Αξιόπιστη Διαχείριση Ονοματοχώρου για Κατανεμημένα Συστήματα Αρχείων

Η ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης

του Τμήματος Πληροφορικής Εξεταστική Επιτροπή

από τον

Ευάγγελο Λάππα

ως μέρος των Υποχρεώσεων για τη λήψη του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ

ΣΤΑ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

Ιούνιος 2009

Αφιερωση

Στο Μορφέα...

Ευχαριστιες

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω όλους αυτούς που με χωρίς την συνεισφορά τους δεν θα είχε γίνει αυτή η εργασία.

Θα ήθελα να ευχαριστήσω τον επιβλέποντά μου, Λέκτορα κ. Στέργιο Αναστασιάδη, για την σημαντική καθοδήγησή του και τις πολύτιμες συμβουλές του καθ όλη τη διάρκεια τις εργασίας. Θα ήθελα ακόμη να ευχαριστήσω τα μέλη της ερευνητικής ομάδας συστημάτων για την άψογη συνεργασία που είχαμε σε όλο το διάστημα.

Επιπλέον θα ήθελα να ευχαριστήσω τον συμφοιτητή και συγκάτοικο μου Α. Βασιλάκη για την ειλικρινή υποστήριξή του.

Τέλος, πρέπει να σημειωθεί πως η δουλειά που παρουσιάζεται σε αυτή την εργασία υποστηρίχθηκε εν μέρει από το ερευνητικό πρόγραμμα Interreg IIA Greece-Italy 2000-2006 Grant No I2101005

Περιεχομενα

1 Εισαγωγή **1**
- 1.1 Στόχοι της Εργασίας . . . 1
- 1.2 Δομή της Εργασίας . . . 2

2 Σχετική Έρευνα **3**
- 2.1 Υπηρεσίες Χώρου Ονομάτων . . . 3
- 2.2 Ενδιάμεσο Λογισμικό για Υπηρεσίες Αρχείων . . . 5
- 2.3 Κατανεμημένα Συστήματα Αρχείων . . . 6
- 2.4 Συστήματα Αρχείων Ομότιμων Κόμβων . . . 9

3 Θεωρητικό Υπόβαθρο στο Paxos **10**
- 3.1 Μηχανή Καταστάσεων Πολλαπλών Αντιγράφων . . . 10
- 3.2 Ο Αλγόριθμος Paxos . . . 12
- 3.3 Περιγραφή του Αλγορίθμου . . . 12
- 3.4 Πρακτικά Ζητήματα . . . 13
- 3.5 Paxos για Σύνθεση Μελών . . . 14
- 3.6 Paxos για Πολλαπλά Αντίγραφα Βάσης Δεδομένων . . . 16

4 Αρχιτεκτονικός Σχεδιασμός **19**
- 4.1 Σχεδιαστικοί Στόχοι . . . 19
- 4.2 Μοντέλο Υλοποίησης . . . 20
- 4.3 Αρχιτεκτονική . . . 21
  - 4.3.1 Υπηρεσία Χώρου Ονομάτων . . . 22
  - 4.3.2 Διαγράμματα Καταστάσεων . . . 25
- 4.4 Περίληψη . . . 26

5 Υλοποίηση **27**
- 5.1 Περιβάλλον Υλοποίησης . . . 27
- 5.2 Πρόγραμμα Εξαγωγής Συστημάτων Αρχείων . . . 28
  - 5.2.1 Υλοποίηση Διακομιστή Ονομάτων . . . 28
  - 5.2.2 Υλοποίηση Πελάτη Ονομάτων . . . 28
- 5.3 Μηχανή Καταστάσεων Πολλαπλών Αντιγράφων . . . 29
  - 5.3.1 Επικεφαλίδα της ΜΚΠΑ . . . 30

5.3.2 Επικεφαλής Διακομιστής 30
5.3.3 Εφεδρικός Διακομιστής 35
5.3.4 Πελάτης ΜΚΠΑ υπηρεσίας 36
5.4 Πρωτόκολλο Paxos 37
5.4.1 Δημιουργία Αντικειμένου 37
5.4.2 Δεδομένα Αντικειμένου 37
5.4.3 Πρώτη Φάση 38
5.4.4 Δεύτερη Φάση 39
5.4.5 Τρίτη Φάση 41
5.4.6 Διαγράμματα Καταστάσεων 43
5.5 Περίληψη 45

6 Πειραματικά Αποτελέσματα **47**
6.1 Περιβάλλον Πειραμάτων 47
6.2 Τοπικοί Διακομιστές 48
6.2.1 Λειτουργία Χωρίς Αποτυχίες 48
6.2.2 Περιπτώσεις Αποτυχιών 51
6.3 Απομακρυσμένοι Διακομιστές 53

7 Επίλογος **57**
7.1 Επίλογος 57

Ευρετηριο Σχηματων

4.1 Παράδειγμα Συστήματος. Πίνακας εξαγωγής στους διακομιστές της υπηρεσίας χώρου ονομάτων 20
4.2 Παράδειγμα Συστήματος. Πίνακας εξαγωγής σε διακομιστή αρχείων 20
4.3 Αρχιτεκτονική συστήματος Orion 21
4.4 Ροή μηνυμάτων κατά τη σωστή λειτουργία του συστήματος 24
4.5 Διάγραμμα Καταστάσεων του Επικεφαλής Διακομιστή 25

5.1 Διάγραμμα Καταστάσεων του Αποστολέα μηνυμάτων του Paxos 44
5.2 Διάγραμμα Καταστάσεων του Αποδέκτη μηνυμάτων του Paxos 45

6.1 Ρυθμαπόδοση Συστήματος 49
6.2 Ρυθμαπόδοση Συστήματος με αποθήκευση σε διάνυσμα των παλιών αιτήσεων 49
6.3 Χρόνος απόκρισης Συστήματος 50
6.4 Χρόνος Εκκίνησης Διακομιστή 51
6.5 Αποτυχία Εφεδρικού Διακομιστή 52
6.6 Αποτυχία Επικεφαλής Διακομιστή 52
6.7 Μέσος χρόνος εξυπηρέτησης για διακομιστές που βρίσκονται μέσα σε μία χώρα σε περίπτωση χαμηλού και υψηλού φορτίου 54
6.8 Μέσος χρόνος εξυπηρέτησης για διακομιστές που βρίσκονται μέσα σε μία ήπειρο σε περίπτωση χαμηλού και υψηλού φορτίου 55
6.9 Περιπτώσεις αποτυχιών στην περίπτωση που οι διακομιστές βρίσκονται μέσα σε μία χώρα 55

Ευρετηριο Πινακων

5.1 Δομές Δεδομένων αντικειμένου ΜΚΠΑ μαζί με μία σύντομη περιγραφή για κάθε πεδίο 30
5.2 Συναρτήσεις Αντικειμένου ΜΚΠΑ μαζί με μία σύντομη περιγραφή για κάθε πεδίο 31
5.3 Πιθανές απαντήσεις από τον διακομιστή με αντίστοιχη περιγραφή και ενέργεια του πελάτη. 36
5.4 Μεταβλητές του Paxos και επεξήγηση αυτών 38

Ευρετηριων Αλγοριθμων

5.1 Μηνύματα heartbeat 32
5.2 Λειτουργίες του Επικεφαλής αφού λάβει ένα αίτημα 33
5.3 Πρώτη Φάση του Πρωτοκόλλου Paxos 38
5.4 Δεύτερη Φάση του Πρωτοκόλλου Paxos 40
5.5 Τρίτη Φάση του Πρωτοκόλλου Paxos 42

Περιληψη

Ευάγγελος Λάππας, MSc,
Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούνιος 2009.
Αξιόπιστη διαχείριση ονοματοχώρου για κατανεμημένα συστήματα αρχείων.

Επιβλέπων: Στέργιος Αναστασιάδης

Τα κατανεμημένα συστήματα αρχείων παρέχουν ένα εύχρηστο και ευέλικτο περιβάλλον κοινοχρησίας δεδομένων στους χρήστες του ίδιου οργανισμού. Παραμένει όμως ανοιχτό το ζήτημα της ενοποίησης του ονοματοχώρου πολλαπλών συστημάτων που υποστηρίζονται από διαφορετικούς διακομιστές, έτσι ώστε το μονοπάτι των αρχείων να μην εξαρτάται από το διακομιστή που τα εξυπηρετεί. Στην παρούσα εργασία εξετάζουμε την αντιμετώπιση του ζητήματος στα υπάρχοντα πρωτόκολλα κατανεμημένων συστημάτων αρχείων και προτείνουμε έναν μηχανισμό ενοποίησης που βασίζεται στην πίνακα εξαγωγής και στην υπηρεσία ανακατεύθυνσης αιτήσεων του συστήματος Network File System (version 4). Στη συνέχεια σχεδιάζουμε και υλοποιούμε μία αξιόπιστη υπηρεσία διαχείρισης πολλαπλών αντιγράφων βασισμένη στον αλγόριθμο Paxos του Lamport και αξιολογούμε πειραματικά την απόδοσή της σε διάφορες συνθήκες λειτουργίας.

Extended Abstract in English

Evangelos Lappas, MSc,
Computer Science Department, University of Ioannina, June 2009.
Thesis Title: Reliable management of distributed file systems namespace.

Supervisor: Stergios Anastasiadis

Distributed file systems provide an easy-to-use and flexible environment for sharing data among the users of the same organization. Federated file systems are a subcategory of distributed file systems that provide data sharing among the users of different organizations. However, it remains open issue the problem of unifying the namespace of multiple systems supported by different servers. Our goal is to have the path of each file be independent of the server location that stores the file.

In the present thesis we study the above problem in the context of existing distributed file systems protocols. We propose a namespace unification mechanism that is based on the export table and the referral service of Network File System (version 4). Furthermore, we design the system architecture, where we define the required functionality across the different parts of the system. The entities communicate via a protocol that we introduce for that purpose.

The namespace service is a decentralized service that is used to synchronize the storage servers in order to provide the global namespace. Each read request to the namespace service returns the table of the exported filesystems. This information is adapted to form the export table of each NFS server that participates in the federation. The unified namespace forms the global namespace of the federated file system.

Our system achieves fault-tolerance through replication of the namespace service across multiple servers. We use the Paxos algorithm of Lamport to maintain the group-membership state of the servers that participate in the service. Additionally, we synchronously replicate the export table across all the namespace servers. Overall, we achieve consensus of the replicas independently of server failures or network problems. Finally, we evaluate experimentally our system's performance across different operation conditions.

Κεφαλαιο 1

Εισαγωγη

1.1 Στόχοι της Εργασίας

1.2 Δομή της Εργασίας

1.1 Στόχοι της Εργασίας

Υπάρχει μία τάση στην έρευνα και στις επιχειρήσεις σήμερα για συνεργασία διαφορετικών οργανισμών μέσω δίκτυων ευρείας περιοχής. Η συνεργασία εξαρτάται από την κοινοχρησία πόρων όπως είναι η επεξεργαστική ισχύς και ο διαμοιρασμός δεδομένων. Οι λόγοι συνεργασίας είναι οι μεγάλες απαιτήσεις για επεξεργαστική ισχύ ώστε να ανταποκρίνονται στην ανάγκη εκτέλεσης χρονοβόρων υπολογισμών. Όσον αφορά τα δεδομένα, η ανάγκη για συνεργασία είναι ακόμη μεγαλύτερη αν όχι επιτακτική, εφόσον είναι το μέσο επικοινωνίας μεταξύ διαφορετικών ομάδων.

Υποθέτουμε έναν ερευνητικό οργανισμό, ο οποίος κάνει πειραματικές μελέτες και χρειάζεται υπολογιστική ισχύ ώστε να έχει όσο το δυνατόν γρηγορότερα αποτελέσματα. Όμως για να διεξάγει τα πειράματα και να προχωρήσει με την έρευνα του πρέπει να έχει πρόσβαση στα δεδομένα, που ενδεχομένως παράγονται από έναν άλλο ερευνητικό οργανισμό. Με τα υπάρχοντα συστήματα οι διαχειριστές των δύο οργανισμών πρέπει να επικοινωνήσουν ώστε να διαθέσει ό ένας στον άλλο τα δεδομένα. Ακόμη μεγαλύτερο πρόβλημα προκύπτει όταν χρειαστεί να συμφωνήσουν για το ποιοι χρήστες θα έχουν πρόσβαση πάνω στα δεδομένα και με τι πολιτικές. Επιπλέον, ένας οργανισμός μπορεί να χρειάζεται δεδομένα από διάφορους άλλους οργανισμούς. Οπότε η παραπάνω διαδικασία πρέπει να γίνει για κάθε οργανισμό ξεχωριστά.

Την λύση στα παραπάνω προβλήματα φέρουν τα ομοσπονδιακά συστήματα αρχείων. Ένα ομοσπονδιακό σύστημα αρχείων καθιστά εύκολο το διαμοιρασμό μεταξύ διαφορετικών οργανισμών χωρίς να υπάρχει ανάγκη συμφωνίας των μελών μεταξύ τους. Ο κάθε

οργανισμός συνεισφέρει τα δικά του δεδομένα και είναι σε θέση να προσπελάσει δεδομένα των υπόλοιπων οργανισμών.

Ομοσπονδιακό σύστημα αρχείων είναι ένα κατανεμημένο σύστημα υπό την έννοια ότι δεν υπάρχει ένας μοναδικός διακομιστής αποθήκευσης αρχείων και επιτυγχάνει ανοχή σε σφάλματα μέσω της διατήρησης πολλαπλών αντιγράφων. Η προσπέλαση των αρχείων γίνεται μέσω ενός καθολικά κοινού χώρου ονομάτων. Ο κοινός χώρος ονομάτων είναι ορατός σε όλους τους πελάτες αρχείων ανεξάρτητα από τον οργανισμό στον οποίο βρίσκονται.

Για να συντηρηθεί η παραπάνω ο χώρος ονομάτων χρειάζεται μία υπηρεσία που να οργανώνει τους διακομιστές αρχείων από διάφορους οργανισμούς. Κάθε διακομιστής προσφέρει τα δικά του συστήματα αρχείων, έτσι πρέπει να γίνεται αντιστοίχιση ενός συστήματος αρχείων στον κατάλληλο διακομιστή. Αντίστοιχα, η υπηρεσία χώρου ονομάτων πρέπει να είναι ανεκτική σε σφάλματα και διαθέσιμη ανά πάσα στιγμή ώστε να εγγυάται τη σωστή λειτουργία του ομοσπονδιακού συστήματος αρχείων.

Στη συγκεκριμένη εργασία περιγράφεται ένας αξιόπιστος μηχανισμός ενοποίησης ονοματοχώρου με βάση των πίνακα εξαγωγής συστημάτων αρχείων. Ο μηχανισμός επιτυγχάνει ανοχή σε σφάλματα διατηρώντας πολλαπλά αντίγραφα του παραπάνω πίνακα. Για να επιτευχθεί συμφωνία μεταξύ των διαφορετικών αντιγράφων χρησιμοποιείται ο αλγόριθμος συμφωνίας Paxos. Επειδή ο αλγόριθμος είναι χρονοβόρος για λειτουργίες κατά τη σωστή εκτέλεση του συστήματος χρησιμοποιείται για τη διατήρηση της σύνθεση των μελών που κρατάνε αντίγραφα του μηχανισμού. Για την ενημέρωση των αντιγράφων στην περίπτωση απουσίας σφαλμάτων χρησιμοποιείται ένας μηχανισμός παρόμοιος του πρωτοκόλλου ολοκλήρωσης δύο φάσεων [8].

Στα πλαίσια της εργασίας πραγματοποιήθηκε μία πρότυπη υλοποίηση του αλγορίθμου συμφωνίας Paxos, ώστε να είναι εγγυημένη η συμφωνία για τη σύνθεση των μελών και υλοποιήθηκε μία μηχανή καταστάσεων πολλαπλών αντιγράφων για την ενημέρωση των αντιγράφων. Επιπλέον αναπτύχθηκε η αρχιτεκτονική του μηχανισμού ενοποίησης του χώρου ονομάτων με χρήση των δύο προαναφερθέντων μηχανισμών και μελετήθηκαν πειραματικά οι επιδόσεις σε διάφορες συνθήκες, ώστε να εκτιμηθεί το κόστος χρήσης του σε συστήματα ομοσπονδιακών αρχείων.

1.2 Δομή της Εργασίας

Η δομή της εργασίας είναι οργανωμένη ως εξής. Στο Κεφάλαιο 2 περιγράφονται Σχετικές Εργασίες σε κατανεμημένα συστήματα αρχείων. Στο Κεφάλαιο 3 αναλύεται η θεωρητική μελέτη του αλγορίθμου συμφωνίας Paxos, πρακτικά ζητήματα και συστήματα που τον χρησιμοποιούν για να πετύχουν συμφωνία. Το Κεφάλαιο 4 περιγράφεται η αρχιτεκτονική του ομοσπονδιακού συστήματος αρχείων Orion και η λειτουργία των δομικών του συστατικών. Στο Κεφάλαιο 5 παρουσιάζεται η υλοποίηση του συστήματος Orion. Στο Κεφάλαιο 6 γίνεται η πειραματική μελέτη της υλοποίησης και τέλος στο Κεφάλαιο 7 γίνεται μία σύνοψη της εργασίας και αναγνωρίζονται προβλήματα για μελλοντική δουλειά.

Κεφαλαιο 2

Σχετικη Ερευνα

2.1 Υπηρεσίες Χώρου Ονομάτων

2.2 Ενδιάμεσο Λογισμικό για Υπηρεσίες Αρχείων

2.3 Κατανεμημένα Συστήματα Αρχείων

2.4 Συστήματα Αρχείων Ομότιμων Κόμβων

2.1 Υπηρεσίες Χώρου Ονομάτων

Η υπηρεσία χώρου ονομάτων Spring παρέχει μια ομοιογενή υπηρεσία χώρου ονομάτων, όπου οποιοσδήποτε τύπος αντικειμένου μπορεί να αντιστοιχηθεί σε οποιοδήποτε όνομα ανεξάρτητα από τη θέση του στο δίκτυο [12]. Ένα *περιβάλλον (context)* είναι ένα αντικείμενο το οποίο περιλαμβάνει συσχετίσεις από όνομα σε αντικείμενο ή συνδέσεις ονομάτων. Ένα αντικείμενο μπορεί να αντιστοιχηθεί σε αρκετά διαφορετικά ονόματα από διαφορετικά *περιβάλλοντα (contexts)* την ίδια στιγμή. Μία *λειτουργία* σε ένα περιβάλλον είναι η αντιστοίχηση του ονόματος, κατά την οποία ανακτούμε το αντικείμενο που δηλώνει το όνομα. *Σύνδεση* του ονόματος σε ένα περιβάλλον είναι μία λειτουργία συσχέτισης ενός ονόματος με ένα συγκεκριμένο αντικείμενο. Η υπηρεσία χώρου ονομάτων υλοποιείται από έναν ή περισσότερους διαχειριστές αντικειμένων, οι οποίοι ονομάζονται εξυπηρετητές ονομάτων και υλοποιούν το αντικείμενο περιβάλλοντος. Δοθέντος του περιβάλλοντος, το όνομα ενός αντικειμένου μπορεί να είναι σύνθετο ή απλό, ανάλογα αν αποτελείται από μία σειρά από συστατικά ή ακριβώς από ένα συστατικό αντίστοιχα.

Η διαδικασία της αντιστοίχησης του ονόματος ξεκινά με την παρουσίαση του ονόματος στην υπηρεσία χώρου ονομάτων διαμέσου ενός αντικειμένου περιβάλλοντος. Αν το όνομα είναι απλό, τότε το περιβάλλον εκτελεί την λειτουργία που ζητήθηκε. Αν το όνομα όμως είναι σύνθετο, το πρώτο περιβάλλον ανακτά το αντικείμενο που αναφέρεται ως πρώτο συστατικό, στη συνέχεια προωθεί τη λειτουργία σε αυτό το περιβάλλον με το υπόλοιπο μέρος

του ονόματος ως παράμετρο. Η αίτηση τελειώνει με μία λειτουργία στο διαχειριστή αντικειμένων. Η διάρκεια της κατάστασης των αντικειμένων αναφέρεται στους διαχειριστές αντικειμένων που αποθηκεύουν την κατάσταση των αντικειμένων για επόμενες αιτήσεις. Η διάρκεια της πρόσβασης του αντικειμένου αναφέρεται στην ύπαρξη ενός χαμηλού επιπέδου αναγνωριστικού, που ταυτοποιεί λογικά το αντικείμενο και απαιτεί μετάφραση στη φυσική τοποθεσία του αντικειμένου κατά τη διάρκεια κλήσης του αντικειμένου. Η μονιμότητα στη σύνδεση ονόματος είναι η προσέγγιση που ακολουθείται από το Spring . Συνδέει ένα αντικείμενο με ένα όνομα σε ένα σταθερό εξυπηρετητή ονομάτων και στη συνέχεια αντιστοιχίζει το όνομα για την πρόσβαση του ακριβούς αντικειμένου. Το αντικείμενο που επιστρέφεται από τον εξυπηρετητή ονομάτων θα περιέχει μία αναπαράσταση κατάλληλη για γρήγορες κλήσεις και για αποδοτικό πέρασμα του αντικειμένου ως παράμετρος.

Οι Anderson et al προτείνουν μια καθολική υπηρεσία χώρου ονομάτων (Global Namespace Service - GNS) με τα αρχεία ως μηχανισμό ονομάτων που συνδέει μεταξύ τους υπάρχουσες πηγές δεδομένων [2]. Θεμελιώδες τμήμα της υπηρεσίας είναι ο διαχωρισμός του ονόματος των δεδομένων από την τοποθεσία τους, ο οποίος επιτρέπει η απεικόνιση μεταξύ τους να γίνει δυναμικά. Η καθολική φύση της υπηρεσίας χρειάζεται τα λογικά ονόματα να είναι ομοιογενή στις περιοχές διαχείρισης για να υποστηρίζουν εικονικούς οργανισμούς και άλλες ομάδες που δεν είναι πλήρως ορισμένες . Ο χώρος ονομάτων είναι ιεραρχικά δομημένος με μονοπάτια ονομάτων, των οποίων τα προθέματα δίνουν από ονόματα , συναθροίσεις δεδομένων συστήματος αρχείων.

Συνολικά, ο καθολικός χώρος ονομάτων μπορεί να διαιρεθεί σε τρία τμήματα: τον αρχικό κατάλογο (root), το GNS και τα φυσικά συστήματα αρχείων. Τα ονόματα στον αρχικό κατάλογο δηλώνουν την προέλευση του ελέγχου στους διάφορους κατόχους χώρου ονομάτων. Το GNS δηλώνει καθολικά προθέματα αρχείων και υποδέντρων. Τα συστήματα αρχείων εξάγουν αρχεία τερματικού, υποδέντρα και συστήματα αρχείων. Ο χώρος ονομάτων αποτελείται από εικονικούς καταλόγους που περιέχουν άλλους εικονικούς καταλόγους και συνενώσεις (junctions) . Η συνένωση είναι ένα αντικείμενο, το οποίο δείχνει σε αρχεία και υποδέντρα που παρέχονται από τις πηγές δεδομένων ή από τους εξυπηρετητές χώρου ονομάτων. Οι πελάτες προσπελαύνουν τα μονοπάτια ονομάτων ανατρέχοντας μία σειρά από στιγμιότυπα GNS και αποκτούν αναφορές σε αρχεία ή συστήματα αρχείων. Αυτή η αναφορά είναι γενικά ένα λογικό όνομα, το οποίο το απεικονίζει μία υπηρεσία τοποθεσίας σε μία αναφορά σε φυσικό εξυπηρετητή αρχείων.

