενώ οι παράλληλες αιτήσεις καταλήγουν να εξυπηρετούνται από έναν OSD.

Εφόσον λοιπόν οι παράλληλες αιτήσεις εγγραφής καταλήγουν στον ίδιο OSD όπου έχουν να συναγωνιστούν για τον ίδιο δίσκο, οι μικρού μεγέθους αιτήσεις όπου δεν επωφελούνται από τη χρήση του δεύτερου thread στους OSD, διατηρούν σταθερό ρυθμό διαμετεγωγής κατά την αύξηση των παράλληλων αιτήσεων. Οι μεγαλύτερου μεγέθους αιτήσεις παρουσίαζουν μικρή αύξηση με τη χρήση του δεύτερου thread στους OSD ενώ η χρήση παραπάνω από δυο παράλληλες αιτήσεις δεν βελτιώνει το ρυθμό διαμεταγωγής. Οι μεγαλύτερου μεγέθους αιτήσεις, πέραν από την εκμεταλλευση του δεύτερου thread, έχουν μεγαλύτερο χρόνο μεταφοράς μιας αίτησης και εκμεταλλεύονται το γεγονός της επαλληλίας του χρόνου μεταφοράς από το χρόνο επεξεργασίας μιας αίτησης από τον OSD. Στις αιτήσεις μεγέθους 2M η αύξηση από τις δυο στις τέσσερις παράλληλες αιτήσεις φέρει σημαντική βελτίωση στο ρυθμό διαμεταγωγής. Αυτό οφείλεται τόσο στο γεγονός ότι λόγω του μεγέθους των αιτήσεων οι πρώτες δυο απευθύνονται σε έναν OSD ενώ οι άλλες δυο πιθανόν να καταλήγουν σε άλλον OSD και συνεπώς ο φόρτος να επιμμερίζεται, όσο και στο γεγονός ότι έχουν τον μεγαλύτερο χρόνο μεταφοράς και επεργασίας μιας αίτησης, οπότε ακόμη και στον ίδιο OSD να καταλήγουν οι αιτησεις, μπορεί να υπάρξει επαλληλλία του χρόνου μεταφοράς και επεξεργασίας και επομένως να υπάρξει βελτίωση του ρυθμού διαμεταγωγής.

Όπως είναι φυσικό, με την επίτευξη του μέγιστου ρυθμού διαμεταγωγής, ο μέσος χρόνος ολοκλήρωσης κάθε αίτησης διπλασιάζεται με τον διπλασιασμό των παράλληλων ενεργών αιτήσεων.

Στις αναγνώσεις χωρίς τη χρήση μνήμης page cache στους OSDs παρατηρείται αύξηση της επίδοσης καθώς αυξάνονται οι παράλληλες αιτήσεις για όλα τα μεγέθη των αιτήσεων. Λόγω του γεγονότος οτί οι αιτήσεις κατανέμονται ασύμμετρα στους OSDs, οι μικρού μεγέθους αιτήσεις επωφελούνται από την αύξηση του αριθμού των παράλληλων αιτήσεων, καθώς οι OSD εκτελούν διάφορες τεχνικές βελτιστοποίησης της απόδοσης τους (πχ readahead cache), ενώ η σειριοποίηση των αναγνώσεων του ίδιου αντικειμένου στο δίσκο μπορεί να οδηγήσει σε συνέννωση των αιτήσεων και σε μια μεγαλύτερη αίτηση ανάγνωσης από το φυσικό μέσο, η οποία είναι σαφώς γρηγορότερη στους δίσκους. Από αυτό το γεγονός, επωφελούνται και αιτήσεις

με μεγαλύτερο όγκο δεδομένων. Οι αιτήσεις με μεγαλύτερο όγκο δεδομένων επωφελούνται επίσης και από το γεγονός οτί το μεγάλο μέγεθος οδηγεί σε γρήγορη αλλαγή αντικειμένου και άρα οι παράλληλες ταυτόχρονες αιτήσεις κατανέμονται σε περισσότερους το ενός OSD. Έτσι ο φόρτος κατανέμεται σε περισσότερους το ενός OSD με τις αιτήσεις να παραμένουν όμως σειριακές στο ίδιο αντικείμενο σε κάθε OSD και άρα εμφανίζονται ακόμη μεγαλύτερες επιδόσεις. Άρα συνολικά το σενάριο αυτό παρουσιάζει καλύτερες επιδόσεις και συμπεριφορα στην ανάγνωσεις χωρίς cache.

Στις αναγνώσεις χωρίς τη χρήση pagecache αξίζει να σημειωθεί οτί ο μέσος χρόνος ολοκλήρωσης μιας αίτησης είναι συγκρίσιμος για τις αιτήσεις μικρού μεγέθους με με τους χρόνους των αναγνώσεων με τη χρήση pagecache. Αυτό οφείλεται στις βελτιστοποιήσεις που υπεισέρχονται κατά τη διάρκεια των σειρακών αναγνώσεων στο ίδιο αντικείμενο και ιδίως στην readahead cache με την οποία αποφεύγονται οι οι αναγνωσεις από το δίσκο για αρκετές αιτήσεις.

Στις αναγνώσεις με χρήση μνήμης cache από τους OSDs, στις αιτήσεις με μεγαλύτερο όγκο δεδομένων (μεγαλύτερο από 256KB) παρουσιάζεται γρήγορα η μέγιστη επίδοση ήδη από τις 2 παράλληλες αιτήσεις που περιορίζεται από το δίκτυο επικοινωνίας του πελάτη, όπως και στο πρώτο σενάριο. Στις μικρότερου μεγέθους αιτήσεις, παρατηρείται αύξηση στις επιδόσεις με την αύξηση των παράλληλων αιτήσεων, με μοναδικό χαμένο τις αιτήσεις των 4KB που παρουσιάζουν μικρότερες επιδόσεις από τις αντίστοιχες του πρώτου σεναρίου και δεν επωφελείται της αύξησης των παράλληλων αιτήσεων πέρα από τις οκτώ ενεργές αιτήσεις. Αυτό οφείλεται στο γεγονός ότι στις αιτήσεις αυτού του μεγέθους η εναλλαγή των αντικειμένων είναι η πιο αργή και εξυπηρετείται για αρκετές συνεχόμενες αιτήσεις από έναν OSD. Έτσι εφόσον υπάρχουν μόνο δυο ενεργά thread άνα OSD, η άυξηση των παράλληλων αιτήσεων έχει ως αποτέλεσμα την αύξηση των αιτήσεων που περιμένουν να ολοκληρωθούν στους OSD η οποία δε δίνει καμία αύξηση των επιδόσεων.

Στις αναγνώσεις με τη χρήση page cache από τους OSD παρατηρούνται και πάλι πολύ μικροί χρόνοι εξυπηρέτησης και συνε- πώς καθυστέρησης των αιτήσεων, λόγω του ότι η ανάκτηση των δεδομένων γίνεται από τη RAM η οποία είναι τάξεις μεγέθους γρηγορότερη από το φυσικό αποθηκευ- τικό μέσο.

Σύμφωνα με τα αποτελέσματα των μετρήσεων, είναι εμφανές οτί και στα 2 σενάρια οι αιτήσεις μεγαλύτερου μεγέθους οδηγούν σε μεγαλύτερο συνολικό throughput ανεξαρτήτως παράλληλων αιτήσεων. Συνοδεύονται όμως από μεγαλύτερο χρόνο εξυπηρέτησης της αίτησης, λόγω του ότι περισσότερα δεδομένα πρέπει να μεταβιβαστούν μέσω του δικτύου, αλλά και να διαβαστούν ή να εγγραφούν στο αποθηκευτικό σύστημα. Συνεπώς από άποψη latency, η αιτήσεις μικρότερου μεγέθους υπερτερούν, ενώ από άποψη throughput καλύτερη είναι η δημιουργία μεγαλύτερων αιτήσεων.

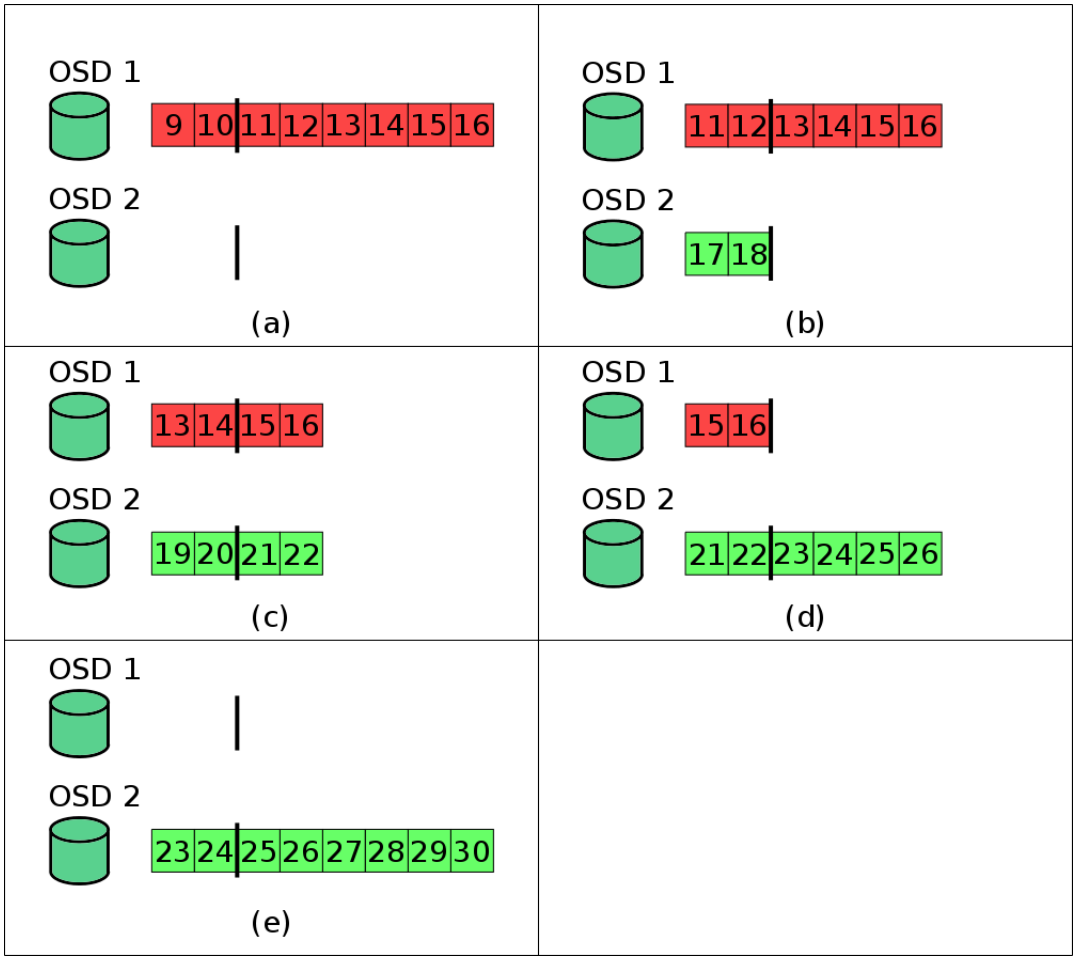
Σημαντικό ρόλο στα παραπάνω αποτελέσματα των μετρήσεων διαδραμάτισε ο αριθμός των ενεργών thread ανά OSD και το scheduling προς τον δίσκο που γινόταν απο τους OSD υπό το συγκεκριμένο περιορισμό. Συγκεκριμένα, ακόμη και αν υπήρχαν 4 αιτήσεις προς τον συγκεκριμένο OSD, μόνο οι δυο θα επεργάζονταν ταυτόχρονα ενώ οι άλλες δυο θα βρίσκονταν στην ουρά προς εξυπηρέτηση. Έτσι, ο OSD και το λειτουργικό του σύστημα αναλαμβάνει την εξυπηρέτηση των αιτήσεων από-/προς το δίσκο γνωρίζοντας μόνο τις δυο αιτήσεις και λαμβάνοντας υπόψιν μόνο αυ-