Γενικεύοντας, ο στόχος που δείχνει η συνένωσης μπορεί να είναι ένα άλλο GNS στιγμιότυπο, ένα λογικό όνομα αρχείου, ένα φυσικό όνομα αρχείου, ένα λογικό όνομα συστήματος αρχείων ή ένα φυσικό όνομα συστήματος αρχείων. Ένα στιγμιότυπο GNS αποθηκεύει ένα δέντρο από εικονικούς καταλόγους, ξεκινώντας από τον αρχικό κατάλογο, καθώς μία συνένωση αναπαριστά μία εξουσιοδότηση(delegation) μίας αρχής για την εύρεση επόμενων συστατικών ονομάτων μονοπατιών. Δέντρα συστήματος αρχείων είναι τερματικά του χώρου ονομάτων, γιατί τα περισσότερα συστήματα αρχείων δεν περιέχουν συνενώσεις. Παρόλα αυτά, είναι δυνατόν να εισαχθεί μία συνένωση σε ένα σύστημα αρχείων για να επιτρέπεται η εμφώλευση δέντρων συστήματος αρχείων μέσα σε άλλα συστήματα αρχείων.

2.2 Ενδιάμεσο Λογισμικό για Υπηρεσίες Αρχείων

Το Glamour είναι ένα πακέτο που προσφέρει την ομοσπονδία συστημάτων αρχείων ευρείας περιοχής. Ο στόχος του συστήματος είναι να παρέχει κατανεμημένη πρόσβαση αρχείων μέσω δίκτυα κλίμακας διαδικτύου, δίκτυα που επιδεικνύουν περιορισμένο εύρος ζώνης, υψηλή χρόνο απόκρισης και χαμηλή αξιοπιστία. Εισάγει το *σύνολο αρχείων (fileset)* ως αφαιρετική έννοια αποθήκευσης μεταξύ ενός συστήματος αρχείων και ενός καταλόγου. Το σύνολο αρχείων κατέχει το δικό του περιεχομένου και μπορεί να μετακινηθεί από ένα σύστημα αρχείων σε άλλο σχετικά εύκολα. Ένα σύνολο αρχείων υλοποιείται μέσω ενός ζεύγους αντικειμένων, του αντικειμένου συνόλου αρχείων(fileset object) και αντικειμένου θέσης συνόλου αρχείων(fileset location object), τα οποία είναι παρόμοια με το μοντέλο του συνόλου αρχείων και του τόμου (volume) στο AFS. Στο Glamour, οι πληροφορίες του συνόλων αρχείων ενσωματώνονται στο χειριστή αρχείου (filehandle) που ανταλλάσσεται μεταξύ του πελάτη και του διακομιστή, κατά προτίμηση από του να αποθηκεύεται στο αντικείμενο στον διακομιστή.

Το Glamour υλοποιεί ένα πρωτόκολλο μεταξύ διακομιστών αντίστοιχο του προτύπου LDAP. Χρησιμοποιεί επίσης μια υπηρεσία διαχείρισης δεδομένων για να διατηρήσει την διαμορφωμένη κατάσταση των συνόλων αρχείων και των μεταφορών συνόλων αρχείων μεταξύ των διακομιστών. Το Glamour διευκολύνει την οργάνωση συνόλων αρχείων με την παροχή ενός αρχικού χώρου ονομάτων που όλοι οι πελάτες βλέπουν από κοινού. Δεν περιέχει κανένα δεδομένο και χρησιμεύει μόνο ως μια θέση για να συνδεθούν(mount) άλλα σύνολα αρχείων δημιουργημένα από το χρήστη. Όλα τα αντικείμενα του Glamour, όπως τα σύνολα αρχείων, οι θέσεις συνόλων αρχείων, και ο αρχικός κατάλογος χώρου ονομάτων συνδέονται με ένα *κύτταρο(cell)*. Τα κύτταρα είναι ανεξάρτητα και δεν αλληλεπιδρούν το ένα με το άλλο. Είναι λογικά κατασκευάσματα, και πολλαπλά από αυτά μπορεί να εξυπηρετηθούν από έναν διακομιστή.

Το σύστημα Storage Resource Broker(SRB) συνδυάζει πολλαπλούς διακομιστές αποθήκευσης με μια αποκεντρικοποιημένη βάση δεδομένων σε μια υπηρεσία ανακάλυψης δεδομένων (data discovery) και μία μονάδα διαμοιρασμού που έχει κατασκευαστεί πάνω σε ένα δίκτυο ευρείας περιοχής [4]. Ο κατάλογος μεταδεδομένων (Metadata CATalog - MCAT) είναι μια μονάδα διαχείρισης πληροφοριών που χρησιμοποιείται για τη διαχείριση των μεταδεδομένων στο SRB. Τόσο το μοντέλο δεδομένων που χρησιμοποιούνται από το MCAT για την εσωτερική δόμηση των μεταδεδομένων, όσο και το σχήμα ανταλλαγής που χρησιμοποιείται για την επικοινωνία μεταδεδομένων με τον εξωτερικό κόσμο περιγράφονται από την προδιαγραφή δομών παρουσίασης ιδιοτήτων μεταδεδομένων (Metadata Attribute Presentation Structure - MAPS).

Το ενδιάμεσο λογισμικό SRB παρέχει ενοποιημένη πρόσβαση στους πολλαπλούς κατανεμημένου διακομιστές αποθήκευσης που συνδέονται με μία μέθοδο ομότιμων κόμβων. Τα μεταδεδομένα του SRB και οι διακομιστές αποθήκευσης χωρίζονται σε ζώνες, με κάθε ζώνη διαχειριζόμενη από ένα απλό MCAT. Κάθε MCAT είναι προσβάσιμο από έναν άλλο ενεργοποιημένο για MCAT διακομιστή αποθήκευσης και είναι σε θέση να αποθηκεύσει σε κρυφή μνήμη(cache) τα μεταδεδομένα από άλλους MCAT. Ένας πελάτης συνδέεται σε

έναν διακομιστή αποθήκευσης μέσω του οποίου έχει πρόσβαση στα δεδομένα ολόκληρης της SRB εγκατάστασης. Ένας πελάτης μπορεί να υποβάλει τις ερωτήσεις για την εύρεση των συνόλων δεδομένων(datasets) που ικανοποιούν συγκεκριμένα κριτήρια αναζήτησης και αιτήματα μεταφοράς δεδομένων για πρόσβαση στα σύνολα δεδομένων που ανακαλύφθηκαν. Κάθε διακομιστής αποθήκευσης επιλύει τα αιτήματα πρόσβασης με την υποβολή ερωτήσεων στο MCAT της ζώνης του και επικοινωνώντας άμεσα με το SRB διακομιστή όπου τα ζητούμενα στοιχεία βρίσκονται.

Το πλέγμα δεδομένων Avaki είναι ιδιόκτητη τεχνολογία που προσφέρει ομοσπονδιακές υπηρεσίες δεδομένων δια μέσων πολλαπλών οργανισμών, αλλά έχουμε μόνο περιορισμένη γνώση για την εσωτερική αρχιτεκτονική της [3]. Ο κατάλογος του δεδομένων πλέγματος παρέχει άμεση πρόσβαση στα υποκρυπτόμενα αντικείμενα αρχείων στις τοποθεσίες προέλευσής τους. Η πιστοποίηση των χρηστών και των ομάδων μπορεί να γίνει μέσω των υπηρεσιών καταλόγου που υπάρχουν ήδη σε κάθε μεμονωμένη τοποθεσία. Τα δεδομένα και τα μεταδεδομένα αρχείων αποθηκεύονται σε κρυφή μνήμη τοπικά για να επιτευχθούν βελτιωμένοι χρόνοι απόκρισης κατά την πρόσβαση. Οι διαδικασίες αναζήτησης αντικειμένου υποστηρίζονται βασιζόμενες στις ιδιότητες που διατηρούνται από τα υποκρυπτόμενα συστήματα αρχείων, ή τις προτιμώμενες ιδιότητες των χρηστών που έχουν δοθεί σε μεμονωμένα αντικείμενα.

2.3 Κατανεμημένα Συστήματα Αρχείων

Ο Pawlowski et al, προσδιόρισαν ότι οι απαιτήσεις του NFS της έκδοσης 4 περιλαμβάνουν βελτιωμένη πρόσβαση και απόδοση μέσω του διαδικτύου, ισχυρή ασφάλεια, ενισχυμένη λειτουργία μεταξύ διαφορετικών πλατφορμών και τη δυνατότητα επέκτασης. Σύμφωνα με την ορολογία του NFS, καθορίζουν ένα *σύστημα αρχείων* ως μία υλοποίηση ενός ενιαίου χώρου ονομάτων αρχείων που περιέχει αρχεία, το οποίο παρέχει τη βάση για τη διαχείριση και τη δέσμευση χώρου. Σε κάθε σύστημα αρχείων αποδίδεται ένα μοναδικό προσδιοριστικό των 128-bit ανά διακομιστή, αποκαλούμενο *προσδιοριστικό συστήματος αρχείων - fsid.* Ένα *αρχείο* είναι ένα ονομαστικό αντικείμενο που αποτελείται από τα δεδομένα και τα χαρακτηριστικά που υπάρχουν σε ένα σύστημα αρχείων. Ένα *κανονικό (regular)* αρχείο είναι μία ροή από byte, αντίθετα από έναν κατάλογο, συμβολικό σύνδεση ή ένα ειδικό(special) αρχείο. Ένας *χειριστής αρχείου (filehandle)* προσδιορίζει μοναδικά ένα αντικείμενο σε έναν διακομιστή και το αντίστοιχο σύστημα αρχείων, ενώ παραμένει αδιαφανής στον πελάτη.

Στην έκδοση 4 του NFS, ο πελάτης έχει πρόσβαση στα εξαγόμενα συστήματα αρχείων του διακομιστή απλώς φορτώνοντας το χειριστή αρχείου για τον αρχικό κατάλογο του δέντρου συστημάτων αρχείων. Επομένως, ο διακομιστής παρουσιάζει μια ενιαία όψη όλων των εξαγόμενων συστημάτων αρχείων στον πελάτη, και αφαιρεί την απαίτηση των παλαιότερων εκδόσεων του NFS που έπρεπε ένας πελάτης να διασυνδέσει όλα τα διαφορετικά εξαγόμενα συστήματα αρχείων ενός διακομιστή. Όταν ένας διακομιστής επιλέγει να εξαγάγει ένα μη-συνδεδεμένο τμήμα του χώρου ονομάτων του, ο διακομιστής δημιουργεί ένα ψευδο-σύστημα αρχείων για να γεφυρώσει τα εξαγόμενα τμήματα και να επιτραπεί στον πελάτη για να έχει

πρόσβαση στα σημεία εξαγωγής από ένα κοινό αρχικό κατάλογο. Κατά συνέπεια, ένας ψευδο-σύστημα αρχείων είναι μια δομή που περιέχει μόνο τους καταλόγους που επιτρέπουν στον πελάτη να διατρέξει την ιεραρχία των εξαγόμενων συστημάτων αρχείων.

Το NFS είναι βασισμένο στις απομακρυσμένες κλήσεις διαδικασιών (Remote Procedure Call - RPC). Η έκδοση 4 εισάγει την σύνθετη εντολή *(compound)* που ομαδοποιεί πολλαπλές σχετικές λειτουργίες σε ένα ενιαίο πακέτο RPC. Η αντίστοιχη απάντηση περιέχει τις απαντήσεις σε όλες τις διαδικασίες. Ομοίως, υποστηρίζει μια πολλών συστατικών λειτουργία *αναζήτησης* που επιτρέπει σε έναν πελάτη για να αντιστοιχίσει ένα όνομα με πλήρη διαδρομή σε μια λειτουργία. Το σύστημα επιτρέπει σε έναν διακομιστεί να εξουσιοδοτήσει(delegate) συγκεκριμένες ενέργειες για ένα αρχείο σε έναν πελάτη για να επιτρέψει την δυναμική αποθήκευση σε κρυφή μνήμη(caching) των δεδομένων και την κατάσταση κλειδώματος. Η ανάκληση της εξουσιοδότησης γίνεται μέσω μιας απομακρυσμένης κλήσης διαδικασίας, της *ανάκληση (callback)*, από το διακομιστή στον πελάτη για να τον ενημερώσει λειτουργίες του διακομιστή. Σε διαδικασίες πάνω σε αντικειμένου, το πρωτόκολλο χρησιμοποιεί τα προσδιοριστικά ονόματα χρηστών και ομάδας της μορφής user@domain και group@domain, αντίστοιχα. Κατά συνέπεια, το σύστημα αποφεύγει τα προσδιοριστικά ακέραιων αριθμών που απαιτούν όλους τους συμμετέχοντες πελάτες και τους διακομιστές να συμφωνήσουν σχετικά με τις αναθέσεις χρηστών και ομάδων.

Οι Zhang και Honeyman εκμεταλλεύονται το ειδικό χαρακτηριστικό αρχείου *fs_locations* του NFSv4 για να υποστηρίξουν διατήρηση πολλαπλών αντιγράφων μόνο για ανάγνωση σε ένα δίκτυο ευρείας περιοχής [16]. Σύμφωνα με την αρχική προδιαγραφή του πρωτοκόλλου, η πρώτη προσπέλαση του πελάτη σε ένα σύστημα αρχείων πολλαπλών αντιγράφουν επιστρέφει μία λίστα με εναλλακτικές τοποθεσίες. Αντ' αυτού, οι συντάκτες προωθούν το αναφερόμενο όνομα σε ένα δαίμονα (daemon) που ρωτά έναν διακομιστή του συστήματος αντιστοίχησης ονομάτων περιοχών (Domain Name System - DNS) για να αντιστοιχίσει το όνομα σε μια ή περισσότερες τοποθεσίες διακομιστών αρχείων. Στη συνέχεια, ο πελάτης επιλέγει έναν διακομιστή αρχείων και τον διασυνδέει.

Διατηρούν τη συμβατική σημασιολογία *"close to open"* του NFS, η οποία εγγυάται ότι μια εφαρμογή που ανοίγει ένα αρχείο βλέπει τα δεδομένα που γράφονται μία εφαρμογή που γράφει και κλείνει το αρχείο. Επιπλέον, η προτεινόμενη επέκταση όσον αφορά την πολιτική πολλαπλών αντιγράφων οργανώνει ταυτόχρονες εγγραφές με την επιλογή ενός πρωτεύον διακομιστή σύμφωνα με τις απαιτήσεις του πελάτη. Όλοι οι άλλοι διακομιστές που αντιγράφουν την συγκεκριμένη πληροφορία καθοδηγούνται να προωθούν τα αιτήματα πελατών για εκείνο το αρχείο στον πρωτεύοντα διακομιστή. Ο πρωτεύον κατανέμει τις ανανεώσεις στους άλλους διακομιστές κατά τη διάρκεια της τροποποίησης αρχείων. Όταν ο τελευταίος που γράφει τελειώσει, ο πρωτεύον διακομιστής ειδοποιεί τους άλλους διακομιστές που αντιγράφουν την πληροφορία ότι δεν είναι πλέον ο πρωτεύον διακομιστής για το αρχείο. Ένας διακομιστής γίνεται πρωτεύον διακομιστής μόνο αφού μαζέψει τις επιβεβαιώσεις από μια πλειοψηφία των διακομιστών που αντιγράφουν την συγκεκριμένη πληροφορία.

Το Andrew (AFS) είναι ένα κλιμακώσιμο κατανεμημένο σύστημα αρχείων που επιτρέπει τη διανομή των δεδομένων μέσω ενός δικτύου ευρείας περιοχής [14]. Τα αρχεία οργανώ-

νονται στους τομείς (volumes) που μπορούν να αντιγραφούν σε πολλαπλές μηχανές, διαμορφώνοντας τις ομάδες αποθήκευσης τόμων. Τα αντίγραφα ενός αρχείου αποθηκεύονται τοπικά σε κρυφή μνήμη στις μηχανές των πελατών. Τα τοπικά αντίγραφα ακυρώνονται προτού ένα αρχείο να είναι σε θέση να τροποποιηθεί σε μια διαφορετική μηχανή. Ένα κοινό δέντρο καταλόγου παρέχει πρόσβαση στα αρχεία με έναν διαφανή ως προς την τοποθεσία τρόπο.

Ο *τόμος (volume)* είναι ένα κομμάτι της ιεραρχίας του συστήματος αρχείων το οποίο είναι μικρότερο από ένα διαμέρισμα(partition) του δίσκου. Μια βάση δεδομένων τοποθεσίας τόμων, οι οποίοι διατηρείται σε πολλαπλά αντίγραφα σε όλους τους διακομιστές, παρέχει τη φυσική τοποθεσία του τόμου, ο οποίος είναι ανεξάρτητος από την τοποθεσία του τόμου στην ιεραρχία του συστήματος αρχείων. Η διεπαφή μεταξύ του πελάτη και του διακομιστή χρησιμοποιεί χαμηλού επιπέδου προσδιοριστικά αρχείων αντί της ολόκληρης διαδρομής ονόματος αρχείου. Αυτή η απόφαση κάνει εφικτές τις λειτουργίες μετονομασίας και απλοποιεί τη διαχείριση κρυφής μνήμης χάρις στην καθορισμένου μήκους φύση των προσδιοριστικών αρχείων. Το προσδιοριστικό αρχείων χωρίζεται σε δύο μέρη, ένα αριθμό τόμου και ένα αριθμό αρχείου μέσα στον τόμο.

Το Farsite είναι ένα κατανεμημένο σύστημα αρχείων που παρέχει τη διαθεσιμότητα και την αξιοπιστία αρχείων μέσω της διατήρησης πολλαπλών αντιγράφων, ασφάλεια περιεχομένου αρχείων μέσω κρυπτογραφίας, και ακεραιότητα μέσω των ανθεκτικών σε σφάλματα πρωτοκόλλων [7]. Κάθε μηχανή λειτουργεί ως πελάτης που εκτελεί τις διαδικασίες για τους τοπικούς χρήστες και ως διακομιστής που παρέχει υπηρεσία καταλόγου στους πελάτες. Για να εκτελέσει μια λειτουργία, ένας πελάτης λαμβάνει ένα αντίγραφο των σχετικών μεταδεδομένων από τον διακομιστή μαζί με μια μίσθωση(lease) για τα μεταδεδομένα. Κατά τη διάρκεια ισχύος της μίσθωσης, ο πελάτης έχει μια έγκυρη τιμή για τα μεταδεδομένα. Εάν η μίσθωση είναι για εγγραφή, τότε ο διακομιστής προσωρινά παραιτείται από την εξουσία του στα μεταδεδομένα, επειδή η μίσθωση επιτρέπει στον πελάτη να τροποποιήσει τα μεταδεδομένα. Το σύστημα παρέχει μια κατανεμημένη υπηρεσία μεταδεδομένων που χωρίζει τα μεταδεδομένα από τα προσδιοριστικά αρχείων που είναι αμετάβλητα. Το προσδιοριστικό αρχείων είναι μια ακολουθία θετικών ακέραιων αριθμών. Τα προσδιοριστικά αρχείων δημιουργούν ένα δενδρικό χώρο, υποδένδρα του οποίου κατανέμονται στους διαφορετικούς διακομιστές.

Το Wheelfs είναι ένα κατανεμημένο σύστημα αρχείων που δίνει την δυνατότητα στο χρήστη να τροποποιήσει τις επιλογές αποθήκευσης και γενικά τα μεταδεδομένα με την χρήση των *σημασιολογικών συνθημάτων (semantic cues)* [15]. Με τον τρόπο αυτό μπορούν να αποδοθούν δυναμικά ιδιότητες σε υποδένδρα του συστήματος αρχείων. Τέτοιες ιδιότητες μπορεί να είναι από το πλήθος των αντιγράφων μέχρι και τον τρόπο με τον οποίο επιτυγχάνεται η συνέπεια των αντιγράφων. Τα σημασιολογικά συνθήματα περιγράφονται στο μονοπάτι του χώρου ονομάτων του συστήματος αρχείων, οπότε και προκύπτουν μονοπάτια της μορφής */wfs/.MaxTime=200/url*. Με αυτό τον τρόπο γίνεται δυναμικά κάθε αλλαγή αλλά δεν παρέχεται ενοποιημένη εικόνα για τον χώρο ονομάτων του συστήματος αρχείων.

Ο συγχρονισμός των διαφορετικών μελών γίνεται από την *υπηρεσία διαμόρφωσης (con-*

figuration service), που παρέχει αντιστοίχιση κάθε αντικειμένου σε ένα πρωτεύον διακομιστή αποθήκευσης. Οι πελάτες και οι διακομιστές κρατάνε τοπικά αντίγραφα για τις αντιστοιχίσεις που γνωρίζουν.

2.4 Συστήματα Αρχείων Ομότιμων Κόμβων

Αντ' αυτού, το Ivy είναι ένα πολλών χρηστών ανάγνωση/εγγραφής σύστημα αρχείων ομότιμων κόμβων που αποτελείται από ένα σύνολο σειριακών αρχείων καταγραφής [1]. Τα σειριακά αρχεία καταγραφής αποθηκεύονται σε ένα κατανεμημένο πίνακας κατακερματισμού. Η ανάκτηση δεδομένων και οι τροποποιήσεις διευθύνονται με την ανίχνευση και την προσθήκη εγγραφών στα σειριακό αρχεία καταγραφής, αντίστοιχα, και οι εφαρμογές έχουν πρόσβαση στο σύστημα μέσω μιας συμβατικής διεπαφής συστημάτων αρχείων.

Κεφαλαιο 3

Θεωρητικο Υποβαθρο στο Paxos

3.1 Μηχανή Καταστάσεων Πολλαπλών Αντιγράφων

3.2 Ο Αλγόριθμος Paxos

3.3 Περιγραφή του Αλγορίθμου

3.4 Πρακτικά Ζητήματα

3.5 Paxos για Σύνθεση Μελών

3.6 Paxos για Πολλαπλά Αντίγραφα Βάσης Δεδομένων

3.1 Μηχανή Καταστάσεων Πολλαπλών Αντιγράφων

Το κατανεμημένο λογισμικό είναι συχνά δομημένο σε πελάτες και υπηρεσίες [13]. Κάθε υπηρεσία περιλαμβάνει έναν ή περισσότερους διακομιστές και εξάγει λειτουργίες που οι πελάτες επικαλούνται με την υποβολή των αιτημάτων. Αν και χρησιμοποιώντας έναν μόνο, κεντρικοποιημένο, διακομιστή είναι ο απλούστερος τρόπος να υλοποιηθεί μια υπηρεσία, η προκύπτουσα υπηρεσία μπορεί μόνο να είναι μόνο ανεχτική σε σφάλματα όσο ο επεξεργαστής που εκτελεί τις λειτουργίες στο διακομιστή. Εάν αυτό το επίπεδο ανοχής σε σφάλματα δεν είναι αποδεκτό, τότε πολλαπλοί διακομιστές οι οποίοι αποτυγχάνουν ανεξάρτητα πρέπει να χρησιμοποιηθούν.