τές για τον προγραμματισμό των αιτήσεων προς το δίσκο. Άρα ακόμη και αν οι άλλες αιτήσεις βρίσκονται σε διαδοχικά σημεία στο δίσκο και επιδέχονται βελτίωσης των επιδόσεων τους, ο OSD αγνοεί το γεγονός αυτό και περιμένει να ολοκληρωθούν οι δυο προηγούμενες πρωτού αναλάβει την εξυπηρέτηση τους.

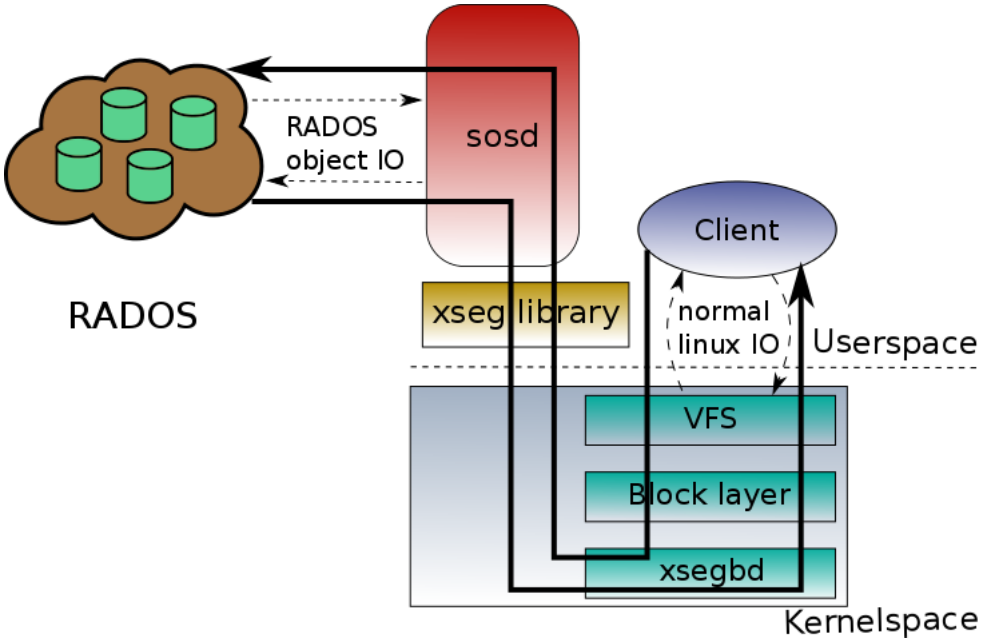
Η ιδιότητα αυτή του συστήματος, πέρα από το περιορισμό της πληροφορίας προς τον IO scheduler, σηματοδοτεί και άλλη μια ιδιομορφία του συστήματος. Επιτρέπει την κατανομή των αιτήσεων στους OSDs ακόμη και αν αυτό δεν είναι άμεσα ορατό από το μοντέλο πρόσβασης και τον αριθμό των παράλληλων αιτήσεων. Αυτό εξηγείται ως εξής: ενώ υπάρχουν περισσότερες από δυο παράλληλες αιτήσεις στο σύστημα, αυτές εξυπηρετούνται ανά δυο και οι υπόλοιπες περιμένουν στην ουρά. Κατά την αλλαγή αντικειμένου στο οποίο απευθύνονται οι αιτήσεις, υπάρχουν στιγμές όπου υπάρχουν ταυτόχρονα ενεργές αιτήσεις προς δυο αντικείμενα: αυτές που υπήρχαν ήδη στην ουρά και οι καινούργιες που προστίθενται προς το νέο αντικείμενο. Έτσι εφόσον οι καινούργιες αιτήσεις απευθύνονναι σε διαφορετικό αντικείμενο, υπάρχει επεξεργασία των ενεργών αιτήσεων από δυο OSDs, εως ότου ολοκληρωθούν οι προηγούμενες αιτήσεις που έχουν απομείνει στην ουρά. Για παράδειγμα στις σειριακές αιτήσεις ανάγνωσης μεγέθους 256KB, παρατηρείται αύξηση των επιδόσεων από τις τέσσερις στις οκτώ παράλληλες ενεργές αιτήσεις, παρόλο που η αλλαγή αντικειμενου γίνεται στις 16 αιτήσεις. Η αύξηση αυτή οφείλεται στη παραλληλοποίηση μέρους των αιτήσεων που βρίσκονται ενεργές κατά τη διάρκεια της αλλαγής του αντικειμένου, εφόσον το επόμενο αντικείμενο καταλήγει σε διαφορετικό OSD, εξαιτίας του περιορισμού της επεξεργασίας δυο μόνο αιτήσεων τη φορά από έναν OSD (Σχήμα 4.6).