Η προσέγγιση μηχανών καταστάσεων είναι μια γενική μέθοδος για να υλοποιηθεί μια ανεκτική σε σφάλματα υπηρεσία με την αντιγραφή του διακομιστή και το συντονισμό των αλληλεπιδράσεων των πελατών με τα αντίγραφα των διακομιστών. Μια μηχανή καταστάσεων αποτελείται από τις *μεταβλητές καταστάσεων*, που κωδικοποιούν την *κατάστασή* της, και τις εντολές, οι οποίες αλλάζουν την κατάστασή της. Κάθε εντολή υλοποιείται από ένα ντετερμινιστικό πρόγραμμα. Ένας πελάτης της μηχανής καταστάσεων υποβάλλει ένα

αίτημα για να εκτελέσει μια εντολή. Το αίτημα διευκρινίζει μια μηχανή καταστάσεων, ονομάζει την εντολή που θα εκτελεστεί, και περιέχει οποιεσδήποτε πληροφορίες απαιτούνται από την εντολή.

Μια μηχανή καταστάσεων επεξεργάζεται τα αιτήματα ένα κάθε φορά, σε μια διάταξη που είναι συνεπής με την πιθανή αιτιότητα. Επομένως, τα αιτήματα που εκδίδονται από έναν απλό πελάτη σε μια δεδομένη μηχανή καταστάσεων SM υφίστανται επεξεργασία από την SM με τη σειρά που εκδόθηκαν. Εάν το γεγονός ότι το αίτημα R υποβλήθηκε σε μια μηχανή καταστάσεων SM από τον πελάτη C θα μπορούσε να έχει προκαλέσει ένα αίτημα R' να γίνει από ένα πελάτη C' στην SM, τότε η SM επεξεργάζεται την R πριν από την R'.

Ένα δομικό συστατικό θεωρείται ελαττωματικό μόλις η συμπεριφορά της δεν είναι πλέον σύμφωνη με τις προδιαγραφές. Σε *βυζαντινές αποτυχίες*, το συστατικό μπορεί να παρουσιάσει αυθαίρετη και κακόβουλη συμπεριφορά, πιθανώς συμπεριλαμβάνοντας τη συνέργεια με άλλα ελαττωματικά συστατικά. Σε *αποτυχίες διακοπής (fail-stop failures)* το συστατικό αλλάζει σε μία κατάσταση που επιτρέπει σε άλλα συστατικά να ανιχνεύσουν ότι μια αποτυχία έχει συμβεί και έπειτα σταματάει. Ένα σύστημα που αποτελείται από ένα σύνολο διακριτών συστατικών είναι *ανεκτικό σε t σφάλματα* εάν ικανοποιεί την παρεχόμενη προδιαγραφή πως όχι περισσότερα από t από εκείνα τα συστατικά δεν θα γίνουν ελαττωματικά κατά τη διάρκεια κάποιου διαστήματος ενδιαφέροντος.

Μια *ανεκτική σε t σφάλματα* έκδοση μιας μηχανής καταστάσεων μπορεί να υλοποιηθεί με την αντιγραφή της μηχανής καταστάσεων και τρέχοντας ένα αντίγραφο σε καθένα από τους επεξεργαστές ενός κατανεμημένου συστήματος. Δεδομένου ότι κάθε αντίγραφο που εκτελείται σε έναν μη- ελαττωματικό επεξεργαστή αρχίζει στην ίδια αρχική κατάσταση και εκτελεί τα ίδια αιτήματα με την ίδια διάταξη, τότε κάθε ένα θα κάνει το ίδιο πράγμα και θα παραγάγει την ίδια έξοδο.

Όταν οι επεξεργαστές μπορούν να συμπεριφερθούν σύμφωνα με τις βυζαντινές αποτυχίες, ένα σύνολο που υλοποιεί μιας ανεκτική σε t σφάλματα μηχανή καταστάσεων, πρέπει να έχει τουλάχιστον $2t+1$ τα αντίγραφα, και η έξοδος του συνόλου είναι η έξοδος που παράγεται από την πλειοψηφία των αντιγράφων. Ουσιαστικά με $2t+1$ αντίγραφα, η πλειοψηφία των εξόδων παραμένει σωστή ακόμα και μετά από t αποτυχίες. Εάν οι επεξεργαστές συμπεριφέρονται μόνο με αποτυχίες διακοπής, τότε ένα σύνολο που περιέχει $t+1$ τα αντίγραφα αρκεί, και η έξοδος του συνόλου είναι η έξοδος που παράγεται από οποιονδήποτε από τα μέλη της.

Η υλοποίηση μιας μηχανής καταστάσεων ανεκτικής σε t σφάλματα απαιτεί ότι όλα τα αντίγραφα λαμβάνουν και επεξεργάζονται την ίδια ακολουθία αιτημάτων (*συντονισμός αντιγράφων*). Αυτή η απαίτηση μπορεί να αποσυντεθεί σε δύο απαιτήσεις:

i Κάθε μη ελαττωματικό αντίγραφο της μηχανής καταστάσεων λαμβάνει κάθε αίτημα (*Συμφωνία-Consensus*),

ii Κάθε μη ελαττωματικό αντίγραφο της μηχανής καταστάσεων επεξεργάζεται τα αιτήματα που λαμβάνει στην ίδια σχετική διάταξη (*Διάταξη (Order)*).

Γενικά, η απαίτηση συμφωνίας μπορεί να ικανοποιήσει με τη χρήση οποιουδήποτε πρω-

τοκόλλου που επιτρέπει σε οποιοδήποτε καθορισμένο επεξεργαστή (*αναμεταδότης*) να διαδώσει μια τιμή σε μερικούς άλλους επεξεργαστές με τέτοιο τρόπο ώστε όλοι οι ελαττωματικοί επεξεργαστές να συμφωνήσουν στην ίδια τιμή. Εάν ο αναμεταδότης είναι μη-ελαττωματικός, τότε όλοι οι μη-ελαττωματικοί επεξεργαστές χρησιμοποιούν την τιμή του ως αυτήν στην οποία συμφωνούν. Η απαίτηση συμφωνίας μπορεί να χαλαρωθεί για τα αιτήματα ανάγνωσης μόνο(read only) όταν υποθέτουμε επεξεργαστές αποτυχίας διακοπής. Όταν οι επεξεργαστές είναι αποτυχίας διακοπής, ένα αίτημα r η του οποίου η επεξεργασία δεν τροποποιεί τις μεταβλητές της κατάστασης χρειάζεται μόνο να σταλεί σε ένα απλό μή-ελαττωματκό αντίγραφο της μηχανής καταστάσεων.

Η απαίτηση διάταξης μπορεί να ικανοποιηθεί με την ανάθεση μοναδικών προσδιοριστικών στα αιτήματα και έχοντας τη μηχανή καταστάσεων πολλαπλών αντιγράφων να επεξεργάζεται τα αιτήματα σύμφωνα με μία συνολική σχέση διάταξης σε αυτά τα μοναδικά προσδιοριστικά. Η πιθανή υλοποίηση της διαταγής στηρίζεται (i) στα λογικά ρολόγια, (ii) στα συγχρονισμένα σε πραγματικό χρόνο ρολόγια, ή (iii) προσδιοριστικά παραγόμενα από αντίγραφα (replica-generated).

3.2 Ο Αλγόριθμος Paxos

Υποθέτοντας μία συλλογή από διεργασίες που μπορεί να προτείνουν διάφορες τιμές, ένας αλγόριθμος συμφωνίας (consensus) εγγυάται πως μία μόνο τιμή επιλέγεται από τις προτεινόμενες [9, 10]. Αν δεν υπάρχει τιμή που να προτείνεται, τότε δεν πρέπει να επιλεγεί καμία τιμή. Αν έχει επιλεγεί μία τιμή, τότε οι διεργασίες πρέπει να είναι σε θέση να μάθουν την επιλεγμένη τιμή. Ο αλγόριθμος συμφωνίας εγγυάται πως:

- Μόνο μία τιμή που έχει προταθεί μπορεί να επιλεγεί
- Μόνο και μόνο μία τιμή επιλέγεται, και
- Μία διεργασία δεν μαθαίνει ποτέ πως μία τιμή έχει επιλεγεί εκτός και αν πραγματικά έχει επιλεγεί

3.3 Περιγραφή του Αλγορίθμου

Ο αλγόριθμος υποθέτει τρεις κατηγορίες πρακτόρων: *εισηγητές*,*αποδέκτες*, *μαθητευόμενοι*. Ένας εισηγητής στέλνει μια προτεινόμενη τιμή σε ένα σύνολο αποδεκτών. Ένας αποδέκτης μπορεί να αποδεχτεί την προτεινόμενη τιμή. Η τιμή επιλέγεται όταν ένα αρκετά μεγάλο σύνολο αποδεκτών την έχει αποδεχτεί. Για να μάθει ότι μια τιμή έχει επιλεχτεί, ένας μαθητευόμενος πρέπει να ανακαλύψει ότι μια πρόταση έχει γίνει αποδεκτή από μια πλειοψηφία των αποδεκτών. Οι πράκτορες επικοινωνούν με το ένα άλλος μέσω της ανταλλαγής μηνυμάτων. Οι πράκτορες λειτουργούν με αυθαίρετη ταχύτητα, μπορεί να αποτύχουν σταματώντας, και μπορεί να ξαναξεκινήσουν. Επίσης, τα μηνύματα μπορεί να πάρουν αυ-

θαίρετα πολύ χρόνο για να παραδοθούν, μπορεί να φτάσουν πολλές φορές, και μπορεί να χαθούν, αλλά δεν αλλοιώνονται.

Ο Αλγόριθμος λειτουργεί στις ακόλουθες δύο φάσεις

Φάση 1

1. Ο εισηγητής επιλέγει έναν αριθμό πρότασης n και στέλνει ένα αίτημα *προετοιμασίας(prepare)* με τον αριθμό n σε μια πλειοψηφία των αποδεκτών.
2. Εάν ένας αποδέκτης λαμβάνει ένα *prepare* αίτημα με τον αριθμό n μεγαλύτερο από οποιοδήποτε *prepare* αίτημα στο οποίο έχει απαντήσει ήδη, τότε απαντάει στο αίτημα με μια υπόσχεση να μην γίνουν αποδεκτές άλλες προτάσεις που έχουν αριθμό μικρότερο από n και με την υψηλότερα αριθμημένη πρόταση (αν υπάρχει) που έχει αποδεχτεί.

Φάση 2

1. Εάν ο εισηγητής λάβει μια απάντηση στα αιτήματα prepare (με αριθμό n) από μια πλειοψηφία των αποδεκτών, τότε στέλνει ένα αίτημα *αποδοχής(accept)* σε κάθε ένας από αυτούς τους αποδέκτες για μια πρόταση με αριθμό n και τιμή v, όπου v είναι η τιμή της υψηλότερα αριθμημένης πρότασης μεταξύ των απαντήσεων, ή είναι οποιαδήποτε τιμή εάν οι απαντήσεις δεν ανέφεραν καμία πρόταση.
2. Εάν ένας αποδέκτης λαμβάνει ένα *accept* αίτημα για μια πρόταση με αριθμό n, αποδέχεται την πρόταση εκτός αν έχει απαντήσει ήδη σε ένα *prepare* αίτημα που έχει έναν αριθμό μεγαλύτερο από n.

Προκειμένου να γίνει γνωστή η τιμή που έγινε αποδεκτή στους μαθητευόμενους, κάθε αποδέκτης όταν δέχεται μια πρόταση, στέλνει σε όλους τους μαθητευόμενους την πρόταση. Εναλλακτικά, οι αποδέκτες απαντάνε με τη τιμή που αποδέχτηκαν σε έναν συγκεκριμένο μαθητευόμενο, ο οποίος ενημερώνει στη συνέχεια τους άλλους μαθητευόμενος, όταν επιλεχτεί μια τιμή. Αυτή η προσέγγιση απαιτεί έναν πρόσθετο γύρω ώστε να ανακαλύψουν οι μαθητευόμενοι την επιλεγμένη τιμή. Είναι επίσης λιγότερο αξιόπιστο, δεδομένου ότι ο συγκεκριμένος μαθητευόμενος μπορεί να αποτύχει. Γενικότερα, οι αποδέκτες θα μπορούσαν να στείλουν τις τιμές που δέχτηκαν σε κάποιο σύνολο συγκεκριμένων αρχαρίων, καθένας από τους οποίος μπορεί έπειτα να ενημερώσει όλους τους μαθητευόμενους πότε μια τιμή έχει επιλεχτεί.

3.4 Πρακτικά Ζητήματα

Είναι εύκολο να κατασκευαστεί ένα σενάριο στο οποίο δύο εισηγητές συνεχώς εκδίδουν μια ακολουθία από προτάσεις με αυξανόμενους αριθμούς, κανένας από τους οποίους ποτέ δεν

επιλέγεται. Για να εγγυηθεί την πρόοδο, ένας συγκεκριμένος εισηγητής πρέπει να επιλεχτεί ως μοναδικός που να προσπαθήσει να εισηγείται προτάσεις. Εάν ο συγκεκριμένος εισηγητής μπορεί να επικοινωνήσει επιτυχώς με μια πλειοψηφία των αποδεκτών, και εάν χρησιμοποιήσει μια πρόταση με τον αριθμό μεγαλύτερο από οποιοιδήποτε ήδη χρησιμοποιηθεί, τότε θα πετύχει στην έκδοση μιας πρότασης που να γίνει αποδεκτή.

Ο αλγόριθμος επιλέγει έναν ηγέτη(leader), ο οποίος διαδραματίζει τους ρόλους του συγκεκριμένου εισηγητή και του συγκεκριμένου μαθητευόμενου(learner). Η σταθερή αποθήκευση, που συντηρείται κατά τη διάρκεια των αποτυχιών, χρησιμοποιείται για να διατηρηθούν οι πληροφορίες που ο αποδέκτης πρέπει να θυμάται. Ένας αποδέκτης καταγράφει την προοριζόμενη απάντησή του σε σταθερή αποθήκευση πριν πραγματικά να στείλει την απάντηση. Ένας απλός τρόπος να υλοποιηθεί ένα κατανεμημένο σύστημα είναι ως μία συλλογή από πελάτες που στέλνουν εντολές σε έναν κεντρικό διακομιστή. Ο διακομιστής μπορεί να περιγραφεί ως μια ντετερμινιστική μηχανή καταστάσεων που εκτελεί τις εντολές των πελατών σε κάποια ακολουθία. Μια υλοποίηση που χρησιμοποιεί έναν απλό κεντρικό διακομιστή αποτυγχάνει εάν αυτός ο κεντρικός διακομιστής αποτυγχάνει. Επομένως, είναι προτιμότερο να χρησιμοποιήσει μια συλλογή από διακομιστές, όπου κάθε μία υλοποιεί μία μηχανή καταστάσεων.

Για να εγγυηθεί ότι όλοι οι διακομιστές εκτελούν την ίδια ακολουθία εντολών μηχανών καταστάσεων, ο Lamport προτείνει να υλοποιηθεί μια ακολουθία διαφορετικών στιγμιοτύπων του αλγορίθμου συμφωνίας Paxos, η τιμή που επιλέγεται από το $i - στο$ στιγμιότυπο είναι η $i - στη$ εντολή της μηχανής καταστάσεων στην ακολουθία. Σε κανονική λειτουργία, ένας μόνο διακομιστής εκλέγεται για να είναι ο ηγέτης(leader), ο οποίος ενεργεί ως ο συγκεκριμένος εισηγητής σε όλες οι περιπτώσεις του αλγορίθμου συναίνεσης. Οι πελάτες στέλνουν τις εντολές στον ηγέτη, ο οποίος αποφασίζει το σημείο στην ακολουθία όπου κάθε εντολή πρέπει να εμφανιστεί.

Δεδομένου ότι η αποτυχία του ηγέτη και η εκλογή ενός νέου πρέπει να είναι σπάνια γεγονότα, το πραγματικό κόστος εκτέλεσης μίας εντολή μηχανής καταστάσεων είναι το κόστος της εκτέλεσης μόνο της φάσης 2 του αλγορίθμου συμφωνίας. Κατά συνέπεια, υπάρχει πάντα ένας μόνο ηγέτης εκτός από μια μικρή περίοδο ανάμεσα στην αποτυχία του τρέχοντος ηγέτη και την εκλογήν ενός νέου. Εάν το σύνολο διακομιστών μπορεί να αλλάξει, τότε πρέπει να υπάρχει κάποιος τρόπος να αποφασιστεί ποιοι διακομιστές υλοποιούν ποιο στιγμιότυπου του αλγορίθμου συμφωνίας. Ο ευκολότερος τρόπος να γίνει αυτό είναι μέσω της ίδιας της μηχανής καταστάσεων. Το τρέχον σύνολο διακομιστών μπορεί να γίνει μέρος της κατάστασης και μπορεί να αλλάξει με συνηθισμένες εντολές μηχανών καταστάσεων.

3.5 Paxos για Σύνθεση Μελών

Ο Mazieres προσδιορίζει τρεις φάσεις στην εκτέλεση Paxos [11]. Στην πρώτη φάση, ένας εισηγητής επιλέγει έναν αριθμό πρότασης και τον στέλνει σε όλους μέσω ενός prepare μηνύματος. Ένα μέλος της ομάδας απορρίπτει το μήνυμα εάν έχει δει ήδη prepare μήνυμα με μεγαλύτερο αριθμό πρότασης. Διαφορετικά, το μέλος δέχεται το μήνυμα και απαντά

με την τιμή που αντιστοιχεί στην υψηλότερα αριθμημένη πρόταση που έχει δει. Εάν το prepare μήνυμα γίνει αποδεκτό από μια πλειοψηφία των μελών ομάδας, ο εισηγητής στέλνει σε όλους ένα μήνυμα πρότασης (propose) με τον αρχικό αριθμό πρότασης και την τιμή της υψηλότερα αριθμημένης απάντησης που έλαβε ο εισηγητής. Ένα μέλος απορρίπτει το propose μήνυμα, αν έχει δει ένα prepare μήνυμα με υψηλότερο αριθμό πρότασης. Εάν μια πλειοψηφία της ομάδας αποδεχτεί το propose μήνυμα, τότε ο εισηγητής στέλνει σε όλους μήνυμα απόφασης(decide) ώστε να δείξει ότι η ομάδα έχει συμφωνήσει στον προτεινόμενο αριθμό και τιμή.

Ένα μέλος της ομάδας υποδεικνύεται ως *πρωτεύον(master)* και οι εναπομείναντες ως *εφεδρικοί (backups)*. Ο συντάκτης χρησιμοποιεί τον όρο *όψη(view)* για να δείξει ένα σύνολο από μέλη (*cohorts*) μίας ομάδας με έναν επιλεγμένο ως πρωτεύον. Υπάρχει ένας μοναδικός *αριθμός όψης (view-id)* για κάθε όψη. Ο πελάτης μαθαίνει τον επικεφαλής διακομιστή μίας ομάδας και αριθμό όψης μέσω μιας εξωτερικής υπηρεσίας καταλόγου. Όταν το σύστημα εκτελεί ένα αίτημα, τέσσερις τύποι μηνυμάτων πρέπει να σταλούν.

1. Ο πελάτης στέλνει το αίτημα του στον πρωτεύον
2. Ο πρωτεύον cohort καταγράφει το αίτημα και το προωθεί σε όλους τους cohorts
3. Οι cohorts καταγράφουν τη λειτουργία και στέλνουν μια επιβεβαίωση πίσω στον πρωτεύοντα
4. Μόλις ο πρωτεύον ξέρει ότι μια πλειοψηφία cohorts (συμπεριλαμβανομένου του ίδιου) έχει καταγράψει τη λειτουργία, εκτελεί τη λειτουργία και στέλνει το αποτέλεσμα στον πελάτη.

Ο πρωτεύον αριθμεί όλα τα αιτήματα τα οποία λαμβάνει κατά μια δεδομένη όψη με τις διαδοχικές τιμές της *χρονοσφραγίδας (timestamp)*,να ξεκινούν από το 1. Η *σφραγίδα όψεως (viewstamp)* που δημιουργείται από το συνδυασμό του view-id και του timestamp καθορίζει τη διάταξη εκτέλεσης όλων των αιτημάτων καθ όλη τη διάρκεια. Ένας cohort καταγράφει διαρκώς το αποτέλεσμα της τελευταίας λειτουργίας που έχει εκτελέσει για κάθε πελάτη, σε περίπτωση που η απάντηση του πρωτεύοντα στον πελάτη χαθεί και ο πρωτεύον αποτύχει στη συνέχεια. Επιβεβαιωμένα(commited) είναι εκείνα τα αιτήματα που εκτελέστηκαν στους διακομιστές και στάλθηκαν πίσω στους πελάτες. Οι καταχωρήσεις στο σειριακό αρχείο καταγραφής επιβεβαιωμένων αιτημάτων μπορούν να αφαιρεθούν από τους εφεδρικούς cohorts, αν και η παρουσία τους διευκολύνει τον συγχρονισμό των καταστάσεων με τις ομάδες που έχασαν μερικές λειτουργίες.

Η σύνθεση των μελών της ομάδας αλλάζει όταν ένας cohort συνδεθεί, ή ένας cohort υποψιάζεται πως έχει αποτύχει επειδή δεν απαντά στα μηνύματα από άλλους cohort. Οποιοδήποτε μέλος της ομάδας μπορεί να αποφασίσει να αλλάξει τη σύνθεσης ομάδας με μια αλλαγή όψης. Για να αρχίζει μια αλλαγή όψης, ένας cohort αρχίζει με την πρόταση μίας νέας ταυτότητας όψης (view-id). Ο cohort που προτείνει τη νέα ταυτότητα όψης καλείται *διαχειριστής* για αυτήν την νέα όψης. Πρώτα επιλέγει μια ταυτότητα όψης που είναι μεγαλύτερη από τη μεγαλύτερη την οποία έχει δει ο διαχειριστή. Ο αλγόριθμος Paxos ξεκινάει

μία φορά για κάθε όψη για να συμφωνήσει η σύνθεση της επόμενης όψης και να εξασφαλίσει ότι το πολύ μια τέτοια σύνθεση μπορεί να προχωρήσει. Οι cohorts που λαμβάνουν τα αιτήματα αλλαγής όψης από έναν διαχειριστή όψης καλούνται *υφιστάμενος (underlings)*. Ο υφιστάμενος που λαμβάνει ένα αίτημα αλλαγής όψης εξετάζει τέσσερις περιπτώσεις.