4.3.2 Αξιολόγηση xsegbd

Για την αξιολόγηση του οδηγού συσκευής για τη πρόσβαση στους τόμους που βρίσκονται αποθηκευμένοι στο αποθηκευτικό σύστημα, χρησιμοποιήθηκε το λογισμικό dd το οποίο αποτελεί μέρος των βασικών προγραμμάτων που συνοδεύουν συνήθως μια διανομή του λειτουργικού συστήματος GNU/linux. Το πρόγραμμα αυτό διαθέτει τη δυνατότητα μεταφοράς δεδομένων από ένα αρχείο σε ένα άλλο κατά block δεδομένων, με δυνατότητα ρύθμισης του μεγέθους αυτού. Διαθέτει επίσης και επιλογές για το τρόπο ανοίγματος των αρχείων (open flags), με τις οποίες ρυθμίζεται ο τρόπος εγγραφής ή ανάγνωσης από αυτά. Η επιλογή που χρησιμοποιήθηκε κατά τη διάρκεια των δοκιμών ήταν η επιλογή O_DIRECT. Η επιλογή αυτή δίνει εντολή σε κάθε εγγραφή ή ανάγνωση από αρχείο να παρακάμπτει τη pagecache του πυρήνα, και να κατευθύνει τις εντολές εισόδου/εξόδου κατευθείαν στη συσκευή. Οι εντολές αυτές επιστρέφουν όταν τελειώσει η επεξεργασία τους από τη συσκευή, χωρίς να δίνεται καμία όμως εγγύηση οτί βρίσκονται σε μόνιμο αποθηκευτικό μέσο. Παράδειγμα σε αυτό, είναι η χρήση κρυφής μνήμης από τις συσκευές δίσκων, όπου οι εγγραφές θεωρούνται ολοκληρωμένες, όταν εγγραφούν σε αυτή και προτού τα δεδομένα εγγραφούν στο μόνιμο αποθηκευτικό χώρο. Εφόσον όμως η αποθηκευτική υποδομή που εξετάζουμε δε διαθέτει τέτοιου είδους λειτουργικότητα αλλά επιστρέφει την εγγραφή όταν βρίσκεται στο μόνιμο αποθηκευτικό χώρο, τα δεδομένα είναι ασφαλή. Για τις



Σχήμα 4.6: Παράδειγμα εξυπηρέτησης αιτήσεων ανάγνωσης μεγέθους 256KB σειριακής πρόσβαση με 8 παράλληλες αιτήσεις με 2 ενεργά thread ανά OSD



Σχήμα 4.7: Μονοπάτι δεδομένων στη συσκευή xsegbd με τη χρήση του προγράμματος dd

δοκιμές χρησιμοποιήθηκε μηχάνημα ίδιων δυνατοτήτων με τα μηχανήματα που χρησιμοποιήθηκαν για την εγκατάσταση του RADOS στις προηγούμενες δοκιμές, ενώ η βιβλιοθήκη επικοινωνίας xseg ρυθμίστηκες για μέγιστο μέγεθος αίτησης 256KB. Για τις δοκιμές απευθείας στη συσκευη, εκτελέστηκαν οι εντολές:

```
dd if=/dev/zero of=/dev/xsegbd bs=blocksize count=nrblocks of=direct
dd if=/dev/xsegbd of=/dev/null bs=blocksize count=nrblocks if=direct
```

για τη μέτρηση των εγγραφών και των αναγνώσεων αντίστοιχα. Για τις εγγραφές χρησιμοποιήθηκε η συσκευή /dev/zero για τα δεδομένα εισόδου, ή οποία είναι μια εικονική συσκευή που επιστρέφει μηδενικά σε κάθε ανάγνωση σε αυτή. Για τις αναγνώσεις, χρησιμοποιήθηκαν τα δεδομένα που είχαν εγγραφεί κατά το στάδιο των μετρήσεων εγγραφής, και σαν αποδέκτης η συσκευή /dev/null η οποία απλά δέχεται κάποιες εγγραφές και ”πετά” τα δεδομένα. Πρόκειται για ειδικές συσκευές σχεδιασμένες για τέτοιους σκοπούς και η επιλογή τους έγινε για τη αποφυγή παρεκκλίσεων στις μετρήσεις. Το μονοπάτι δεδομένων που ακολουθείται κατά τη παραπάνω διαδικασία περιγράφεται στο σχήμα 4.7. Οι μετρήσεις πραγματοποιήθηκαν για μεγέθη block 4KB, 16KB, 64KB, 256KB, 512KB, 1MB, 2MB για επαρκή όγκο δεδομένων. Με αυτό το τρόπο μέτρησης, τα δεδομένα προσπελαύνονται σειριακά μέσα στο τόμο και επομένως ο τρόπος πρόσβαση είναι αντίστοιχος του πρώτου σεναρίου από την αξιολόγηση του sosd. Τα αποτελέσματα των μετρήσεων φαίνονται στο διάγραμμα 13.

Στη συνέχεια, η συσκευή προς δοκιμή συνδέθηκε με τον εικονικό δίσκο μιας εικονικής μηχανής, ρυθμίστηκε ο επόπτης να μην κάνει caching των δεδομένων του

δίσκου, και δοκιμάστηκαν με παρόμοιο τρόπο οι επιδόσεις της. Συγκεκριμένα ακολουθήθηκαν οι εντολές:

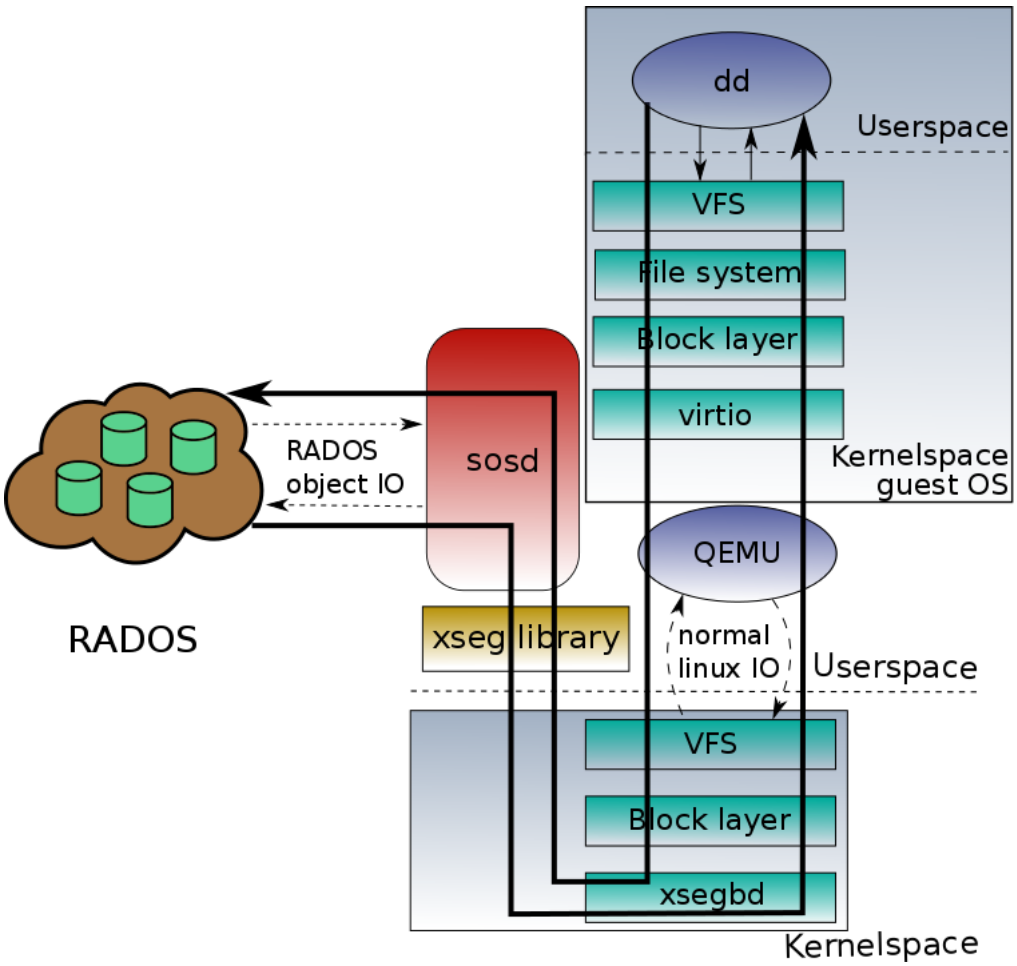
```
dd if=/dev/zero of=foo bs=blocksize count=nrblocks of=direct
dd if=foo of=/dev/null bs=blocksize count=nrblocks if=direct
```

για εγγραφή και ανάγνωση για τα ίδια μεγέθη block και τον ίδιο όγκο δεδομένων. Η διαφορά εδώ είναι ότι τα δεδομένα δε γράφονται απευθείας πάνω στη συσκευή, αλλά σε αρχείο στο σύστημα αρχείων του τόμου που βρίσκεται εγκατεστημένο στη συσκευή. Έτσι παρεμβάλλονται και οι λειτουργίες του συστήματος αρχείων, καθώς και οποιαδήποτε λειτουργία μπορεί να εκτελεί το λειτουργικό σύστημα της εικονικής μηχανής. Παρόλα αυτά, ο τρόπος πρόσβασης στα δεδομένα είναι σειριακά μέσα στην εικόνα του δίσκου, καθώς το σύστημα αρχείων τοποθετεί διαδοχικά δεδομένα του ίδιου αρχείου σε διαδοχικές θέσεις στο αποθηκευτικό μέσο. Ένα ελαφρώς απλοποιημένο μονοπάτι δεδομένων για αυτή τη μέτρηση δίνεται στο σχήμα 4.8. Τα αποτελέσματα των μετρήσεων δίνονται στο διάγραμμα 14.