1. Ο cohort απορρίπτει το αίτημα επειδή είτε τουλάχιστον μια επόμενη όψη ήδη επιτυχώς έχει συμφωνηθεί μετά από αυτήν που ο διαχειριστής θέλει να αλλάξει, ή ένας άλλος διαχειριστής έχει προτείνει ήδη μια υψηλότερη νέα ταυτότητα όψης.

2. Εάν η προτεινόμενη ταυτότητα όψης είναι η μεγαλύτερη που ο υφιστάμενος έχει δει μέχρι τώρα, ο cohort αποδέχεται το αίτημα. Εάν ο cohort έχει συμφωνήσει ήδη σε μια σύνθεση που είναι νεώτερη από την τρέχουσα όψη του διαχειριστή, τότε ειδοποιεί το διαχειριστή για την προηγούμενη σύνθεση που η ομάδα έχει συμφωνήσει.

Ένας διαχειριστή όψης μπορεί να διαμορφώσει μια νέα όψη, εάν τουλάχιστον ένας cohort στη νέα όψη ξέρει όλες τις προηγούμενες επιβεβαιωμένες λειτουργίες. Αυτό είναι δυνατό εάν είτε η νέα όψη μοιράζεται μια πλειοψηφία από cohort με την παλαιά όψη, ή οι δύο όψεις έχουν τον ίδιο πρωτεύον. Επομένως, ο διαχειριστής όψης στρατολογεί τους cohorts για τη νέα όψη, αρχίζοντας από εκείνους τους cohorts που δέχτηκαν το αίτημα και συμφώνησαν να συμμετέχουν κατά τη νέα όψη. Επιπλέον, ο διαχειριστής της όψης προσθέτει στη νέα όψη τον πρωτεύοντα της παλαιάς όψης, εάν δέχτηκε το αίτημα αλλαγής όψης, αλλά δεν συμφώνησε απαραιτήτως να συμμετέχει στη νέα όψη. Εάν ο παλαιός πρωτεύον δεν δέχτηκε το αίτημα αλλαγής όψης, ο διαχειριστής όψης προσθέτει τον απαραίτητο αριθμό από cohorts από εκείνους που δέχτηκαν την αλλαγή όψης. Ο διαχειριστής όψης εξετάζει αυτούς τους cohorts σε φθίνουσα σειρά με βάση τη σφραγίδα όψης (viewstamp) από το τελευταίο τους αίτημα που έχουν αποδεχτεί πριν από την αλλαγή άποψης. Τέλος, ο διαχειριστής όψης επιλέγει τον πρωτεύοντα cohort να είναι, είτε ο πρωτεύον της παλαιάς όψης, είτε εναλλακτικά τον cohort με την υψηλότερη σφραγίδα όψης στα αιτήματα που έγινε αποδεκτά πριν από την αλλαγή άποψης.

3.6 Paxos για Πολλαπλά Αντίγραφα Βάσης Δεδομένων

Οι Chandra et al, χρησιμοποιούν το Paxos για να υλοποιηθεί μια ανεκτική σε σφάλματα βάση δεδομένων διατήρηση πολλών αντιγράφων που αποτελείται από ένα στιγμιότυπο βάσης δεδομένων και ένα σειριακό αρχείο καταγραφή για επανάληψη (replay-log) των λειτουργιών της βάσης δεδομένων [5]. Οι νέες λειτουργίες της βάσης δεδομένων υποβάλλονται στο σειριακό αρχείων καταγραφής πολλαπλών αντιγράφων (replicated log). Όταν μια λειτουργία βάσης δεδομένων εμφανίζεται σε ένα αντίγραφο, εφαρμόζεται στο τοπικό αντίγραφο της βάσης δεδομένων. Το Paxos χρησιμοποιείται ως αλγόριθμος συμφωνίας που εκτελείται από ένα σύνολο αντιγράφων για να συμφωνήσει μια τιμή παρουσία αποτυχιών. Ο αλγόριθμος αποτελείται από τρεις φάσεις:

1. Εκλέγει ένα αντίγραφο για να είναι ο συντονιστής. Όταν ένα αντίγραφο θέλει να γίνει συντονιστής, παράγει έναν μοναδικό αριθμό ακολουθίας υψηλότερο από οποιοιδήποτε έχει δει, και στέλνει σε όλα τα αντίγραφα σε ένα propose μήνυμα. Εάν μια πλειοψηφία των αντιγράφων απαντήσουν και αναφέρουν ότι δεν έχουν δει έναν υψηλότερο αριθμό ακολουθίας, τότε το αντίγραφο λειτουργεί ως συντονιστής. Αυτές οι απαντήσεις καλούνται μηνύματα υπόσχεσης, δεδομένου ότι τα αντίγραφα υπόσχονται εφεξής να απορρίψουν μηνύματα propose από τους παλαιούς συντονιστές.

2. Ο συντονιστής επιλέγει μια τιμή και τη στέλνει σε όλα τα αντίγραφα σε ένα μήνυμα αποκαλούμενο accept μήνυμα. Τα άλλα αντίγραφα είτε επιβεβαιώνουν αυτό το μήνυμα είτε το απορρίπτουν.

3. Μόλις μια πλειοψηφία των αντιγράφων επιβεβαιώσει το συντονιστή, η συμφωνία έχει επιτευχθεί, και ο συντονιστής στέλνει ένα επιβεβαίωσης (commit) μήνυμα για να ειδοποιήσει τα αντίγραφα.

Σε περίπτωση που ο συντονιστής αποτύχει, πολλά αντίγραφα μπορεί να αποφασίσουν να γίνουν συντονιστές και να εκτελέσουν τον αλγόριθμο οποιαδήποτε στιγμή. Η συμφωνία επιτυγχάνεται μόλις λάβει μια πλειοψηφία των αντιγράφων μήνυμα αποδοχής (accept) από το συντονιστή και το επιβεβαιώσει. Κατόπιν, το Paxos πρέπει να αναγκάσει τους μελλοντικούς συντονιστές να επιλέξουν την ίδια τιμή προκειμένου να εξασφαλιστεί συνεχιζόμενη συμφωνία. Ο νέος συντονιστής απαιτεί μια απάντηση στο μήνυμα propose από μια πλειοψηφία των αντιγράφων. Επομένως, εάν η συμφωνία επιτεύχθηκε από έναν προηγούμενο συντονιστή, ο νέος συντονιστής είναι εγγυημένο ότι θα ακούσει, για την τιμή που αποφασίστηκε πιο πριν, από τουλάχιστον ένα αντίγραφο. Επαγωγικά, η τιμή αυτή θα έχει τον υψηλότερο αριθμό ακολουθίας από το σύνολο των λαμβανόμενων απαντήσεων, και έτσι θα επιλεγεί από το νέο συντονιστή.

Τα πραγματικά συστήματα χρησιμοποιούν το Paxos ως δομική μονάδα για να επιτύχουν τη συμφωνία σε μια ακολουθία τιμών, όπως σε ένα σειριακό αρχείο καταγραφής πολλαπλών αντιγράφων(replicated log). Ο απλός τρόπος να υλοποιηθεί αυτό είναι να εκτελεσθεί επαναληπτικά ο αλγόριθμος Paxos, όπου κάθε εκτέλεση καλείται στιγμιότυπου του Paxos. Κάθε αντίγραφο διατηρεί ένα τοπικά μόνιμο σειριακό αρχείο καταγραφής(persistent log) για να καταγράψει όλες τις ενέργειες Paxos. Όταν ένα αντίγραφο αποτύχει και στη συνέχεια επανέλθει, επαναλαμβάνει το μόνιμο σειριακό αρχείο καταγραφής για να ανακατασκευάσει την κατάστασή του πριν από την αποτυχία. Τα αντίγραφα χρησιμοποιούν επίσης αυτό το αρχείο όταν χρειαστεί να βοηθήσει άλλα αντίγραφα που έχουν μείνει πίσω, ώστε να φτάσουν και αυτά στις τελευταίες αλλαγές.

Τα μηνύματα propose μπορεί να παραλειφθούν εάν η ταυτότητα των συντονιστών δεν αλλάζει μεταξύ των διαφορετικών στιγμιοτύπων. Μπορούμε να σχεδιάσουμε το Paxos να επιλέγει έναν συντονιστή για μεγάλες χρονικές περιόδους, προσπαθώντας να μην αφήσουμε να συμβούν αλλαγές συντονιστών. Αυτός ο συντονιστής καλείται επικεφαλής. Αυτό δεν παρεμποδίζει τις ιδιότητες του Paxos επειδή οποιοδήποτε αντίγραφο μπορεί οποιαδήποτε

στιγμή να προσπαθήσει να γίνει συντονιστής στέλνοντας ένα μήνυμα propose με έναν υψηλότερο αριθμό ακολουθίας.

Η χρήση του Paxos για να υλοποιηθεί μια δομή δεδομένων πολλαπλών αντιγράφων κάνει απαραίτητη την εκτέλεση ενός στιγμιότυπου του Paxos για κάθε λειτουργία ανάγνωσης(read). Αυτό σειριοποιεί τις λειτουργίες ανάγνωσης κατ' αναλογία των λειτουργιών ανανέωσης(update) και εξασφαλίζει ότι η ανάγνωση επιστρέφει την τρέχουσα κατάσταση. Κατά συνέπεια, η ανάγνωση πρέπει να συμβουλευθεί μια πλειοψηφία των αντιγράφων παρά να επιστρέψει μία τιμή από το αντίγραφο του επικεφαλής. Εναλλακτικά, προκειμένου να εξασφαλιστεί ότι ο επικεφαλής έχει ενημερωμένη δομή δεδομένων και εξυπηρετεί τοπικά τα αιτήματα διαβάσματος, ο επικεφαλής κρατά μια μίσθωση(lease) που εγγυάται ότι άλλα αντίγραφα δεν μπορούν να υποβάλουν τις τιμές στο Paxos. Ο επικεφαλής ανανεώνει το lease του πριν από τη λήξη για να το διατηρήσει για μια μεγάλη περίοδο.

Είναι δυνατό ένας επικεφαλής να χάσει την θέση του ως επικεφαλής στη χρονική διάρκεια από την λήψη ενός αιτήματος ως την πραγματική ενημέρωση της βάσης δεδομένων. Μία λύση για να ανιχνευθεί αυτό είναι με την χρήση ενός αριθμού εποχής (epoch number) που αυξάνεται κάθε φορά που αλλάζει ο επικεφαλής. Ο τρέχων αριθμός εποχής επισυνάπτεται σε όλες τις λειτουργίες της βάσης δεδομένων. Μια λειτουργία εγκαταλείπεται εάν ο αριθμός εποχής αλλάξει κατά τη διάρκεια της επεξεργασίας της λειτουργίας. Τέλος, προκειμένου να αποφευχθεί η επ'άπειρο συντήρηση ενός συνεχώς αυξανόμενου σειριακού αρχείου καταγραφής, είναι δυνατό να αποθηκευτεί μόνιμα ένα στιγμιότυπο της τρέχουσας κατάστασης. Κατόπιν, το Paxos διαγράφει αυτές τις καταχωρήσεις του σειριακού αρχείου καταγραφής οι οποίες προηγούνται του στιγμιότυπου. Τα στιγμιότυπα λαμβάνονται ανεξάρτητα στα αντίγραφα και δεν είναι απαραιτήτως συγχρονισμένα μεταξύ τους.

Κεφαλαιο 4

Αρχιτεκτονικος Σχεδιασμος

4.1 Σχεδιαστικοί Στόχοι

4.2 Μοντέλο Υλοποίησης

4.3 Αρχιτεκτονική

4.4 Περίληψη

Σε αυτό το κεφάλαιο περιγράφονται τις σχεδιαστικές αρχές τις οποίες πρέπει να έχει ένα ομοσπονδιακό σύστημα αρχείων. Σύμφωνα με τις αρχές αυτές περιγράφουμε το σύστημα Orion το οποίο βασίζεται σε ένα μηχανισμό ενοποίησης ονοματοχώρου με βάση των πίνακα εξαγωγής. Τέλος ορίζονται οι λειτουργίες των οντοτήτων στο σύστημα Orion και οι επικοινωνία μεταξύ των οντοτήτων.

4.1 Σχεδιαστικοί Στόχοι

Το ομοσπονδιακό σύστημα αρχείων είναι ένα κατανεμημένο σύστημα αρχείων στο οποίο συμμετέχει ανεξάρτητοι μεταξύ τους οργανισμοί. Κάθε οργανισμός προσφέρει τα δικά του συστήματα αρχείων μέσω των διακομιστών αρχείων. Κάθε διακομιστής αρχείων έχει ένα πίνακα εξαγωγής των συστημάτων αρχείων που προσφέρει.

Θεωρούμε ως βασικό χαρακτηριστικό του ομοσπονδιακού συστήματος αρχείων έναν μηχανισμό που να κάνει ορατή μια κοινή δομή καταλόγων προς όλους τους πελάτες. Ο μηχανισμός αυτός πρέπει να είναι διαθέσιμος ανά πάσα στιγμή και σε κάθε περίπτωση να μας δίνει την συνολική εικόνα του ομοσπονδιακού συστήματος αρχείων. Επίσης οι αλλαγές πρέπει να φαίνονται άμεσα στους κόμβους πελάτες.

Οι πελάτες εκτός από το να βλέπουν άμεσα τις αλλαγές πρέπει να είναι σε θέση να ανακαλύψουν αν υπάρχει δευτερεύων διακομιστής που να αποθηκεύει τα αρχεία, στην περίπτωση που ο πρώτος έχει τεθεί εκτός λειτουργίας ή έχει παρουσιαστεί κάποιο σφάλμα.

Ο μηχανισμός που κάνει ορατή την κοινή δομή καταλόγων πρέπει να αποθηκεύει όλα τα προσφερόμενα συστήματα αρχείων και τους διακομιστές που τα προσφέρουν. Η πληροφορία αυτή είναι παρόμοια με τη πληροφορία που αποθηκεύεται στο πίνακα εξαγωγής που υπάρχει για το σύστημα αρχείων NFSv4. Κάθε διακομιστής αρχείων προσαρμόζει την παραπάνω πληροφορία έτσι ώστε να έχουν όλοι το ίδιο χώρο ονομάτων.

Έστω πως υπάρχουν ένα σύνολο από τρεις διακομιστές που συνεισφέρουν τέσσερα συστήματα αρχείων, δηλαδή οι δύο από ένα και ο τρίτος συνεισφέρει δύο συστήματα αρχείων. Ο μηχανισμός που κρατάει αυτή την πληροφορία θα έχει τη μορφή που φαίνεται στο Σχήμα 4.1. Όπως είπαμε παραπάνω αυτή η πληροφορία πρέπει να προσαρμοστεί στον κάθε διακομιστή ξεχωριστά Αυτό φαίνεται στο Σχήμα 4.2.

```
node1     /physics  /export/physics     nohide,no_subtree_check
node2     /chemist  /export/chemist     nohide,no_subtree_check
node3     /math     /export/math        nohide,no_subtree_check
node3     /cs       /export/cs          nohide,no_subtree_check
```

Σχήμα 4.1: Παράδειγμα Συστήματος. Πίνακας εξαγωγής στους διακομιστές της υπηρεσίας χώρου ονομάτων

Αυτό που φαίνεται στο Σχήμα 4.2 είναι πως ο διακομιστής έχει ένα επιπλέον σύστημα αρχείων που συνεισφέρει. Αυτό είναι ο αρχικός κατάλογος, στον οποίο όμως δεν γράφει κανείς. Τα δύο επόμενα συστήματα αρχείων είναι αυτά τα οποία προσφέρει μέσω της επιλογής της ανακατεύθυνσης. Τα τελευταία δύο είναι τα τοπικά του συστήματα αρχείων. Για να υπάρχει ομοιομορφία θεωρούμε πως ο διαχειριστής του συστήματος έχει διασυνδέσει στο μονοπάτι */export/math* με επιλογή *–bind* το υποδένδρο που θέλει να διαμοιραστεί.

```
/export         *(fsid=0,insecure,no_subtree_check)
/export/physics *(nohide,refer=/@node1:/physics/,no_subtree_check)
/export/chemist *(nohide,refer=/@node2:/chemist/,no_subtree_check)
/export/math    *(rw,nohide,no_subtree_check)
/export/cs      *(rw,nohide,no_subtree_check)
```

Σχήμα 4.2: Παράδειγμα Συστήματος. Πίνακας εξαγωγής σε διακομιστή αρχείων

4.2 Μοντέλο Υλοποίησης

Στο σύστημα Orion οι επικοινωνία των διαφόρων οντοτήτων γίνεται με κλήση απομακρυσμένων ρουτινών (remote procedure call - rpc). Έτσι θεωρούμε πως κάθε μήνυμα μπορεί να χαθεί, να φτάσει παραπάνω από μία φορά ή να φτάσει κανονικά. Πάντως σε καμία περίπτωση δεν μπορεί να φτάσει αλλοιωμένο.

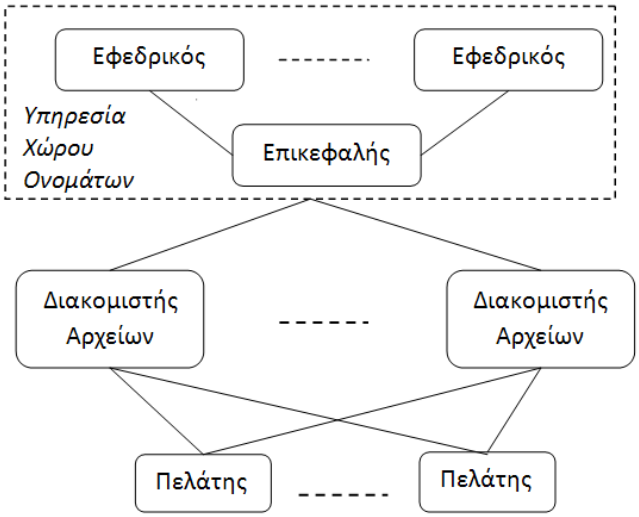
Κάθε διακομιστής ή πελάτης μπορεί να βρίσκεται οσοδήποτε μακρυά από το σύστημα αρκεί ο χρόνος που κάνει για να θεωρηθούν άκυρο τα μήνυμα να είναι αρκετός ώστε να

φτάσει το αίτημα στο σύστημα να γίνει οι κατάλληλες ενέργειες και να λάβει πίσω απάντηση. Αυτός ο χρόνος είναι μία παράμετρος του συστήματος και μπορεί να αλλαχθεί και να προσαρμοστεί στον κάθε πελάτη, χωρίς να επηρεάζει το υπόλοιπο σύστημα.

Οι διακομιστές μπορεί ανά πάσα στιγμή να αποτύχουν χωρίς να ολοκληρώσουν αυτό που εκτελούσαν. και χωρίς να επικοινωνήσουν με κάποιον ώστε να τον ειδοποιήσουν ότι τερματίζουν. Σε καμία περίπτωση δεν θεωρούμε πως κάποιος διακομιστής μπορεί να συμπεριφέρεται με βυζαντινή συμπεριφορά. Ένας διακομιστής όταν εμφανίζει βυζαντινή συμπεριφορά λειτουργεί πέρα του πρωτοκόλλου λειτουργίας και μπορεί να προσπαθεί να πείσει ότι δεν έχει αποτύχει. Συνήθως οι βυζαντινή συμπεριφορά είναι αποτέλεσμα κακόβουλων χρηστών που προσπαθούν να βλάψουν τη λειτουργία του συστήματος.

Το σύστημα Orion, ένα ομοσπονδιακό σύστημα αρχείων, δεν προϋποθέτει καμία ασφάλεια στην επικοινωνία των διακομιστών και των πελατών. Αρα θεωρούμε πως δεν έχουμε κακόβουλους κόμβους που προσπαθούν να βλάψουν το σύστημα.

4.3 Αρχιτεκτονική



Σχήμα 4.3: Αρχιτεκτονική συστήματος Orion

Το σύστημα Orion είναι ένα ομοσπονδιακό σύστημα αρχείων, το οποίο διασυνδέει ξεχωριστά συστήματα αρχείων που προσφέρονται από διαφορετικούς οργανισμούς. Στο Σχήμα 4.3 φαίνεται η αρχιτεκτονική του Συστήματος το οποίο αποτελείται από τριών ειδών οντότητες

1. Ένα σύνολο από διακομιστές οι οποίοι υλοποιούν την υπηρεσία χώρου ονομάτων

2. Οι διακομιστές αρχείων οι οποίοι είτε εξάγουν απευθείας στους πελάτες κάποιους καταλόγους του τοπικού τους συστήματος αρχείων, είτε κάνουν ορατούς στους πε-

λάτες καταλόγους από άλλους απομακρυσμένους διακομιστές μέσω της υπηρεσίας ανακατεύθυνσης.

3. Οι πελάτες διασυνδέονται με τα συστήματα αρχείων μέσω των διακομιστών

Η Υπηρεσία Χώρου Ονομάτων είναι η καρδιά όλου του συστήματος, γιατί συντονίζει τους διάφορους διακομιστές αρχείων. Για να το πετύχει αυτό προσφέρει ένα πίνακα εξαγωγής συστημάτων αρχείων στον οποίο αποθηκεύονται όλα τα συστήματα αρχείων που προσφέρονται από τους διακομιστές. Για κάθε ένα από αυτά κρατάει το μονοπάτι του ομοσπονδιακού συστήματος καθώς και τις επιλογές διασύνδεσης. Η συγκεκριμένη υπηρεσία κρατάει αντίγραφα ώστε να είναι διαθέσιμη ανά πάσα στιγμή.

Οι Διακομιστές αρχείων λαμβάνουν τον πίνακα εξαγωγής όπως τον προσφέρει η υπηρεσία χώρου ονομάτων και τον προσαρμόζουν στη δική τους περίπτωση. Κάθε σύστημα αρχείων είτε είναι τοπικό για τον εκάστοτε διακομιστή ή αποτελεί *παραπομπή(referral)* προς κάποιον άλλο διακομιστή αρχείων. Με αυτό τον τρόπο κάθε διακομιστή δίνει στους πελάτες του μία καθολική εικόνα του συστήματος. Διατηρώντας τα μονοπάτια και τις επιλογής διασύνδεσης ίδιες σε όλους του διακομιστές αρχείων, οι πελάτες αντιλαμβάνονται το ίδιο χώρο ονομάτων αρχείων ανεξαρτήτως από τον αρχικό διακομιστή στον οποίο συνδέονται.

Ο κάθε διακομιστής που συνδέεται στο σύστημα δεν είναι απαραίτητο να συνεισφέρει και ο ίδιος αποθηκευτικό χώρο στο ομοσπονδιακό σύστημα αρχείων. Μπορεί να λειτουργεί ως ενδιάμεσος για να προσφέρει συστήματα αρχείων στους πελάτες. Με αυτό τον τρόπο οι πελάτες δεν χρειάζεται να γνωρίζουν τίποτα για την υπηρεσία χώρου ονομάτων. Μία τέτοια επιλογή φαντάζει δυνατή στην περίπτωση που έχουμε μικρά υπολογιστικά κέντρα τα οποία απλώς θέλουν να έχουν πρόσβαση σε δεδομένων μεγαλύτερων οργανισμών. Απλώς με τη διασύνδεση του τοπικού διακομιστή αρχείων με το σύστημα Orion μπορούν να προσπελάσουν τα δεδομένα.