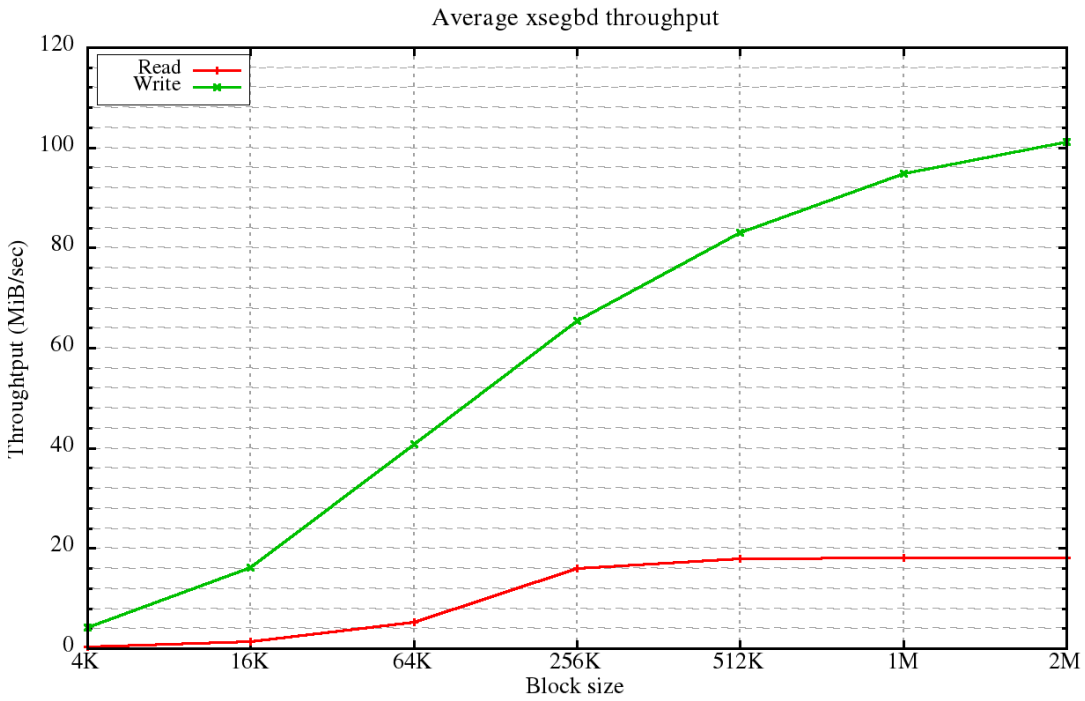
Για την αξιολόγηση των αποτελεσμάτων θα πρέπει να ληφθεί υπόψιν η επιλογή του ορίου δεδομένων της αίτησης της βιβλιοθήκης επικοινωνίας xseg. Εφόσον αυτό ορίστηκε στα 256KB, για μεταφορές δεδομένων ανά block μέχρι αυτό το όριο υπάρχει μόνο μια αίτηση κάθε χρονική στιγμή ενεργή, καθώς τα δεδομένα χωράνε σε μια αίτηση της βιβλιοθήκης xseg. Για μεγαλύτερα μεγέθη block δεδομένων, προκύπτουν περισσότερες παράλληλες ενεργές αίτησης. Αυτό συμβαίνει καθώς καθε αίτηση με μεγαλύτερο μέγεθος δεδομένων από το όριο της βιβλιοθήκης xseg, μετατρέπεται εσωτερικά σε περισσότερες αιτήσεις της βιβλιοθήκης xseg, οι οποίες δίνονται για εκτέλεση παράλληλα. Εξαιτίας αυτού του γεγονότος οι παραπάνω μετρήσεις είναι συγκρίσιμες μόνο με τις αντίστοιχες μετρήσεις για τον sosd, δηλαδή για τα μεγέθη block 4K έως 256K μια παράλληλη αίτηση, ενώ για τα μεγέθη 512K, 1M, 2M, υπάρχουν 2,4,8 παράλληλες αιτήσεις των 256K.

Συγκρίνοντας τις δυο διαφορετικές μετρήσεις απευθείας στη συσκευή και μέσα από την εικονική μηχανή, παρατηρείται παρόμοια συμπεριφορά καθώς αυξάνεται το μέγεθος του block δεδομένων που εγγράφονται ή διαβάζονται, ενώ οι μετρήσεις στην εικονική μηχανή είναι σταθερά πιο αργες. Η συμπεριφορά αυτή ήταν αναμενόμενη καθώς υπεισέρχονται πλήθως παραμέτρων στην εκτέλεση της μέτρησης μέσα από την εικονική μηχανή, όπως το overhead του virtualization, του filesystem, και του μικρού buffer που έχει ο hypervisor(kvm). Επίσης κάθε μεγαλύτερη αίτηση εισόδου/εξόδου στο αρχείο δε μετατρεπόταν πάντα σε αιτήσεις των 256KB αλλά κάποιες φορές προέκυπταν περισσότερες, λόγω εσωτερικών λειτουργιών του λειτουργικού συστήματος της εικονικής μηχανής.

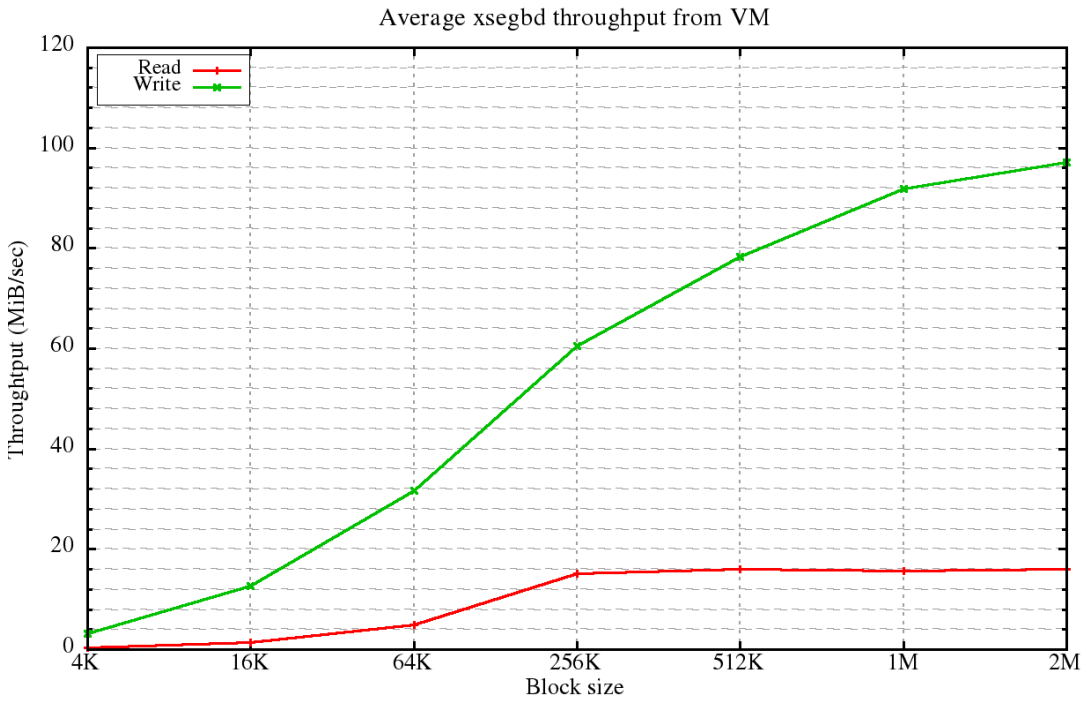
Αξιολογώντας τα αποτελέσματα των μετρήσεων σε σχέση με τις μετρήσεις απευθείας στο sosd παρατηρούνται παρόμοια αποτελέσματα υπό τις ίδιες συνθήκες αιτήσεων ανάγνωσης και εγγραφής.