4.3.1 Υπηρεσία Χώρου Ονομάτων

Στην πιο απλή της μορφή, η υπηρεσία ονομάτων του συστήματος απαρτίζεται από έναν και μόνο διακομιστή, ο οποίος διαχειρίζεται τις αιτήσεις διαχείρισης του πίνακα εξαγωγής του συστήματος. Βέβαια οι αιτήσεις αυτές είναι κρίσιμες και στην περίπτωση αποτυχιών του διακομιστή το σύστημα παραμένει σε μία στάσιμη κατάσταση, όπου οι διακομιστές αρχείων δεν μπορούν να

- Ανανεώσουν τον τρέχοντα πίνακα εξαγωγής
- Εξάγουν επιπλέον συστήματα αρχείων

Κανονικά η υπηρεσία χώρου ονομάτων πρέπει να εγγυάται τη συνεχή λειτουργία πρόσβασης στον κοινό πίνακα εξαγωγής. Για το λόγο αυτό στο σχεδιασμό μας εισάγουμε μια ομάδα διακομιστών ονομάτων που κρατάει αντίγραφα από τα εξής

- Τον κοινό πίνακα εξαγωγής

- Τους ενεργούς διακομιστές της υπηρεσίας χώρου ονομάτων

Για να λειτουργεί σωστά η υπηρεσία ονομάτων, οι διακομιστές που την απαρτίζουν είναι οργανωμένοι σε δύο επίπεδα. Στο πρώτο επίπεδο βρίσκεται ο επικεφαλής που έχει την ευθύνη να σειριοποιεί τα αιτήματα των διακομιστών. Στο δεύτερο επίπεδο βρίσκονται οι εφεδρικοί διακομιστών ένας από τους οποίους αναπληρώνει τον επικεφαλής όταν ο τελευταίος εμφανίσει σφάλμα λειτουργίας. Έτσι οι διακομιστές αρχείων επικοινωνούν με τον επικεφαλής διακομιστή ονομάτων και αυτός προωθεί το αίτημα στους εφεδρικούς διακομιστές ονομάτων. Όταν όλοι οι ενεργοί εφεδρικοί απαντήσουν θετικά τότε ο επικεφαλής εκτελεί τοπικά το αίτημα και στέλνει τον ανανεωμένο πίνακα εξαγωγής σε όλους τους ενεργούς διακομιστές αρχείων.

Το πρωτόκολλο λειτουργίας της υπηρεσίας ονομάτων στην παραπάνω περιγραφή είναι παρόμοιο με το πρωτόκολλο ολοκλήρωσης δύο φάσεων. Το τελευταίο είναι ένας κατανεμημένος αλγόριθμος όπου οι διακομιστές είτε συμφωνούν να εκτελέσουν όλοι μία αίτηση, είτε αποτυγχάνει να εξυπηρετηθεί η αίτηση. Όπως λέει και το όνομά του λειτουργεί σε δύο φάσεις. Στην πρώτη στέλνει την αίτηση ώστε να σημειωθεί στο σειριακό αρχείο καταγραφής (log) και στη δεύτερη γίνεται η ίδια η αλλαγή. Το δικό μας σύστημα ενσωματώνει τις δύο φάσεις σε μία, όπου εκτελείται η αίτηση και στέλνεται πίσω η απάντηση.

Το πρωτόκολλο ολοκλήρωσης δύο φάσεων λειτουργεί σωστά όταν όλοι οι διακομιστές λειτουργούν σωστά. Στην περίπτωση αλλαγών της σύνθεση της ομάδας που διατηρούν τα αντίγραφα το συγκεκριμένο πρωτόκολλο δεν εγγυάται σωστή λειτουργία. Τέτοιες αλλαγές μπορεί να συμβούν είτε γιατί προστίθεται ένας νέος διακομιστής ονομάτων, είτε γιατί ανιχνεύθηκε αποτυχία σε έναν από τους υπάρχοντες. Ειδικά στην περίπτωση αποτυχίας του επικεφαλής δεν προβλέπεται αυτόματη διαδικασία ώστε αν αναλάβει κάποιος άλλος και οι υπόλοιποι να τον δεχτούν ως επικεφαλής.

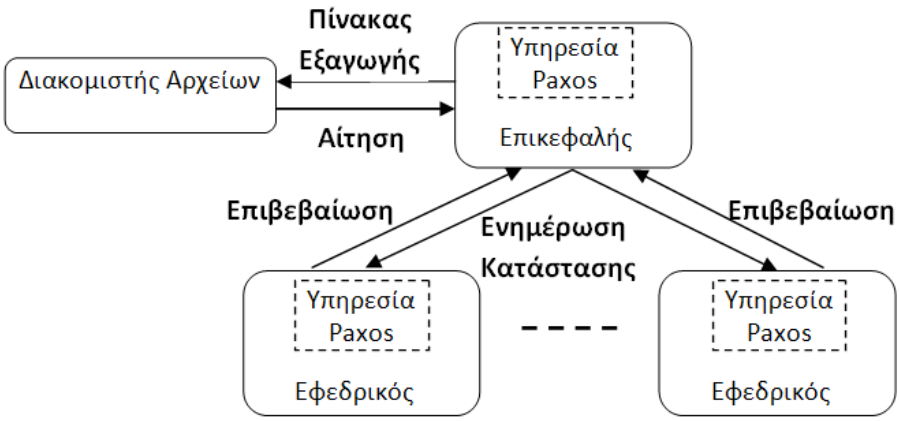
Στο σύστημα Orion για να καλυφθούν οι περιπτώσεις αποτυχιών χρησιμοποιείται το πρωτόκολλο Paxos. Κάθε φορά που ανιχνεύεται μία αλλαγή στην τρέχουσα σύνθεση των διακομιστών που κρατάνε αντίγραφα ξεκινάει ένα νέο στιγμιότυπου του πρωτοκόλλου. Στις επόμενες υποενότητες περιγράφουμε τις λειτουργίες του επικεφαλής και των εφεδρικών διακομιστών.

Επικεφαλής διακομιστής ονομάτων

Ο επικεφαλής διακομιστής ονομάτων είναι αυτός που λαμβάνει τα αιτήματα από τους διακομιστές αρχείων. Στο Σχήμα 4.4 φαίνεται η ροή μηνυμάτων στην περίπτωση σωστής λειτουργίας. Όταν λάβει ένα αίτημα εκτελεί τις επόμενες ενέργειες

1. τα προωθεί στους εφεδρικούς
2. αφού λάβει απάντηση από όλους το εκτελεί και ο ίδιος
3. στέλνει την απάντηση πίσω.

Στην περίπτωση που τουλάχιστον ένας από τους εφεδρικούς δεν απαντήσει τότε πρέπει να τον διαγράψει από τη σύνθεση και να ξεκινήσει ένα στιγμιότυπο του Paxos ώστε να



Σχήμα 4.4: Ροή μηνυμάτων κατά τη σωστή λειτουργία του συστήματος

συμφωνηθεί μία νέα σύνθεση. Όσο μία τέτοια διαδικασία είναι σε εξέλιξη το σύστημα δεν είναι δυνατόν να δεχθεί νέες αιτήσεις από τους διακομιστές αρχείων. Αν φτάσει όμως μία αίτηση επιστρέφεται πίσω η αντίστοιχη απάντηση για να προσπαθήσει αργότερα.

Ο επικεφαλής λαμβάνει περιοδικά μηνύματα ελέγχου από τους εφεδρικούς και τους διακομιστές αρχείων ώστε να ξέρει ποιοι λειτουργούν κανονικά και ποιοι όχι. Στην περίπτωση που σταματήσει να λειτουργεί ένας διακομιστής αρχείων, ο επικεφαλής τον διαγράφει και ενημερώνει τους εφεδρικούς για την αλλαγή. Στην περίπτωση που δεν λειτουργεί ένας εφεδρικός διακομιστής τότε ο επικεφαλής εκτελεί μια αλλαγή στη σύνθεσης της ομάδας υπηρεσίας χώρου ονομάτων σύμφωνα με το πρωτόκολλο Paxos.

Εφεδρικοί διακομιστές ονομάτων

Κάθε εφεδρικός διακομιστής είναι υπεύθυνος για δύο λειτουργίες.

- λαμβάνει αιτήματα από τον επικεφαλής, τα εκτελεί και απαντάει στον επικεφαλής θετικά αν εκτέλεσε το αίτημα
- στέλνει περιοδικά μηνύματα ελέγχου στον επικεφαλής

Τα μηνύματα ελέγχου επιβεβαιώνουν τη σωστή επικοινωνία των δύο άκρων. Στην περίπτωση που δεν απαντήσει ο επικεφαλής τότε ο εκάστοτε εφεδρικός που ανιχνεύει το πρόβλημα διαγράφει τον επικεφαλής από τη σύνθεση της ομάδας και εκτελεί νέα εκλογή αρχηγού σύμφωνα με το πρωτόκολλο Paxos.

Διακομιστής Αρχείων

Στο Σχήμα 4.3 φαίνεται πως οι διακομιστές αρχείων λειτουργούν ως πελάτες στην υπηρεσία χώρου ονομάτων. Από αυτή ζητάνε να μάθουν τον τρέχοντα πίνακα εξαγόμενων συστημάτων αρχείων και μπορούν να προσθέσουν ή να αφαιρέσουν ένα σύστημα αρχείων που προσφέρουν για κοινοχρησία.

Ο διακομιστής αρχείων συνδέεται με την υπηρεσία χώρου ονομάτων αφού γνωρίσει έναν τουλάχιστον από τους διακομιστές χώρου ονομάτων και στείλει αρχικό αίτημα για τη διασύνδεση. Επομένως αρχικά στέλνει ένα αίτημα ζητώντας τη λίστα με όλους τους διακομιστές ονομάτων του συστήματος. Ο διακομιστής αρχείων για να μάθει τον αρχικό κόμβο στον οποίο μπορεί να συνδεθεί χρησιμοποιεί το σύστημα DNS. Έτσι μπορεί να ρωτήσει και να πάρει πληροφορία για μία σειρά από διακομιστές χώρου ονομάτων.

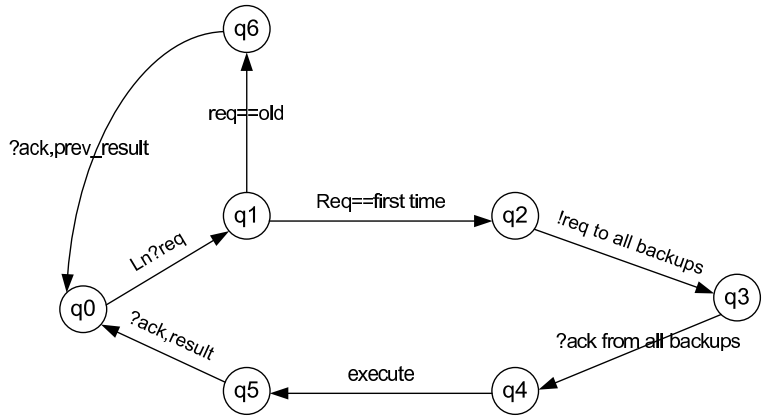
Ο διακομιστής χώρου ονομάτων μπορεί να είναι απασχολημένος με διαδικασίες εκλογής αρχηγού ή αλλαγής της σύνθεσης του συνόλου των διακομιστών οπότε να μην μπορεί να απαντήσει. Ο διακομιστής αρχείων ξαναπροσπαθεί ανά τακτά χρονικά διαστήματα μέχρι να λάβει απάντηση από τον διακομιστή χώρου ονομάτων. Όταν πλέον ο διακομιστής αρχείων λάβει το σύνολο των διακομιστών υποθέτει ότι επικεφαλής είναι ο λεξικογραφικά μικρότερος και είναι έτοιμος να αποστείλει τα αιτήματα.

Όταν ο διακομιστής αρχείων έχει ένα αίτημα για εξυπηρέτηση το στέλνει σε αυτό που γνωρίζει ως επικεφαλής Υπάρχει περίπτωση ενδιάμεσα να έχει αλλάξει η σύνθεση της υπηρεσίας χώρου ονομάτων. Σε αυτή την περίπτωση προσπαθεί να ξαναδιαβάσει τη νέα σύνθεση ώστε να αναγνωρίσει εκ νέου τον επικεφαλής.

4.3.2 Διαγράμματα Καταστάσεων

Σε αυτή την Ενότητα περιγράφεται το διαγράμματα καταστάσεων του επικεφαλής διακομιστή. Η μοντελοποίηση των κατανεμημένων πρωτοκόλλων με την χρήση πεπερασμένων μηχανών καταστάσεων είναι ο φορμαλιστικός τρόπος να δειχθεί η διαδικασία λειτουργίας τους. Στη συνέχεια χρησιμοποιείται ο συμβολισμός ? για είσοδο στο αυτόματο και ! για την έξοδο από αυτό. Με τους όρους είσοδο και έξοδο εννοούμε την αντίστοιχα λήψη και αποστολή μηνυμάτων.

Επικεφαλής Διακομιστής



Σχήμα 4.5: Διάγραμμα Καταστάσεων του Επικεφαλής Διακομιστή

Στο Σχήμα 4.5 φαίνεται το διάγραμμα καταστάσεων του επικεφαλής διακομιστή. Αρχικά βρίσκεται στην κατάσταση q_0 και περιμένει μέχρι να λάβει ένα αίτημα, οπότε και

μεταβαίνει στην κατάσταση q_1. Σε αυτή την κατάσταση κάνει ένα έλεγχο αν το αίτημα είναι νέο ή αν αποτελεί επανεκπομπή προηγούμενου. Στην περίπτωση που το μήνυμα είναι νέο μεταβαίνει στην κατάσταση q_2, όπου προωθεί σε όλους τους εφεδρικούς το αίτημα και πηγαίνει στην κατάσταση q_3. Όταν λάβει επιβεβαίωση από όλους τους εφεδρικούς προχωράει στην κατάσταση q_4 ώστε να εκτελέσει τοπικά το αίτημα και μετά να μεταβεί στην κατάσταση q_5 απ' όπου θα στείλει την απάντηση στον πελάτη. Σε αυτό το σημείο ολοκληρώνεται ο κύκλος ενός αιτήματος και το αυτόματο πηγαίνει στην κατάσταση q_0, ώστε να είναι σε θέση να εξυπηρετήσει το επόμενο αίτημα. Αν το αίτημα που έχει φτάσει αποτελεί διπλότυπο προηγούμενου μηνύματος τότε από την κατάσταση q_1 πηγαίνει στην q_6 ώστε να αποστείλει πίσω την προηγούμενη απάντηση.

4.4 Περίληψη

Το ομοσπονδιακά συστήματα αρχείων χρειάζονται μία υπηρεσία που να ενοποιεί τα προσφερόμενα συστήματα αρχείων. Αυτή η υπηρεσία δεν αρκεί να αποτελείται από ένα μόνο κόμβο που αναλαμβάνει να οργανώσει τους υπόλοιπους, για λόγους ανοχής σε σφάλματα.

Σε αυτό το Κεφάλαιο ορίστηκε η αρχιτεκτονική και οι ρόλοι που υπάρχουν στο σύστημα Orion. Το Orion χρησιμοποιεί ένα αξιόπιστο μηχανισμός ενοποίησης ονοματοχώρου με βάση τον πίνακα εξαγωγής. Η ενοποιημένη εικόνα του χώρου ονομάτων αποτελεί το χώρο ονομάτων του ομοσπονδιακού συστήματος αρχείων. Ακόμη περιγράφηκε πως κάθε διακομιστής αρχείων προσαρμόζει τον πίνακα εξαγωγής.

Επιπλέον ορίστηκε το πρωτόκολλο επικοινωνίας μεταξύ των διαφόρων οντοτήτων του συστήματος Orion. Επιπλεον, κάνοντας χρήση του πρωτοκόλλου Paxos περιγράφηκε μηχανισμός ώστε το σύστημα να ανακάμπτει σε περιπτώσεις σφαλμάτων των διακομιστών του χώρου ονομάτων. Στο τέλος του Κεφαλαίου σχεδιάστηκαν τα διαγράμματα καταστάσεων του επικεφαλής διακομιστή.

Κεφαλαιο 5

Υλοποιηση

5.1 Περιβάλλον Υλοποίησης

5.2 Πρόγραμμα Εξαγωγής Συστημάτων Αρχείων

5.3 Replicated State Machine

5.4 Πρωτόκολλο Paxos

5.5 Περίληψη

Σε αυτό το κεφάλαιο παρουσιάζεται η υλοποίηση του συστήματος σύμφωνα με το σχεδιασμός που περιγράφηκε στο Κεφάλαιο 4. Η υλοποίηση χωρίζεται σε δύο μέρη. Το πρώτο αναλαμβάνει την σωστή εξυπηρέτηση των αιτημάτων όταν η σύνθεση των μελών της υπηρεσίας χώρου ονομάτων παραμένει σταθερή. Το δεύτερο τμήμα εξασφαλίζει πως όταν η σύνθεση των μελών αλλάξει τότε οι κόμβοι που συμμετέχουν σε αυτή και είναι ενεργοί θα συμφωνήσουν σε μία νέα σύνθεση.

5.1 Περιβάλλον Υλοποίησης

Η υλοποίηση του συστήματος έγινε στην αντικειμενοστραφή γλώσσα προγραμματισμού C++ και χρησιμοποιήθηκε η βιβλιοθήκη STL η οποία προσφέρει υλοποιήσεις δομών για την αποθήκευση πληροφορίας όπως ένα *hash_map* το οποίο υλοποιεί ένα κοκκινόμαυρο δένδρο και αντιστοιχεί τα δεδομένα σε κατάλληλο κλειδί και ένα διάνυσμα που προσφέρει λειτουργίες εισαγωγής και διαγραφής. Κάθε server χρησιμοποιεί νήματα όπως υλοποιούνται στην βιβλιοθήκης των *Posix Threads*

Η επικοινωνία μεταξύ των διαφόρων οντοτήτων γίνεται με την χρήση RPC μηνυμάτων. Τα μηνύματα αυτά είναι δυνατόν να χαθούν, να σταλούν δύο φορές η να παραδοθούν κανονικά. Κάθε rpc μήνυμα δημιουργεί ένα καινούργιο νήμα το οποίο αναλαμβάνει να εξυπηρετήσει την αίτηση καλώντας την κατάλληλη ρουτίνα.

5.2 Πρόγραμμα Εξαγωγής Συστημάτων Αρχείων

Σε αυτή την ενότητα περιγράφεται η υπηρεσίας της οποίας θέλουμε να κρατάμε ασφαλή αντίγραφα, αυτό είναι το Πρόγραμμα Εξαγωγής Συστημάτων Αρχείων (ΠΕΣΑ). Η συγκεκριμένη υπηρεσία βασίζεται στο αρχείο όπου κρατάμε τον πίνακα εξαγωγών των συστημάτων αρχείων του NFSv4. Υποστηρίζει λειτουργία εισαγωγής, διαγραφής, ανανέωσης και λειτουργία διαβάσματος του τρέχοντος πίνακα.

Ο διακομιστής ΠΕΣΑ αποτελεί μέρος μία *Μηχανής Καταστάσεων Πολλαπλών αντιγράφων (ΜΚΠΑ)*. Όταν δημιουργείται ένας διακομιστής ΠΕΣΑ δημιουργείται ένα αντικείμενο τύπου *ΜΚΠΑ*. Στη συνέχεια γίνεται η σύνδεση των δύο αντικειμένων ώστε να μπορεί η (ΜΚΠΑ) να εκτελέσει τις κατάλληλες λειτουργίες που της ζητούνται και είναι μέρος της υλοποίησης του προγράμματος ΠΕΣΑ.

Με αυτό τον τρόπο υλοποίησης μπορεί να γίνει χρήση της ΜΚΠΑ και για άλλες υπηρεσίες που θέλουμε να υλοποιήσουμε. Δεν εισάγει επιπλέον επιβάρυνση ούτε σε χρόνο ούτε σε μνήμη, και οι λειτουργίες μεταξύ τους είναι ξεκάθαρες.

5.2.1 Υλοποίηση Διακομιστή Ονομάτων

Με τον όρο διακομιστή Ονομάτων εννοούμε ένα διακομιστή που συμμετέχει στη υπηρεσία χώρου ονομάτων. Ο συγκεκριμένος διακομιστής πρέπει να μπορεί να αντιστοιχίσει ένα σύστημα αρχείων στον κατάλληλο διακομιστή NFSv4. Για να το κάνει αυτό κρατάει σε ένα αρχείο εγγραφές με:

1. τον διακομιστή αρχείων που το προσφέρει το σύστημα αρχείων
2. το μονοπάτι στον διακομιστή αρχείων
3. το μονοπάτι στο οποίο γίνεται διασύνδεση στο ομοσπονδιακό σύστημα αρχείων
4. τα ορίσματα για το συγκεκριμένο σύστημα αρχείων

Τα πρώτα δύο χρειάζονται για να ξέρουμε ποια είναι τα συστήματα αρχείων και ποιος προσφέρει το κάθε ένα, αυτά είναι απαραίτητα αλλιώς δεν γίνεται να τα μάθουν οι υπόλοιποι διακομιστές. Τα υπόλοιπα χρειάζονται ώστε όλοι οι πελάτες να βλέπουν στο ίδιο μονοπάτι τα συστήματα αρχείων ανεξαρτήτως τον NFSv4 διακομιστή που γνωρίζουν ως αρχικό.

Οι διακομιστές ονομάτων απλώς λαμβάνουν αιτήματα από του NFSv4 διακομιστές και καταγράφουν την πληροφορία που τους στέλνουν. Δεν κάνουν κανένα έλεγχο αν αυτό που έλαβαν είναι σωστό ή αν ο διακομιστής που τους το έστειλε προσφέρει όντως το συγκεκριμένο σύστημα αρχείων

5.2.2 Υλοποίηση Πελάτη Ονομάτων

Ο πελάτης Ονομάτων στο σύστημα Orion είναι ένας NFSv4 διακομιστής. Τα αιτήματα που στέλνει στην υπηρεσία ονομάτων είναι είτε για να διαβάσει το τρέχοντα πίνακα εξαγομένων

συστημάτων αρχείων ή να προσθέσει ή να διαγράψει κάποιο υπάρχων. Η προσθήκη και η διαγραφή υλοποιείται με μία κατάλληλη απομακρυσμένη κλήση.