Σχήμα 4.8: Μονοπάτι δεδομένων στη συσκευή xsegbd με τη χρήση του προγράμματος dd μέσα από εικονική μηχανή



Διάγραμμα 13: Ρυθμός διαμεταγωγής xsegbd απευθείας πάνω στη συσκευή



Διάγραμμα 14: Ρυθμός διαμεταγωγής xsegbd μέσα από την εικονική μηχανή

Κεφάλαιο 5

Επίλογος

5.1 Σύνοψη

Κατά τη παραπάνω μελέτη, αναπτύχθηκε το πρόβλημα που ανακύπτει κατά τη δημιουργία εικονικών μηχανών στην IaaS υπηρεσία του okeanos και παρουσιάστηκε μια λύση για την αντιμετώπιση του καθώς και η αξιολόγηση της.

5.2 Χρήση και μελλοντικές επεκτάσεις

Η λύση που παρουσιάστηκε στη παρούσα διπλωματική εργασία αναμένεται να αποτελέσει μέρος του λογισμικού που υποστηρίζει την υπηρεσία να βελτιώσει τις επιδόσεις και την εμπειρία των χρηστών της. Χρήζει βέβαια αρκετών περιθωρίων βελτίωσης προκειμένου να παρουσιάσει ακόμη καλύτερες επιδόσεις.

Μια άμεση μελλοντική επέκταση της παρούσας εργασίας είναι η μελέτη των παραπάνω λύσεων λογισμικού, σε διαφορετική φυσική εγκατάσταση για την απαλοιφή ορισμένων φυσικών περιορισμών του υλικού όπως είναι το δίκτυο διασύνδεσης Ethernet 1Gpbs το οποίο περιόριζε το ρυθμο διαμεταγωγής. Επίσης άμεσο ενδιαφέρον παρουσιάζει η συμπεριφορά της λύσης σε φυσική εγκατάσταση η οποία εκμεταλεύεται πιο ταχύτερα αποθηκευτικά μέσα, όπως π.χ. αποθηκευτικούς δίσκους SSD.

Ακόμη αρκετά χρήσιμο θα ήταν η ανάπτυξη ειδικού οδηγού συσκευής για το λογισμικό εικονοποίησης qemu, ο οποίος θα έδινε άμεση πρόσβαση στον εικονικό δίσκο της μηχανής μέσω της βιβλιοθήκης xseg, παρακάμπτοντας έτσι το block layer του πυρήνα.

Επίσης μια πιθανή μελλοντική κατεύθυνση που θα μπορούσε να ερευνηθεί, είναι η χρήση κρυφής μνήμης και buffers σε κάθε κόμβο που χρησιμοποιεί τη παραπάνω λύση, κατά τη πρόσβαση στον αποθηκευτικό χώρο. Έτσι θα μπορούσαμε να εκμεταλλευτούμε τη τοπική μνήμη για τη προσωρινή διακράτηση των δεδομένων ώστε να αποφεύγεται η χρήση του δικτύου διασύνδεσης και του συστήματος αποθήκευσης, καθώς επίσης και τη συνένωση πολλών διαδοχικών αιτήσεων εγγραφής σε μεγαλύτερες, οι οποίες επιτυγχάνουν καλύτερους ρυθμούς διαμεταγωγής. Για τα παραπάνω θα

πρέπει να ληφθεί υπόψιν η κατανεμημένη φύση των υποδομών και να διασφαλιστεί η ασφάλεια των δεδομένων καθώς και η συνέπεια και η συνάφεια τους.

Βιβλιογραφία

[1] Sage A. Weil Andrew W. Leung Scott A. Brandt Carlos Maltzahn. RADOS: A Scalable, Reliable Storage Service for Petabyte-scale Storage Clusters, 2007.

[2] Sage A. Weil Scott A. Brandt Ethan L. Miller Darrell D. E. Long. Ceph: A Scalable, High-Performance Distributed File System, 2006.

Παράρτημα Α

Πίνακες αποτελεσμάτων μετρησεων sosd 1ου σεναρίου

Πίνακας Α.1: Μεσος χρόνος εξυπηρέτησης αίτησης εγγραφής

Object size \ Parallel IO	1	2	4	8	16
4K	7749	10617	18401	35709	72672
16K	7524	11243	19821	41457	83880
65K	8700	11951	24163	50466	102891
256K	14640	16876	31071	71654	146628
512K	22530	24678	42994	89224	180958
1M	38731	40883	63868	131551	269665
2M	69219	73198	99952	201220	406173

Πίνακας Α.2: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης

Parallel IO / Object size	1	2	4	8	16
4K	2295	3068	4782	8703	17063
16K	3357	5435	8502	16711	33581
65K	6381	8879	15819	31452	63653
256K	13080	17596	30607	62740	126019
512K	23659	31802	51736	99520	200681
1M	42504	54991	86638	168876	337489
2M	69752	85405	126774	242239	485694

Πίνακας Α.3: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης με χρήση cache στους OSDs

Parallel IO / Object size	1	2	4	8	16
4K	912	856	757	695	916
16K	1120	1065	956	1144	2266
65K	1624	1676	2249	4474	8941
256K	3774	4752	8919	17841	35677
512K	7120	12121	20797	38953	77417
1M	14365	26188	43678	82930	163643
2M	23097	43084	82332	156660	311853

Πίνακας Α.4: Συνολικός ρυθμός διαμεταγωγής για εγγραφές

Parallel IO / Object size	1	2	4	8	16
4K	513	750	866	892	877
16K	2121	2840	3223	3083	3047
65K	7321	10673	10569	10132	9940
256K	17348	30138	32828	28522	27892
512K	22541	41169	47400	45773	45179
1M	26199	49664	63766	62035	60574
2M	29316	55478	81405	81041	80346

Πίνακας Α.5: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις

Object size \ Parallel IO	1	2	4	8	16
4K	1735	2597	3332	3662	3736
16K	4760	5881	7518	7651	7615
65K	10024	14410	16176	16270	16077
256K	19567	29092	33448	32634	32479
512K	21638	32195	39578	41142	40779
1M	24089	37234	47264	48478	48468
2M	29360	47953	64600	67545	67282

Πίνακας Α.6: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις με χρήση cache στος OSDs

Object size \ Parallel IO	1	2	4	8	16
4K	4364	9295	21021	45788	69544
16K	14249	29966	66770	111621	112750
65K	39353	76277	113727	114346	114434
256K	67795	107688	114780	114745	114733
512K	71891	84445	98443	105065	105604
1M	71273	78196	93745	98692	99893
2M	88661	95058	99461	104399	104660

Παράρτημα Β

Πίνακες αποτελεσμάτων μετρησεων sosd 2ου σεναρίου

Πίνακας Β.1: Μεσος χρόνος εξυπηρέτησης αίτησης εγγραφής

Parallel IO / Object size	1	2	4	8	16
4K	13213	26198	53741	104573	208049
16K	12050	24087	48081	96275	192217
65K	12746	25947	52017	103522	207388
256K	15549	29880	60887	120997	241844
512K	23965	34689	69678	139212	269124
1M	40944	47173	89138	173962	337857
2M	71095	97154	129554	250759	489364

Πίνακας Β.2: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης

Object size \ Parallel IO	1	2	4	8	16
4K	1026	788	769	1454	3362
16K	1121	1234	1350	1946	4013
65K	1967	2446	4306	8202	16596
256K	6140	9077	17034	29287	47083
512K	10957	18646	34699	49607	88786
1M	22259	40093	57018	93909	172157
2M	47731	84815	110849	176108	325917

Πίνακας Β.3: Μεσος χρόνος εξυπηρέτησης αίτησης ανάγνωσης με χρήση cache στους OSDs

Object size \ Parallel IO	1	2	4	8	16
4K	1273	739	706	953	1915
16K	1053	1056	937	1198	3106
65K	1693	2503	2242	5225	8950
256K	3923	4626	8923	17836	35669
512K	6745	9425	17879	35661	71302
1M	12315	19440	35686	71270	142486
2M	23665	41485	71700	142662	284803

Πίνακας Β.4: Συνολικός ρυθμός διαμεταγωγής για εγγραφές

Object size \ Parallel IO	1	2	4	8	16
4K	301	304	296	304	306
16K	1325	1326	1329	1328	1330
65K	5005	4925	4916	4942	4934
256K	16328	17064	16786	16906	16921
512K	21203	29356	29309	29370	30378
1M	24777	43101	45765	46959	48374
2M	28596	41885	62888	65087	66753

Πίνακας Β.5: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις

Parallel IO / Object size	1	2	4	8	16
4K	3880	10103	20687	21917	18956
16K	14240	25886	47330	65682	62740
65K	32500	52279	59420	62381	61650
256K	41683	56370	60100	69871	86908
512K	46719	54906	59006	82537	92191
1M	45998	51074	71815	87166	95023
2M	42905	48285	73870	92933	100312

Πίνακας Β.6: Συνολικός ρυθμός διαμεταγωγής για αναγνώσεις με χρήση cache στος OSDs

Parallel IO / Object size	1	2	4	8	16
4K	3127	10769	22541	33394	33246
16K	15161	30248	68108	106631	82248
65K	37762	51091	114098	97828	114270
256K	65223	110640	114718	114759	114720
512K	75894	108621	114513	114799	114774
1M	83137	105329	114743	114867	114838
2M	86534	98710	114208	114745	114842

Παράρτημα C

Πίνακες αποτελεσμάτων μετρησεων xsegbd

Πίνακας C.1: Συνολικός ρυθμός διαμεταγωγής απευθείας στο xsegbd

Object size \ Action	Write	Read
4K	315	4296
16K	1464	16178
65K	5273	40820
256K	16015	65429
512K	17968	83040
1M	18164	94860
2M	18164	101236

Πίνακας C.2: Συνολικός ρυθμός διαμεταγωγής μέσα από VM

Object size \ Action	Write	Read
4K	357	3107
16K	1465	12695
65K	4883	31738
256K	15137	60547
512K	16016	78320
1M	15723	91895
2M	16016	97168