Στην περίπτωση που ο NFSv4 διακομιστής διαβάσει ένα πίνακα εξαγομένων συστημάτων αρχείων πρέπει να τον μετατρέψει σε μορφή τέτοια ώστε να είναι κατάλληλος για να αντικαταστήσει το αρχείο */etc/exports* Για τα συστήματα αρχείων που ανήκουν σε κάποιο άλλο διακομιστή αρκεί να γράψει στο αρχείο το μονοπάτι στο οποίο γίνεται η διασύνδεση, να βάλει τα ορίσματα και να βάλει και ένα επιπλέον όρισμα που είναι η επιλογή πως το συγκεκριμένο σύστημα αρχείων γίνεται διαθέσιμο μέσω της υπηρεσίας της ανακατεύθυνσης του NFSv4. Τέλος χρειάζεται να δημιουργήσει ένα διαφορετικό *fsid* για το συγκεκριμένο φάκελο.

Το fsid είναι το αναγνωριστικό κάθε συστήματος αρχείων και είναι μοναδικό για κάθε υπολογιστή. Όταν ο πελάτης του συστήματος NFSv4 ζητάει να διασυνδέσει ένα σύστημα αρχείων το ζητάει με βάση το fsid που έχει στον διακομιστή αρχείων. Για να καταλάβει ο πελάτης πώς ένα μονοπάτι είναι διαφορετικό σύστημα αρχείων πρέπει να έχει και διαφορετικό fsid. Με αυτό τον τρόπο θα ζητήσει να διασυνδέσει και το συγκεκριμένο σύστημα αρχείων Στην περίπτωσης της ανακατεύθυνσης ο διακομιστής αποδίδει στο φάκελο στο οποίο είναι η ανακατεύθυνση ένα διαφορετικό fsid. Με τον τρόπο αυτό ο πελάτης θα ζητήσει να διασυνδεθεί με τον συγκεκριμένο σύστημα αρχείων και τότε ο πυρήνας του λειτουργικού του διακομιστή θα αναγνωρίσει πως το συγκεκριμένο σύστημα αρχείων δεν είναι τοπικό αλλά απομακρυσμένο, οπότε θα δώσει οδηγίες στο πελάτη που βρίσκεται το απομακρυσμένο σύστημα αρχείων. Για να γίνει η διαδικασία της απόδοσης ενός fsid σε ένα φάκελο χωρίς να υπάρχει πραγματικά ένα διαφορετικό σύστημα αρχείων από πίσω χρειάζεται να εκτελεστεί η εντολή

$$mount\ --bind\ < path > < path >$$

Στην περίπτωση που το εξαγόμενο σύστημα αρχείων το προσφέρει ο ίδιος δεν χρειάζεται να κάνει κάτι διαφορετικό. Απλώς γράφει το μονοπάτι μαζί με τα κατάλληλα ορίσματα. Τέλος όταν έχει τελειώσει με την όλη διαδικασία χρειάζεται να κάνει μία ανανέωση των εξαγομένων συστημάτων στις δομές που καταγράφει ο πυρήνας. Για να γίνει αυτό αρκεί να τρέξει την εντολή:

$$exportfs\ -r$$

5.3 Μηχανή Καταστάσεων Πολλαπλών Αντιγράφων

Σε αυτή την ενότητα περιγράφεται η υλοποίηση της Replicated State Machine (RSM). Το συγκεκριμένο τμήμα του συστήματος αποτελείται από δύο ειδών διακομιστές. Είναι ο επικεφαλής ο οποίος δέχεται τα αιτήματα από τους πελάτες και οι εφεδρικοί οι οποίοι αποθηκεύουν αντίγραφα της κατάστασης ώστε να είναι διαθέσιμη η υπηρεσία σε περίπτωση αποτυχίας του επικεφαλής διακομιστή. Σε αυτό το κομμάτι ο ρόλος των παραπάνω είναι διακριτός. Στη συνέχεια όμως και με την χρήση του Paxos όπου θα επιτρέπουμε αλλαγές

στη σύνθεση, κάθε διακομιστής μπορεί να είναι είτε εφεδρικός είτε επικεφαλής, και να υπάρχουν αλλαγές όταν αποτύχει ένας επικεφαλής. Για το λόγο αυτό τα αντικείμενα αυτά δεν είναι χωριστά, αλλά ομαδοποιούνται σε ένα ενιαίο.

5.3.1 Επικεφαλίδα της ΜΚΠΑ

Στον Πίνακα 5.3.1 φαίνονται τα δεδομένα που αποθηκεύει κάθε αντικείμενο τύπου ΜΚΠΑ. Τα δύο πρώτα είναι ώστε να στέλνει και να λαμβάνει μηνύματα για απομακρυσμένες κλήσεις συναρτήσεων. Το επόμενο είναι το σύνολο των διακομιστών που γνωρίζει, ενώ το επόμενο είναι το αντικείμενο τύπου Paxos ώστε να μπορεί να συμμετέχει στην λήψη αποφάσεων όσον αφορά την σύνθεση των διακομιστών.

Πίνακας 5.1: Δομές Δεδομένων αντικειμένου ΜΚΠΑ μαζί με μία σύντομη περιγραφή για κάθε πεδίο

Τύπος	**Αντικείμενο**	**Περιγραφή**
*rpcc**	*cl*	για την αποστολή rpc μηνυμάτων στου υπόλοιπους διακομιστές
*rpcs**	*rsmrpc*	για την λήψη rpc μηνυμάτων
*view**	*grp*	αποθηκεύει το σύνολο των διακομιστών
*paxos**	*paxos_obj*;	χρησιμοποιείται για να συμφωνούν οι διακομιστές στην τρέχουσα σύνθεση. Περιγράφεται αναλυτικά στην Ενότητα 5.4
HASH_TABLE	*client_requests*	ένα hash table που χρησιμοποιείται για τον έλεγχο τον διπλότυπων μηνυμάτων

Στον Πίνακα 5.3.1 φαίνονται οι απομακρυσμένες συναρτήσεις που προσφέρει ένα αντικείμενο τύπου ΜΚΠΑ. Η πρώτη ρουτίνα είναι ένα αίτημα που στέλνει ο πελάτης ώστε να διαβάσει την τρέχουσα σύνθεση. Αυτό το αίτημα μπορεί να φτάσει σε οποιονδήποτε από τους διακομιστές. Όπως θα δούμε στη συνέχεια αυτή η ρουτίνα χρησιμοποιείται κατά την αρχικοποίηση του πελάτη. Η δεύτερη ρουτίνα χρησιμοποιείται πάλι από τον πελάτη ώστε να αποσταλεί το αίτημα στον επικεφαλής διακομιστή. Η τρίτη ρουτίνα είναι και αυτή αποτέλεσμα αιτήματος rpc προς τον επικεφαλής, αλλά την αποστέλλει ένας εφεδρικός. Η τελευταία ρουτίνα είναι αυτή με την οποία ο επικεφαλής προωθεί το μήνυμα προς τους εφεδρικούς.

5.3.2 Επικεφαλής Διακομιστής

Ο Επικεφαλής διακομιστής καθ όλη τη διάρκεια που το σύστημα λειτουργεί σωστά έχει δύο ρόλους. Ο ένας αφορά τη σωστή επικοινωνία με τους εφεδρικούς, ενώ ο δεύτερος είναι η διεπαφή των πελατών με το σύστημα.

Αρχικά λαμβάνει μηνύματα από τους εφεδρικούς ώστε αφενός οι εφεδρικοί να γνωρίζουν πως λειτουργεί σωστά και αφετέρου να μπορεί και αυτός να γνωρίζει πως οι εφεδρικοί λειτουργούν σωστά. Για το λόγο αυτό κάθε εφεδρικός στέλνει κάθε ένα δευτερόλεπτο ένα

Πίνακας 5.2: Συναρτήσεις Αντικειμένου ΜΚΠΑ μαζί με μία σύντομη περιγραφή για κάθε πεδίο

client_members(vector<string> &r)	ρουτίνα που αποστέλλει στον πελάτη την τρέχουσα σύνθεση της υπηρεσίας. Το αποτέλεσμα αποθηκεύεται στο διάνυσμα r
client_invoke(int procno, string req, int cl_id, int cl_seqno, int now, string &r)	ρουτίνα που καλεί ο πελάτης για να αποστείλει ένα αίτημα στο επικεφαλής διακομιστή. Το procno είναι ο αριθμός της ρουτίνας, το req είναι το αίτημα που στέλνει, το cl_id είναι το αναγνωριστικό του πελάτη, το cl_seqno είναι ο αριθμός αιτήσεως που έχει δώσει ο πελάτης και το now είναι η χρονική στιγμή που έστειλε ο πελάτης το αίτημα. Το αποτέλεσμα της κλήσης επιστρέφεται στο r.
heartbeat(string m, int &r)	λαμβάνει ένα μήνυμα από ένα εφεδρικό διακομιστή για να βεβαιωθεί η σωστή λειτουργία. Το m είναι το αναγνωριστικό του εφεδρικού διακομιστή και r είναι το αποτέλεσμα. Όταν δηλαδή λειτουργεί ο εφεδρικός επιστρέφει OK
invoke(int proc, seqno, string req, int id, int clientseq, int &r)	ρουτίνα που καλεί ο επικεφαλής διακομιστής για να προωθήσει ένα αίτημα προς τους εφεδρικούς. Τα πεδία είναι ίδια με αυτής που στέλνει ο πελάτης στον επικεφαλής.

heartbeat μήνυμα στον επικεφαλής. Αυτό γίνεται με την κλήση της rpc ρουτίνας *heartbeat*, ο κώδικας της οποίας εκτελεί ο κάθε εφεδρικός φαίνεται στον Αλγόριθμο 5.1

Η σημαντικότερη λειτουργία του επικεφαλής είναι η λήψη αιτημάτων από τον πελάτη. Όλα τα μηνύματα πρέπει να περνάνε από τον επικεφαλής ώστε να σειριοποιούνται. Η λήψη των αιτημάτων γίνεται με την ρουτίνα *client_invoke*, της οποίας ο κώδικας φαίνεται στον Αλγόριθμο 5.2.

```
while (1) {
  gettimeofday(&now, NULL);
  next_timeout.tv_sec = now.tv_sec + 1;
  pthread_cond_timedwait(&heartbeat_cond,
    &rsm_mutex, &next_timeout);
  m = grp->master();
  /*if i not master i have to send heartbeat message*/
  if (m != grp->me()) {
    int ret, r;
    pthread_mutex_unlock(&rsm_mutex);
    /*send message*/
    ret = cl.call(grp->sockaddr(m), rsm_protocol::heartbeat,
      grp->me(), r, rpcc::to(HEARTBEAT_TIMEOUT));
    pthread_mutex_lock(&rsm_mutex);
    if (ret != rsm_protocol::OK) {
      /*if an error occured start new paxos instance*/
      printf("heartbeater: master doesn't respond; remove\n");
      paxos_obj->change(m);
      printf("new master is %s\n", grp->master().c_str());
    }
  }
}
```

Αλγόριθμος 5.1: Μηνύματα heartbeat

Ο επικεφαλής πρέπει να αποθηκεύει τα αιτήματα που δέχεται από τους πελάτες ώστε να μπορεί να αναγνωρίσει αν είναι νέο το μήνυμα ή αν αποτελεί επανεκπομπή προηγούμενου μηνύματος. Για τον λόγο αυτό αποθηκεύει σε ένα hash_map το ζεύγος $< cl\_id, cl\_seqno >$, τα οποία είναι το αναγνωριστικό του πελάτη μαζί με τον αύξοντα αριθμό που δίνει ο πελάτης στην αίτηση του. Το hash_map είναι η μεταβλητή client_requests που φαίνεται στον Πίνακα 5.3.1. Για κάθε τέτοιο ζεύγος αποθηκεύει τέσσερα χαρακτηριστικά.

- την ώρα που το έστειλε ο πελάτης ώστε να μπορεί να αναγνωρίζει αν είναι πολύ παλιό το μήνυμα
- τον αριθμό αίτησης που θα δώσει ο επικεφαλής σε αυτή την αίτηση
- το αποτέλεσμα ώστε σε περίπτωση επανεμφάνισης του μηνύματος να γνωρίζει τι είχε απαντήσει και να μην χρειάζεται να το υπολογίσει ξανά
- η κατάσταση της αίτησης

```
rsm_client_protocol::status
rsm::client_invoke(int procno, string req, unsigned int id,
  unsigned int seqno, unsigned int req_time, string &r)
{
  if(!stable()) return BUSY;
  if(!amimaster()) return NOTMASTER;
  membs=grp->members(); /*read current members*/
  check=check_client(id,seqno,req_time);
  if(check==-1)   throw "DUPLICATED MESSAGE";
  else if(check == -3)  return VERY_OLD;
  else if(check==0){
    r=check_it->second.result;
    return OK;
  }
  grp->incseqno();
  seq=grp->seqno();
  /*send request to all members but me(master)*/
  for(i=0; i<membs.size(); i++){
    /*if i become unstable stop serving the request*/
    if(paxos_obj->stable()==false)
      return rsm_client_protocol::BUSY;
    if(membs.at(i)!=grp->me()){
      struct sockaddr_in tmp=grp->sockaddr(membs.at(i));
      ret=cl.call(tmp, rsm_protocol::invoke, procno, seq, req,
        id,seqno,r2, rpcc::to(RSM_SLAVE_TIMEOUT));
      /*if an error occured remove member and begin
       *new paxos instance*/
      if(ret==-1 || ret==rsm_protocol::BUSY){
        if(ret==-1)
          paxos_obj->change(membs.at(i));
        return rsm_client_protocol::BUSY;
      }
    }
  }
  /*execute the command locally and store it in hash_map*/
  r=execute(procno,req);
  ctmp=(id,seqno);
  rtmp.status=rsm_protocol::OK;
  rtmp.result=r;
  rtmp.masterseq=grp->seqno();
  client_requests[ctmp]=rtmp;
  return rsm_client_protocol::OK;
}
```

Αλγόριθμος 5.2: Λειτουργίες του Επικεφαλής αφού λάβει ένα αίτημα

Η κατάσταση της αίτησης μπορεί να είναι είτε σε αναμονή (PENDING), αν τώρα εξυπηρετείται είτε να είναι τελειωμένη (DONE), αν έχει εξυπηρετηθεί. Υπάρχει πιθανότητα να λάβει ένα αίτημα να το προωθήσει για εξυπηρέτηση και ενώ δεν έχει τελειώσει να λάβει και δεύτερη φορά το ίδιο αίτημα. Σε αυτή την περίπτωση δεν έχει κάποια απάντηση να του δώσει αλλά επίσης δεν μπορεί να το εξυπηρετήσει για δεύτερη φορά. Έτσι το συγκεκριμένο μήνυμα πρέπει να αγνοηθεί.

Η παραπάνω δομή συνεχώς αυξάνει με αποτέλεσμα να μεγαλώνει πάρα πολύ. Για να λυθεί το παραπάνω πρόβλημα υπάρχει ένα νήμα το οποίο αναλαμβάνει περιοδικά να κάνει εκκαθάριση, διαγράφοντας παλιά αιτήματα. Αυτό επιτυγχάνεται συγκρίνοντας την τρέχουσα χρονική στιγμή με την χρονική στιγμή στην οποία έλαβε το αίτημα. Αν η διαφορά αυτών των δύο ξεπερνάει το κατώφλι $CLEAN\_TIMEOUT$, τότε διαγράφεται από τη λίστα. Το παραπάνω κατώφλι αποτελεί παράμετρο του συστήματος όπως και η περίοδος με την οποία ξεκαθαρίζονται οι παλιές αιτήσεις, η $CLEAN\_INTERVAL$.

Η παραπάνω εκκαθάριση εισάγει ένα επιπλέον πρόβλημα. Μπορεί ο πελάτης να έχει στείλει ένα αίτημα το οποίο για περίεργους λόγους καθυστέρησε και έχει περάσει ο χρόνος $CLEAN\_TIMEOUT$. Όταν θα φτάσει στο σύστημα δεν θα υπάρχει τρόπος να ελέγχει αν είναι καινούργιο μήνυμα ή αν αποτελεί επανεκπομπη. Από την στιγμή που έχει περάσει το κατώφλι ότι υπήρχε μέσα στο hash_map θα έχει διαγραφεί. Για να λυθεί αυτό το πρόβλημα ο πελάτης στέλνει μαζί με την αίτηση και την χρονική στιγμή που την έστειλε. Έτσι ο επικεφαλής όταν λάβει ένα αίτημα συγκρίνει αυτή τη χρονική στιγμή πόσο απέχει από την τρέχουσα χρονική στιγμή του συστήματος. Αυτή η διαφορά δεν πρέπει να είναι παραπάνω από $CLEAN\_TIMEOUT$ τότε του στέλνει απάντηση πως το αίτημα είναι πολύ παλιό(VERY_OLD) και να ξαναπροσπαθήσει, να δεν έχει λάβει ήδη κάποια απάντηση. Για να λειτουργήσει σωστά ο παραπάνω μηχανισμός υποθέτουμε πως τα ρολόγια τον διαφόρων οντοτήτων είναι συγχρονισμένα. Αυτό γίνεται κάνοντας χρήση του πρωτοκόλλου ntp [6].

Ο επικεφαλής πρέπει να προβλέψει και το ενδεχόμενο που κάποιος εφεδρικός παρουσιάσει πρόβλημα και αποτύχει. Σε αυτή την περίπτωση θα προσπαθήσει να στείλει το αίτημα, για το οποίο δεν θα πάρει ποτέ απάντηση και θα λήξει ο χρόνος αναμονής (timeout). Τότε πρέπει να διακοπεί η λειτουργία εξυπηρέτησης αιτημάτων από τους πελάτες και να διαγράφει ο εφεδρικός από την τρέχουσα σύνθεση των διακομιστών χρησιμοποιώντας το πρωτόκολλο Paxos. Για να γίνει αυτό ξεκινάει ένα στιγμιότυπο του πρωτοκόλλου Paxos, του οποίου η λειτουργία περιγράφεται στην Ενότητα 5.4

Συνοψίζοντας, όταν λάβει ένα αίτημα ο επικεφαλής εκτελεί τα επόμενα βήματα:

1. Ελέγχει αν το αίτημα το έχει λάβει πρώτη φορά ή αν αποτελεί επανεκπομπή προηγούμενου μηνύματος. Για να το κάνει αυτό χρησιμοποιεί το *hash_map*, client requests.

2. Αποδίδει στη αίτηση ένα μονοτονικά αυξανόμενο αριθμό. Αυτός αποτελεί πλέον το αναγνωριστικό της αίτησης.

3. Προωθεί το αίτημα στους εφεδρικούς διακομιστές κάνοντας χρήση της απομακρυσμένης ρουτίνας, *invoke* Αν όλοι απαντήσουν θετικά τότε προχωράει παρακάτω. Αν όμως παρουσιαστεί κάποιο πρόβλημα τότε διακόπτει τη λειτουργία εξυπηρέτησης αιτημάτων

και προσπαθεί να διαγράψει τον εφεδρικό διακομιστή που έχει αποτύχει. Η διαδικασία προώθησης αιτημάτων στους εφεδρικούς γίνεται σειριακά και όχι παράλληλα. Ο λόγος της συγκεκριμένης επιλογής είναι η περίπτωση ανίχνευσης πολλαπλών αποτυχιών. Τότε ο επικεφαλής θα ξεκινούσε πολλαπλά στιγμιότυπα του Paxos το οποίο είναι λανθασμένο.

4. Αν τελειώσει επιτυχώς η διαδικασία προώθησης του αιτήματος στους εφεδρικούς, τότε εκτελεί τοπικά το αίτημα, αποθηκεύει την απάντηση στη δομή με τις προηγούμενες απαντήσεις και τέλος στέλνει το αποτέλεσμα στον πελάτη.

5.3.3 Εφεδρικός Διακομιστής

Οι εφεδρικοί διακομιστές κατά τη διαδικασία σωστής λειτουργίας του συστήματος αναλαμβάνουν και αυτοί, όπως και ο επικεφαλής δύο λειτουργίες. Η πρώτη σχετίζεται και εδώ με τη σωστή επικοινωνία με τον επικεφαλής ενώ η δεύτερη ασχολείται με την διατήρηση αντιγράφων της υπηρεσίας.

Σύμφωνα με τα παραπάνω στέλνουν περιοδικά, κάθε ένα δευτερόλεπτο, μηνύματα τύπου *heartbeat* με τα οποία εξασφαλίζουν τη σωστή επικοινωνία με τον επικεφαλής. Σε περίπτωση που το αίτημα αποτύχει, δηλαδή περάσει χρόνος $HEARTBEAT\_TIMEOUT$, τότε ξεκινάει ένα στιγμιότυπο του Paxos για να αφαιρεθεί ο επικεφαλής από την τρέχουσα σύνθεση και οι εφεδρικοί να αποφασίσουν ποίος θα είναι ο νέος επικεφαλής. Υπάρχει περίπτωση ένας εφεδρικός μόνο να χάσει την επικοινωνία τους με τον επικεφαλής. Τότε θα ξεκινήσει ένα στιγμιότυπο του Paxos αλλά θα αποτύχει να αλλάξει τη σύνθεση καθώς οι υπόλοιποι θα ξέρουν ότι ο επικεφαλής είναι ενεργός και λειτουργεί σωστά.

Η δεύτερη λειτουργία των εφεδρικών αφορά τη διατήρηση αντιγράφων. Κάθε φορά που ο επικεφαλής λάβει ένα αίτημα το στέλνει τους εφεδρικούς ώστε να εφαρμόσουν και αυτοί το αίτημα που ζητάει ο client. Ο κάθε εφεδρικός ελέγχει αν το αίτημα είναι το επόμενο από αυτό που είχε λάβει τελευταίο, ώστε να μπορεί να αναγνωρίσει αν έχει χάσει κανένα μήνυμα ή αν του έρχονται με διαφορετική σειρά. Έτσι στη συνέχεια εκτελεί το αίτημα, και επιστρέφει στον επικεφαλής αν έγινε επιτυχώς ή όχι.

Στις προηγούμενες υλοποιήσεις του πρωτοκόλλου Paxos προτείνεται η καταγραφή σε ένα *σειριακό αρχεία καταγραφής (log)* της αίτησης που πρόκειται να εκτελεστεί. Ύστερα πρέπει να γνωστοποιήθει στον επικεφαλής πως καταγράφηκε η αλλαγή και στη συνέχεια να γίνει η ίδια η αλλαγή. Ο λόγος που προτείνεται αυτό είναι πως η ίδια η αλλαγή μπορεί να πάρει αρκετή ώρα. Έτσι στη περίπτωση που θα προκύψει κάποια αποτυχία θα μπορεί να ανακάμψει διαβάζοντας το σειριακό αρχεία καταγραφής. Γενικώς στη βιβλιογραφία θεωρείται πως η εγγραφή στο ημερολόγιο είτε γίνεται είτε όχι, και ο λόγος είναι πως είναι πολύ μικρή και διαρκεί ελάχιστο.

Στο σύστημα Orion η υπηρεσία της οποίας θέλουμε να κρατάμε αντίγραφα είναι ο πίνακας προσφερόμενων συστημάτων αρχείων του NFSv4. Έτσι μία αλλαγή σε αυτό το αρχείο είναι πολύ μικρή. Είτε πρέπει να προστεθεί στο τέλος μία εγγραφή, αντίστοιχα με το ημερολόγιο είτε πρέπει να διαγράφει μία γραμμή το οποίο πάλι διαρκεί ελάχιστο.

5.3.4 Πελάτης ΜΚΠΑ υπηρεσίας

Στη συγκεκριμένη ενότητα περιγράφεται η λειτουργία του πελάτη της RSM υπηρεσίας. Στη δική μας περίπτωση πελάτης είναι ένας διακομιστής NFSv4, ο οποίος ζητάει από την υπηρεσία να λάβει το τρέχοντα πίνακα εξαγόμενων συστημάτων αρχείων. Στη γενική περίπτωση είναι ένας πελάτης της εκάστοτε υπηρεσίας. Η υλοποίηση έχει γίνει έτσι ώστε να είναι μην χρειάζεται η συγκεκριμένη οντότητα να έχει γνώση της όλης υπηρεσίας. Με αυτό τον τρόπο μπορούν να κάνουν χρήση του συγκεκριμένου αντικειμένου διάφορες υπηρεσίες. Στη συνέχεια περιγράφεται η λειτουργία του πελάτη.

Πίνακας 5.3: Πιθανές απαντήσεις από τον διακομιστή με αντίστοιχη περιγραφή και ενέργεια του πελάτη.

Απάντηση	**Περιγραφή**	**Αντίδραση**
OK	το αίτημα ολοκληρώθηκε κανονικά και επιστράφηκε το αποτέλεσμα	χρήση του αποτελέσματος
BUSY	το σύστημα είναι απασχολημένο με αιτήματα συντήρησης	περιμένει για τυχαίο χρονικό διάστημα $[0, RSM\_SLEEP]$, και επανεκπέμπει το αίτημα με καινούργιο σειριακό αριθμό
NOT_MASTER	ο server που στάλθηκε το αίτημα δεν είναι ο master, αλλά ένας slave	ζητάει από τους servers που γνωρίζει να του στείλουν το ανανεωμένο σύνολο με τους servers του συστήματος, και θέτει ως master το λεξικογραφικά μικρότερο
VERY_OLD	το αίτημα είναι πολύ παλιό. Αυτό μπορεί να συμβεί στην περίπτωση που αργήσει να φτάσει λόγω συνθηκών του δικτύου	επανεκμπέμπει το αίτημα με νέο seqno
-1(ERROR)	ο server δεν απαντάει	προχωράει στον επόμενο server που γνωρίζει

Την πρώτη φορά που θα προσπαθήσει ο πελάτης να συνδεθεί πρέπει να γνωρίζει ένα διακομιστή. Σε αυτόν στέλνει το αρχικό αίτημα για διασύνδεση ζητώντας τη λίστα με όλους τους διακομιστές του συστήματος. Η υπηρεσία μπορεί να είναι απασχολημένη με διαδικασίες εκλογής αρχηγού ή αλλαγής της σύνθεσης του συνόλου των διακομιστών οπότε να μην μπορεί να απαντήσει. Ο πελάτης ξαναπροσπαθεί ανά τακτά χρονικά διαστήματα μέχρι να λάβει απάντηση από τον διακομιστή που γνωρίζει. Σε περίπτωση που ο συγκεκριμένος διακομιστής είναι ανενεργός ή έχει αποτύχει τότε ο πελάτης δεν μπορεί να συνδεθεί με την υπηρεσία. Σε αυτή την περίπτωση δεν μπορεί να λυθεί το πρόβλημα και υποθέτουμε πως πρέπει να μάθει κάποιον άλλο. Όταν πλέον ο πελάτης λάβει το σύνολο των διακομιστών

υποθέτει ότι επικεφαλής είναι ο λεξικογραφικά μικρότερος και είναι έτοιμος να αποστείλει τα αιτήματα.

Όταν ο πελάτης έχει ένα αίτημα για εξυπηρέτηση το στέλνει σε αυτό που γνωρίζει ως επικεφαλής. Στο μήνυμα ενσωματώνει το μοναδικό αριθμό (id) που περιγράφει ποιος πελάτης είναι, ένα μονοτονικά αυξανόμενο σειριακό αριθμό (seqno) και την τοπική ώρα του συστήματός του. Η λόγο αποστολής του κάθε ορίσματος περιγράφεται στην Ενότητα 5.3.2 όπου γίνεται περιγραφή του επικεφαλής διακομιστή. Στον Πίνακας 5.3.4 φαίνονται οι απαντήσεις που μπορεί να λάβει ο πελάτης μαζί με μία περιγραφή καθώς και την ενέργεια στην οποία θα προχωρήσει.

5.4 Πρωτόκολλο Paxos

Σε αυτό το κεφάλαιο περιγράφεται η υλοποίηση του πρωτοκόλλου Paxos. Η επικοινωνία μεταξύ των διακομιστών γίνεται με χρήση απομακρυσμένων κλήσεων διαδικασιών (RPC) όπως και στην υλοποίηση του πρωτοκόλλου ολοκλήρωσης δύο φάσεων. Υπάρχουν τριών είδη μηνύματα που αφορούν το ίδιο το πρωτόκολλο. Αυτά είναι τα *prepare,accept,decide*. Επιπλέον υπάρχει μία απομακρυσμένη διαδικασία η οποία αναλαμβάνει να κάνει μεταφορά της κατάστασης από ένα διακομιστή σε ένα δεύτερο.

5.4.1 Δημιουργία Αντικειμένου

Ένα αντικείμενο τύπου Paxos δημιουργείται όταν ξεκινάει ο διακομιστής. Τότε δημιουργείται ένα νήμα το οποίο είναι υπεύθυνο να ξεκινάει τα διαφορετικά στιγμιότυπα Αυτό το νήμα ονομάζεται *διαχειριστής* και είναι υπεύθυνο να κάνει και τις αλλαγές ανάμεσα στις διαφορετικά φάσεις του πρωτοκόλλου. Το νήμα διαχειριστής κοιμάται όσο λειτουργεί το σύστημα και δεν υπάρχουν αποτυχίες διακομιστών. Σε περίπτωση όμως που ανιχνευθεί μία αλλαγή στη σύνολο των διακομιστών τότε του στέλνεται ένα σήμα ώστε να ξεκινήσει νέο στιγμιότυπο Το σήμα το στέλνει το ΜΚΠΑ αντικείμενο, το οποίο έχει ανιχνεύσει την αλλαγή και έχει διαγράψει το διακομιστή που έχει αποτύχει.

5.4.2 Δεδομένα Αντικειμένου

Σύμφωνα με το Πρωτόκολλο Paxos κάθε διακομιστής πρέπει να καταγράφει κάποια δεδομένα. Αυτά έχουν να κάνουν τις τιμές που έχει δεχτεί ή έχει απαντήσει σε αιτήματα proposal. Όλες οι τιμές φαίνονται στον Πίνακα 5.4.2, όπου υπάρχει περιγραφή για το τί αποθηκεύει η κάθε μεταβλητή.

Όταν ο διαχειριστής λάβει το σήμα τότε αρχικοποιεί τις μεταβλητής ώστε να ξεκινήσει η πρώτη φάση του αλγορίθμου. Οι μεταβλητές που χρειάζονται αρχικοποίηση είναι οι: $maxnum, myprop, maxprop, maxval$ και $done$. Οι τρεις πρώτες αρχικοποιούνται στο μηδέν, η επόμενη στο κενό σύνολο και η τελευταία στη λογική τιμή $false$. Κάθε διακομιστής μπορεί και να λάβει και να στείλει αιτήματα. Έτσι σε κάθε φάση υπάρχουν δύο ρόλοι, ένας

Πίνακας 5.4: Μεταβλητές του Paxos και επεξήγηση αυτών

Αντικείμενο	**Περιγραφή**
maxval	μεγαλύτερη τιμή που έχει αποδεχτεί
maxnum	μεγαλύτερο προτεινόμενο αριθμό που έχει αποδεχτεί
maxprop	η μεγαλύτερη προτεινόμενη τιμή που έχει συναντήσει σε αίτημα προετοιμασίας
myprop	προτεινόμενο αριθμό που έχει χρησιμοποιήσει σε αυτό το στιγμιότυπο του Paxos
maxvid	μεγαλύτερο αριθμό όψης/σύνθεσης που έχει αποδεχτεί
done	λογική μεταβλητή που υποδηλώνει αν έχει επιτευχθεί συμφωνία για την σύνθεση.

για αυτούς τους διακομιστές που στέλνουν προτάσεις και ένας για αυτούς που λαμβάνουν.

5.4.3 Πρώτη Φάση

```
1  one or multiple nodes decides to begin a Paxos instance
2    myprop = max(maxprop, myprop)+1, append node ID
3    done = false
4    sends prepare(maxvid+1, myprop) to all nodes in
5      [currentview, initial contact node, itself]
6
7  if node receives prepare(vid, n):
8    if vid <= maxvid:
9      return oldview(maxvid, views[maxvid])
10   else if n > maxprop:
11     maxprop = n
12     done = false
13     return prepareres(maxnum, maxval)
14   else:
15     return reject()
```

Αλγόριθμος 5.3: Πρώτη Φάση του Πρωτοκόλλου Paxos

Στον Αλγόριθμο 5.3 φαίνεται σε μορφή ψευδοκώδικα οι ενέργειες στις οποίες προβαίνει και αυτός που ξεκινάει ένα στιγμιότυπου του πρωτοκόλλου Paxos αλλά και αυτός που λαμβάνει τα μηνύματα. Στη συνέχεια υπάρχει λεπτομερής περιγραφεί για κάθε περίπτωση.

Αποστολή Αιτήσεων

Στην πρώτη φάση τουλάχιστον ένας διακομιστής αποφασίζει να προτείνει μία αλλαγή. Για να το κάνει αυτό πρέπει να επιλέξει ένα αριθμό πρότασης. Ο αριθμός αυτός είναι το μέγιστο ανάμεσα στην προηγούμενη τιμή που είχε προτείνει και στον μεγαλύτερο αριθμό πρότασης

αν έχει λάβει κάποιον, αυξημένος κατά ένα. Πιο φορμαλιστικά ορίζεται ως:

$$my\_n = max(n\_h, my\_n) + 1 \tag{5.1}$$

Ό αριθμός αίτησης εκτός από μία αριθμητική τιμή περιλαμβάνει και το αναγνωριστικό του διακομιστή. Έτσι στην περίπτωση όπου δύο διακομιστές προτείνουν την ίδια τιμή θα επικρατήσει αυτός με το λεξικογραφικά μεγαλύτερο αναγνωριστικό.

Αφού διαλέξει ποια ακριβώς τιμή θα προτείνει τότε στέλνει ένα αίτημα προετοιμασίας(prepare), στο οποίο ενσωματώνει την τιμή που διαλέξει καθώς και το αναγνωριστικό του στιγμιότυπου του Paxos που ξεκινάει. Αυτό το στιγμιότυπο είναι ο αριθμός της σύνθεσης της οποίας θέλουμε να αποφασίσουμε. Ο παραπάνω αριθμός κάθε φορά αυξάνει κατά ένα.

Λήψη Αιτήσεων

Όταν ένας διακομιστής λάβει ένα αίτημα προετοιμασίας με ορίσματα vid, n τότε διακρίνουμε τις εξής περιπτώσεις:

1. Αρχικά υπάρχει περίπτωση να είναι αίτημα για κάποιο από τα προηγούμενα στιγμιότυπα του Paxos. Τότε απορρίπτει το αίτημα και στέλνει πίσω την τρέχουσα σύνθεση μαζί με τον αναγνωριστικό της.

2. Στην περίπτωση που το αίτημα είναι για ένα νέο στιγμιότυπο του Paxos τότε πρέπει να αποδεχθεί το αίτημα αν δεν έχει λάβει κάποιο αίτημα με μεγαλύτερο αριθμό. Σε αυτή την περίπτωση ο διακομιστής πρέπει να σταματήσει ότι άλλο κάνει που εξαρτάται από το πρωτόκολλο Paxos, γιατί πρόκειται να υπάρχει αλλαγή στη σύνθεση. Το πετυχαίνει αυτό θέτοντας την μεταβλητή $done$ σε $false$.

3. Σε κάθε άλλη περίπτωση ο διακομιστής απορρίπτει το αίτημα.

Η περιγραφή του Paxos ορίζει πως στην τελευταία περίπτωση το μήνυμα πρέπει να αγνοηθεί. Στο σύστημα μας όμως προτιμήθηκε να απαντάει πίσω με μήνυμα $reject$ ώστε να μην χρειάζεται ο διακομιστής που έστειλε αρχικά το αίτημα να περιμένει να περάσει ο χρόνος αναμονής. Στην περίπτωση όπου ο διακομιστής που έλαβε το μήνυμα απαντάει πίσω με την τρέχουσα σύνθεση, υπάρχει δυνατότητα να αποσταλεί πίσω αυτή τη σύνθεση την οποία πρότεινε ο άλλος διακομιστής. Ένα τέτοιο σενάριο εφαρμόζεται αν χρειάζεται οι διακομιστές να γνωρίζουν το πλήρες ιστορικό των συνθέσεων. Στο δικό μας πρόβλημα κάτι τέτοιο δεν είναι αναγκαίο, και η χρήση του θα καθυστερούσε τις διαδικασίες συμφωνίας, οπότε να προτιμήθηκε η κατευθείαν αποστολή της τρέχουσας σύνθεσης.

5.4.4 Δεύτερη Φάση

Η δεύτερη φάση του Πρωτοκόλλου φαίνεται σε μορφή ψευδοκώδικα στον Αλγόριθμο 5.4. Οι διαφορετικές περιπτώσεις αναλύονται λεπτομερώς στη συνέχεια.

```
if the node gets oldview(vid, v):
  views[vid] = v
  maxvid = vid
  view change
  restart paxos
else if gets reject():
  delay and restart paxos
else if gets prepareres from majority of nodes in views[maxvid]:
  if any prepareres(n_i, v_i) exists such that v_i is not empty:
    v = non-empty value v_i corresponding to highest n_i received
  else leader gets to choose a value:
    v = set of pingable nodes (including self)
  send accept(maxvid+1, myprop, v) to all responders
else:
  delay and restart paxos

if node gets accept(vid, n, v):
  if vid <= maxvid:
    return oldview(maxvid, views[maxvid])
  else if n >= maxprop:
    maxnum = n
    maxval = v
    return acceptres()
  else
    return reject()
```

Αλγόριθμος 5.4: Δεύτερη Φάση του Πρωτοκόλλου Paxos

Αποστολή Αιτήσεων

Σε αυτή τη φάση υποψήφιοι για να ξεκινήσουν αποστολή αιτημάτων τύπου *accept* είναι οι διακομιστές που έστειλαν στην πρώτη φάση. Θα προχωρήσουν στην αποστολή μηνυμάτων τύπου *accept* όσοι έχουν λάβει θετική απάντηση από μία πλειοψηφία. Οι ενέργειες του διακομιστή εξαρτώνται από το τί θα λάβουν. Συγκεκριμένα:

1. Αν λάβει πίσω μία σύνθεση μαζί με το αναγνωριστικό, τότε αυτό σημαίνει πως το σύστημα έχει προχωρήσει σε νέες συνθέσεις. Αποθηκεύει τη σύνθεση την οποία έλαβε και στην συνέχεια ξεκινάει ένα νέο στιγμιότυπο του Paxos.

2. αν ο διακομιστής λάβει ένα *reject*, περιμένει για τυχαίο χρονικό διάστημα και επενακκινεί αργότερα

3. αν λάβει θετική απάντηση από μία πλειοψηφία κόμβων, πρέπει να ξεκινήσει αποστολή μηνυμάτων τύπου *accept*.

4. αν δεν θα λάβει θετική απάντηση από μία πλειοψηφία της τρέχουσας σύνθεσης περιμένει για τυχαίο χρονικό διάστημα και ξεκινάει την διαδικασία αργότερα, ελπίζοντας να ολοκληρωθεί θετικά αυτή την φορά.

Η περίπτωση να λάβει *reject* συνεπάγεται πως στο σύστημα δρομολογείται αλλαγή, και από κάποιον άλλο διακομιστή, ο οποίος έχει μεγαλύτερο προτεινόμενο αριθμό. Για το λόγο αυτό ο συγκεκριμένος διακομιστής περιμένει και επενακκινεί αργότερα. Ο λόγος που γίνεται αυτό είναι ώστε να περιμένει να γίνει η αλλαγή που προτείνει ο άλλος διακομιστής και στη συνέχεια να προτείνει την δική του αλλαγή στη σύνθεση.

Η αποστολή μηνυμάτων τύπου *accept* συμπεριλαμβάνει και τη σύνθεση που θέλει να προτείνει Αν στις απαντήσεις που έλαβε στο προηγούμενο βήμα έλαβε και κάποια σύνθεση τότε διαλέγει να προτείνει αυτή που αντιστοιχεί στο μεγαλύτερο αριθμό πρότασης που έλαβε. Σε διαφορετική περίπτωση πρέπει να αποφασίσει ο ίδιος ποια θα είναι η σύνθεση. Έτσι θα στείλει το αίτημα η σύνθεση να αποτελείται από τους κόμβους που ο ίδιος γνωρίζει ότι είναι ενεργοί. Τελικά ο διακομιστής θα αποστείλει μηνύματα τύπου *accept* με ορίσματα το αναγνωριστικό της σύνθεσης που προτείνει, την ίδια την σύνθεση και τον αριθμό πρότασης, ο οποίος είναι αυτός που είχε προτείνει και στο πρώτο βήμα.

Λήψη Αιτήσεων

Οι ενέργειες στις οποίες προχωράει σε αυτό το βήμα ένας κόμβος ο οποίος λάβει ένα μήνυμα είναι παρόμοιες με αυτές που θα εκτελούσε στην πρώτη περίπτωση.

1. Στην περίπτωση που λάβει ένα μήνυμα για μία σύνθεση παλιότερη της τρέχουσας επιστρέφει πίσω την τρέχουσα σύνθεση μαζί με το αναγνωριστικό της.

2. Αν λάβει μήνυμα όπου ο προτεινόμενος αριθμός είναι μεγαλύτερος από κάθε προτεινόμενο αριθμό που έχει συναντήσει για αυτή τη σύνθεση τότε αποδέχεται αυτή την σύνθεση και απαντάει πίσω θετικά.

3. Αλλιώς απαντάει πίσω με μήνυμα τύπου *reject*.

Η τελευταία περίπτωσή είναι πιθανό να συμβεί αν έχει λάβει στην πρώτη φάση μήνυμα και απαντήσει θετικά, και μέχρι να του στείλει αίτημα τύπου *accept*, έχει λάβει και νέο αίτημα τύπου *prepare* με μεγαλύτερο προτεινόμενο αριθμό και απαντάει και σε αυτό θετικό. Έτσι στην δεύτερη φάση αφού έχει απαντήσει θετικά σε ένα με μεγαλύτερο προτεινόμενο αριθμό, δεν μπορεί να αποδεχτεί αυτή τη σύνθεση.

5.4.5 Τρίτη Φάση

Τέλος στον Αλγόριθμο 5.5 φαίνεται το τελευταίο βήμα του Πρωτοκόλλου.

```
1  if node gets oldview(vid, v):
2    views[vid] = v
3    maxvid = vid
4    view change
5    restart paxos
6  else if gets acceptres from a majority of nodes in views[maxvid]:
7    send decide(maxvid+1, maxval) to all (including self)
8  else:
9    delay and restart paxos
10
11
12 if node gets decide(vid, v):
13   if vid <= maxvid:
14     return oldview(maxvid, views[maxvid])
15   else:
16     primary is lowest-numbered node in v
17     views[vid] = v
18     maxvid = vid
19     view change
20     done = true
```

Αλγόριθμος 5.5: Τρίτη Φάση του Πρωτοκόλλου Paxos

Αποστολή Αιτήσεων

Η τρίτη φάση είναι αυτή στην οποία γίνεται ουσιαστικά η αλλαγή της σύνθεσης, μέχρι τώρα δεν έχουν γίνει αλλαγές, απλώς προσπαθούν οι διακομιστές να συμφωνήσουν σε μία σύνθεση. Ο διακομιστής ο οποίος έχει αποστείλει τα αιτήματα τύπου *accept* στη δεύτερη φάση είναι υπεύθυνος να στείλει και σε αυτή τη φάση. Για να προχωρήσει όμως στην αποστολή μηνυμάτων τύπου *decide* πρέπει να έχει λάβει θετική απάντηση από μία πλειοψηφία διακομιστών. Υπάρχουν όμως και σε αυτή τη φάση περιπτώσεις όπου δεν μπορεί να συνεχίσει.

Αν λάβει κάποια υπάρχουσα σύνθεση συνεπάγεται πως το σύστημα προχώρησε σε νέα σύνθεση μέχρι ο συγκεκριμένος διακομιστής να βρεθεί σε θέση να προχωρήσει. Το γεγονός πως βρίσκεται σε αυτή τη φάση σημαίνει πως είχε λάβει θετική απάντηση από το πρώτο βήμα και πως εκείνη τη στιγμή δεν υπήρχε άλλη σύνθεση. Όμως μέχρι να στείλει τα μηνύματα της δεύτερης φάσης έχει ξεκινήσει και ένας άλλος την διαδικασία. Ο άλλος μπορεί να στείλει μεγαλύτερο προτεινόμενο αριθμό, οπότε να έλαβε θετική απάντηση στο πρώτο βήμα, και να προχώρησε την διαδικασία και να συμφωνήθηκε μία άλλη σύνθεση. Έτσι ο τρέχων διακομιστής προτείνει κάτι που πλέον είναι παλιό.

Συνοψίζοντας οι ενέργειες του διακομιστή, ανάλογα το τι έλαβε είναι:

1. αν λάβει μία κάποια υπάρχουσα σύνθεση τότε πρέπει να αποθηκεύσει αυτή την σύνθεση και να επανεκκινήσει την διαδικασία, αντίστοιχα όπως έκανε και στην δεύτερη φάση.

2. αν λάβει θετική απάντηση από μία πλειοψηφία τότε στέλνει τα μηνύματα τύπου *decide*

στο οποία συμπεριλαμβάνει τη σύνθεση η οποία αποφασίστηκε μαζί με το μοναδικό αναγνωριστικό της.

3. Σε διαφορετική περίπτωση, όταν δηλαδή λάβει θετικές απαντήσεις αλλά αυτές δεν αποτελούν την πλειοψηφία, τότε πρέπει να ξεκινήσει το πρωτόκολλο αργότερα με την ελπίδα να πετύχει αυτή τη φορά

Λήψη Αιτήσεων

Οι ενέργειες που προχωράει και σε αυτή την περίπτωση κάποιος που θα λάβει ένα αίτημα είναι παρόμοιες με τις προηγούμενες φάσεις.

1. Αρχικά αν η αίτηση έχει αριθμό σύνθεσης μικρότερο ή ίσο του τρέχοντος τότε δεν την αποδέχεται και στέλνει πίσω την τελευταία σύνθεση που γνωρίζει μαζί με το αναγνωριστικό της. Η περίπτωση αυτή καλύπτει καθυστερημένα μηνύματα, καθώς και γρήγορες αλλαγές δύο συνθέσεων.

2. Σε αντίθετη περίπτωση πρέπει να δεχτεί την αλλαγή. Όποτε πρέπει να θέσει την μεταβλητή *done* στη λογική τιμή *true*, να κάνει αλλαγή της σύνθεσης και τέλος πρέπει να αλλάξει τον επικεφαλής. Η επιλογή του επικεφαλής πρέπει να είναι ίδια για όλους του κόμβους. Μία επιλογή θα ήταν εκτός από τη σύνθεση να αποστέλλεται και ο επικεφαλής μαζί με τη σύνθεση που προτείνεται, κάτι το οποίο αποτελεί πλεονασμό. Μία ικανή συνθήκη είναι ο επικεφαλής να επιλέγεται ντετερμινιστικά από το σύνολο τον διακομιστών που αποτελούν τη σύνθεση. Αν η επιλογή γίνεται ντετερμινιστικά τότε όλοι θα γνωρίζουν τον ίδιο επικεφαλής αφού γνωρίζουν την ίδια σύνθεση. Έτσι ο επικεφαλής επιλέγεται αυτός που είναι ο λεξικογραφικά μικρότερος.

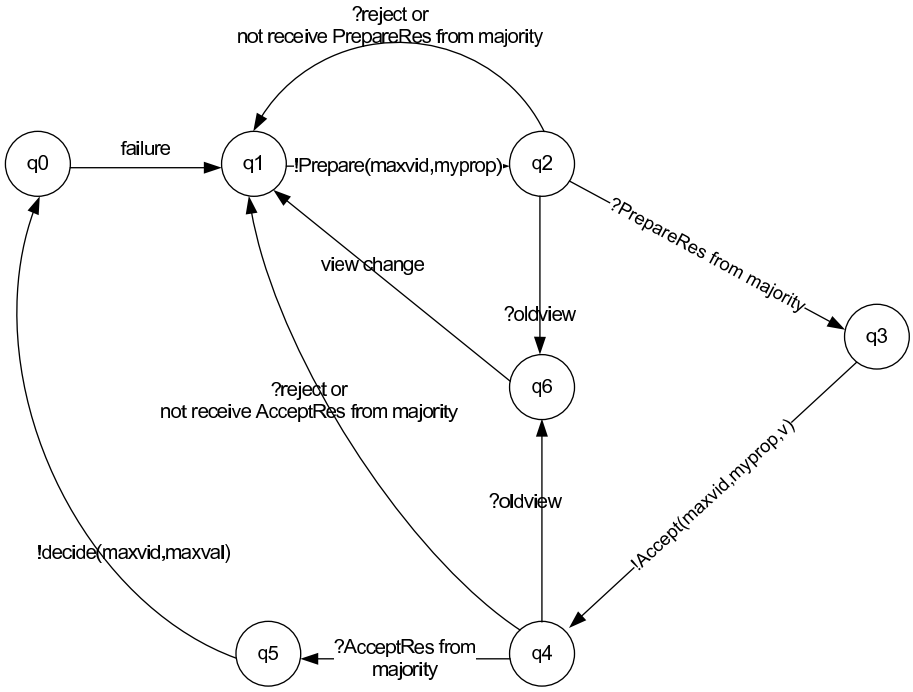
5.4.6 Διαγράμματα Καταστάσεων

Σε αυτή την ενότητα περιγράφονται τα διαγράμματα καταστάσεων τόσο του διακομιστή που ανιχνεύει την αλλαγή και ξεκινάει το στιγμιότυπου του αλγορίθμου συμφωνίας Paxos όσο και του διακομιστή που λαμβάνει μηνύματα από αυτόν που ξεκίνησε το στιγμιότυπο

Αποστολέας μηνυμάτων του Paxos

Στο Σχήμα 5.1 φαίνεται το διάγραμμα καταστάσεων για την λειτουργία ενός διακομιστή που ανίχνευσε μία αλλαγή στη σύνθεση των μελών και χρειάστηκε να ξεκινήσει ένα στιγμιότυπο του Paxos.

Η κατάσταση q_0 είναι η αρχική κατάσταση, όπου το σύστημα λειτουργεί κανονικά χωρίς αλλαγές. Μόλις ανιχνευθεί μια αλλαγή μεταβαίνει στην κατάσταση q_1 απ όπου ξεκινάει την αποστολή μηνυμάτων του αλγορίθμου συμφωνίας Paxos. Από την q_1 στέλνει μηνύματα prepare και προχωράει στην κατάσταση q_2. Από εκεί αν λάβει θετική απάντηση από μία πλειοψηφία τότε πηγαίνει στην κατάσταση q_3 για να ξεκινήσει την αποστολή μηνυμάτων accept και να καταλήξει στην κατάσταση q_4. Αν και πάλι λάβει θετική απάντηση από μία



Σχήμα 5.1: Διάγραμμα Καταστάσεων του Αποστολέα μηνυμάτων του Paxos

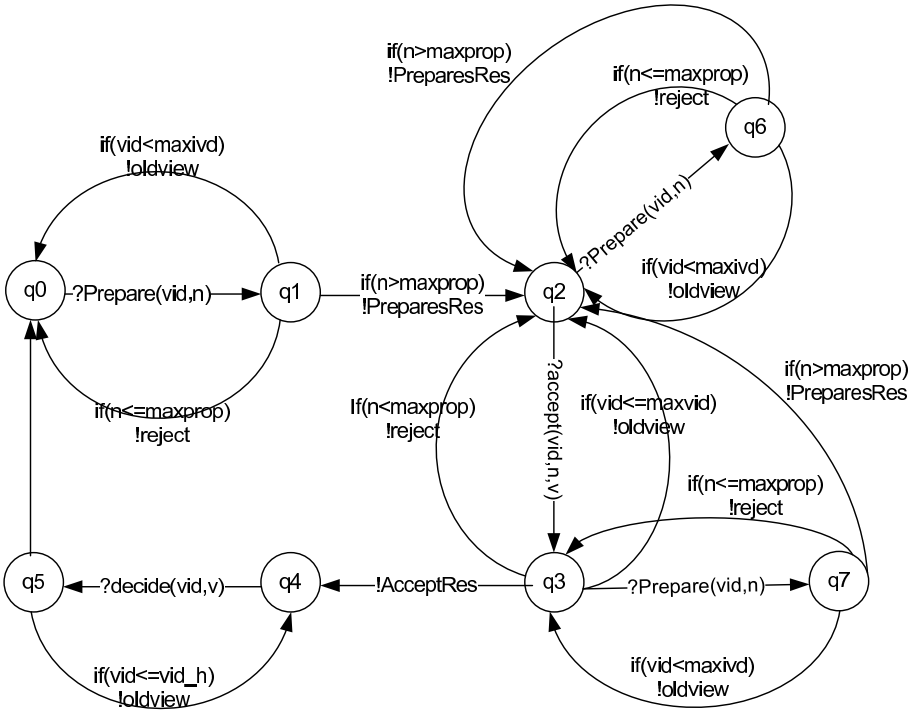
πλειοψηφία τότε μεταβαίνει στην κατάσταση q_5 για να στείλει τα τελευταία μηνύματα, τα decide και να γυρίσει σε σταθερή κατάσταση. Αυτή είναι η διαδικασία ώστε να συμφωνηθεί η νέα σύνθεση.

Υπάρχουν όμως περιπτώσεις που δεν λαμβάνει το απαιτούμενο πλήθος θετικών απαντήσεων ή λαμβάνει απαντήσει reject ή παλιές όψεις. Στην περίπτωση που λάβει κάποια προηγούμενη όψη σημαίνει πως προτείνει κάτι το οποίο πλέον είναι παλιό. Για το λόγο αυτό κάνει αλλαγή της όψης και ξαναξεκινάει ένα νέο στιγμιότυπο. Η λήψη προηγούμενης όψης οδηγεί στην κατάσταση q_6, από τις καταστάσεις q_2, q_4, όπου έχει στείλει τα μηνύματα και περιμένει απαντήσει Από την q_6 κάνει αλλαγή της όψης και πηγαίνει πίσω στην κατάσταση q_1 για να ξεκινήσει από την αρχή.

Τέλος, στις καταστάσεις q_2, q_4 είναι δυνατόν να λάβει ένα μήνυμα reject ή να μην λάβει θετική απάντηση από μία πλειοψηφία. Σε αυτές τις δύο περιπτώσεις πηγαίνει κατευθείαν στην κατάσταση q_1, ώστε να ξεκινήσει από την αρχή.

Αποδέκτης Αιτημάτων του Paxos

Σε αυτή την ενότητα περιγράφεται ένας διακομιστής που λαμβάνει αιτήματα από κάποιον άλλο διακομιστή ώστε να συμφωνηθεί μία νέα σύνθεση. Ένας διακομιστής δεν είναι αναγκαστικό να λειτουργεί μόνο ως αποστολέας ή μόνο ως παραλήπτης. Μπορεί παράλληλα να προτείνει νέα σύνθεση αλλά και να ακούει σε προτάσεις άλλων. Το διάγραμμα καταστάσεων του, ως παραλήπτης φαίνεται στο Σχήμα 5.2. Αρχικά ο διακομιστής βρίσκεται στην κατάσταση q_0, όπου και λειτουργεί κανονικά. Αν λάβει ένα αίτημα prepare, τότε μεταβαίνει στην κατάσταση q_1, και στην περίπτωση που η πρόταση που έλαβε έχει μεγαλύτερο αριθμό



Σχήμα 5.2: Διάγραμμα Καταστάσεων του Αποδέκτη μηνυμάτων του Paxos

από όλες όσες έχει δει μέχρι τώρα απαντάει θετικά πίσω και πηγαίνει στην κατάσταση q_2. Τώρα πλέον δεν μπορεί να δεχθεί αιτήματα της μηχανής καταστάσεων πολλαπλών αντιγράφων. Για να επιστρέψει σε σταθερή κατάσταση πρέπει να συμφωνηθεί μία νέα όψη. Έτσι από την q_2 περιμένει να λάβει μήνυμα τύπου accept για να μεταβεί στην κατάσταση q_3, και στην περίπτωση που η πρόταση έχει πάλι αριθμό μεγαλύτερο από οποιονδήποτε έχει συναντήσει τότε απαντάει θετικά και πηγαίνει στην κατάσταση q_4. Εκεί μόλις λάβει και το μήνυμα decide το οποίο όμως να μην αναφέρεται σε προηγούμενη σύνθεση, είναι πλέον σε σταθερή κατάσταση και έχει συμμετέχει στη συμφωνία για τη νέα σύνθεση

Υπάρχει περίπτωση να λάβει μήνυμα είτε με προηγούμενη σύνθεση είτε με αριθμό μικρότερο από το μέγιστο που έχει συναντήσει. Αυτές είναι οι καταστάσεις q_3, q_6, όπου είτε απαντάει με προηγούμενη σύνθεση είτε απαντάει απορρίπτοντας την αλλαγή. Σε κάθε περίπτωση γυρνάει ακριβώς στην προηγούμενη κατάσταση. Υπάρχει πιθανότητα ενώ βρίσκεται σε μία κατάσταση να λάβει και δεύτερη φορά μήνυμα prepare, τότε αν αυτό έχει αριθμό μεγαλύτερο από οποιονδήποτε έχει συναντήσει μέχρι τώρα τότε μεταβαίνει στην κατάσταση q_2.

5.5 Περίληψη

Σε αυτό το Κεφάλαιο περιγράφηκε το πρωτόκολλο Paxos και η υλοποίησή του. Επιπλέον επισυμάνθηκαν μερικές αποφάσεις που χρειάστηκε να γίνουν στην πορεία όσων αφορά την υλοποίηση.

Περιγράφηκε ο τρόπος με τον οποίο οι διακομιστές αρχείων προσαρμόζουν τον πίνακα

εξαγωγής. Με τον προσαρμοσμένο πίνακα εξαγωγής προσφέρουν το πλήθος των συστημάτων αρχείων στους πελάτες τους χρησιμοποιώντας το σύστημα NFSv4 και την επιλογή της ανακατεύθυνσης.

Κεφαλαιο 6

Πειραματικα Αποτελεσματα

6.1 Περιβάλλον Πειραμάτων

6.2 Τοπικοί Διακομιστές

6.3 Απομακρυσμένοι Διακομιστές

Σε αυτό το κεφάλαιο γίνεται μία πειραματική μελέτη της υλοποίησης. Εξετάζεται η δυνατότητα κλιμάκωσης του πρωτοκόλλου Paxos σύμφωνα με διάφορες παραμέτρους. Ακόμη εξετάζεται και ο χρόνος ανάκαμψης από σφάλματα. Η μελέτη περιλαμβάνει δύο περιβάλλοντα, το ένα για ένα τοπικό οργανισμό και το δεύτερο για συνεργασίες οργανισμών μέσω του διαδικτύου.

6.1 Περιβάλλον Πειραμάτων

Τα πειράματα έγιναν σε υπολογιστές με τον τετραπύρηνο επεξεργαστή quad-core 2.33GHz, και μνήμη 2GB και λειτουργικό σύστημα GNU/Linux με έκδοση πυρήνα 2.6.18. Όταν δύο διακομιστές θεωρούμε πως είναι στο ίδιο ερευνητικό κέντρο τότε τρέχουν στο ίδιο μηχάνημα. Στην περίπτωση που θεωρούμε διαφορετικούς οργανισμούς τότε οι διακομιστές βρίσκονται σε διαφορετικούς υπολογιστές, και χρήση χρήση του πακέτου iproute ώστε να προσομοιωθεί ο χρόνος διάδοσης. Η διασύνδεση των υπολογιστών γίνεται με ethernet ταχύτητας 1Gbit.

Επιλέχθηκαν οι παράμετροι έτσι ώστε να υπάρχει μέγιστος χρόνος αίτησης(timeout) μεταξύ των διακομιστών 500ms, ενώ ο πελάτης θεωρεί πως ένα πακέτο χάθηκε μετά από 2sec. Το νήμα που κάνει εκκαθάριση στον επικεφαλής των προηγούμενων αιτήσεων τρέχει κάθε 1sec ενώ κρατάμε αιτήσεις για 10sec. Μία άλλη παράμετρος του συστήματος είναι ο χρόνος που θα περιμένει ο πελάτης πριν επαναλάβει την αίτηση στην περίπτωση που το σύστημα είναι απασχολημένο Αυτό επιλέχθηκε να είναι ένας τυχαίος αριθμός στο διάστημα $[0, 5]sec$.

Στις επόμενες ενότητες περιγράφονται πειράματα όπου οι αιτήσεις φτάνουν με ένα ρυθμό, αυτός είναι ένας σταθερός ρυθμός. Προτιμήθηκε να μην χρησιμοποιηθεί ρυθμός αφίξεως αιτημάτων με κατανομή Poisson ώστε να είναι ευκολότερη η ερμηνεία των αποτελεσμάτων.

6.2 Τοπικοί Διακομιστές

Σε αυτή τη σειρά πειραμάτων υποθέτουμε πώς όλοι οι διακομιστές βρίσκεται στον ίδιο κέντρο οπότε ο χρόνος διάδοσης από το ένα μηχάνημα στο άλλο είναι αμελητέος. Έτσι έχουμε βάλει όλους τους διακομιστές σε ένα κόμβο αλλά να ακούν σε διαφορετικές θύρες. Αυτή η παραδοχή δεν είναι περιοριστική καθώς οι κόμβοι σε ένα υπολογιστικό κέντρο συνδέονται στη χειρότερη των περιπτώσεων με ethernet 1Gbit. Ακόμη, από τους πόρους του συστήματος κανείς δεν φτάνει στα όρια του ώστε να δημιουργεί συμφόρηση και να μειώνει τη ρυθμαπόδοση των πειραμάτων. Στη συνέχεια υπάρχουν δύο κατηγορίες πειραμάτων. Αρχικά έχουμε τα πειράματα κατά τα οποία οι διακομιστές της υπηρεσίας χώρου ονομάτων λειτουργούν κανονικά και δεν υπάρχουν αποτυχίες, ενώ στη δεύτερη σειρά πειραμάτων υποθέτουμε ύπαρξη αποτυχιών.

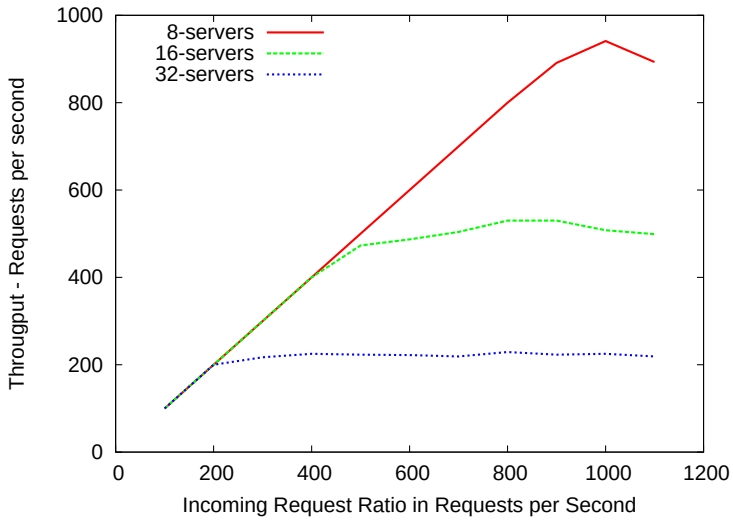
6.2.1 Λειτουργία Χωρίς Αποτυχίες

Στο Σχήμα 6.1 φαίνεται η ρυθμαπόδοση της υπηρεσίας χώρου ονομάτων με βάση δύο παραμέτρους, τον αριθμό των διακομιστών και τον ρυθμό άφιξης των αιτήσεων. Τα αιτήματα που φτάνουν είναι όλα για ανάγνωση καθώς το σύστημα στην πλειοψηφία του θα δέχεται αιτήματα για ανάγνωση. Επίσης το μέγεθος του πίνακα εξαγόμενων συστημάτων είναι μηδενικό. Η μεταφορά των δεδομένων είναι αμελητέα καθώς θεωρούμε δίκτυο 1Gbit. Το συγκεκριμένο πείραμα είναι αποτέλεσμα μία εκτέλεσης επειδή η διασπορά είναι πολύ μικρή και δεν χρειάζεται να επαναληφθεί.

Στο συγκεκριμένο πείραμα παρατηρείται η ακόλουθη συμπεριφορά. Όσο το σύστημα δεν έχει φτάσει στα όριά του, η ρυθμαπόδοση είναι όσο και ο ρυθμός άφιξης των αιτήσεων. Όταν το σύστημα φτάσει στα όρια του τότε η ρυθμαπόδοση δεν επηρεάζεται σε σχέση με τον ρυθμό άφιξης των αιτήσεων. Για κάθε μήνυμα που φτάνει σε ένα διακομιστή δημιουργείται ένα νέο νήμα. Αυτά τα νήματα περιμένουν μέχρι να εξυπηρετηθούν προηγούμενες αιτήσεις. Οπότε σε κάθε περίπτωση θα είναι σε αποκλεισμό και δεν θα επηρεάζουν τη λειτουργία.

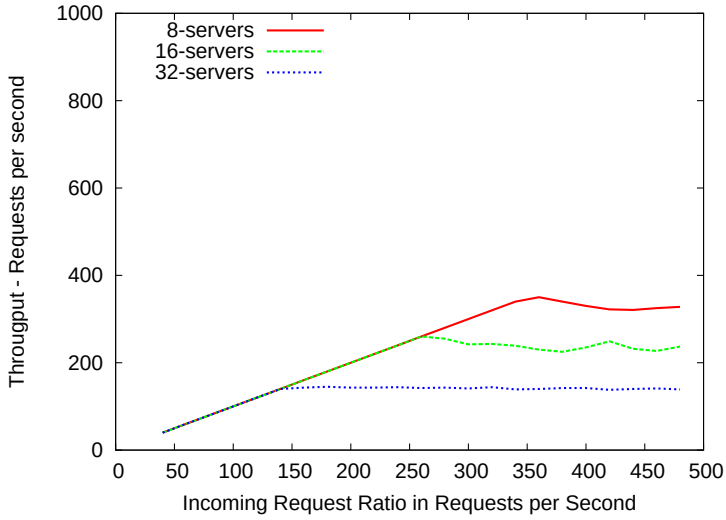
Αυτό που περιορίζει τη ρυθμαπόδοση είναι η επικοινωνία των διακομιστών. Ο χρόνος που χρειάζεται να στείλει το αίτημα στους εφεδρικούς για να το εκτελέσουν και να λάβει πίσω απάντηση. Αυτό είναι μια επιθυμητή συμπεριφορά γιατί ακόμη και πολλές αιτήσεις να είναι προς εξυπηρέτηση τότε το σύστημα δεν καταρρέει. Για να καταρρεύσει πρέπει να γεμίσει η ενδιάμεση μνήμη(buffer) του πρωτοκόλλου στον πυρήνα.

Τέλος παρατηρούμε πως όσο αυξάνει ο αριθμός των διακομιστών τόσο μειώνεται η απόδοση. Συγκεκριμένα, όταν διπλασιάζουμε το πλήθος, υποδιπλασιάζεται ο ρυθμαπόδοση. Αυτό συμβαίνει γιατί διπλασιάζοντας το πλήθος των διακομιστών διπλασιάζεται ο αριθμός



Σχήμα 6.1: Ρυθμαπόδοση Συστήματος

των μηνυμάτων που πρέπει να προωθηθούν στους εφεδρικούς ώστε να εξυπηρετηθεί τελικά το αίτημα.

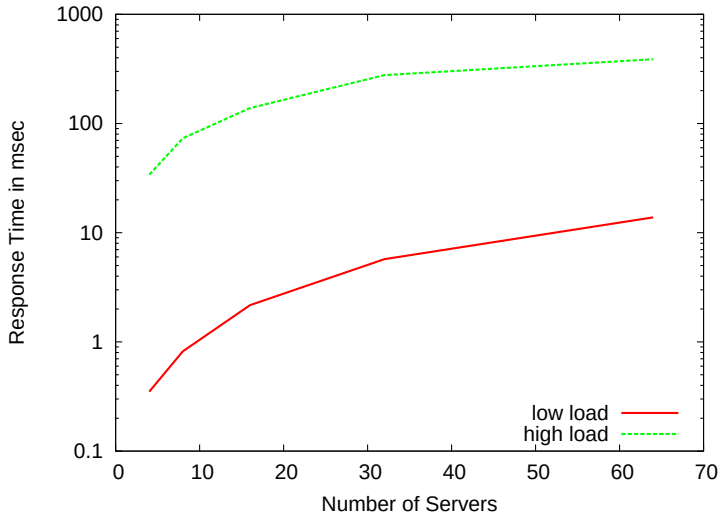


Σχήμα 6.2: Ρυθμαπόδοση Συστήματος με αποθήκευση σε διάνυσμα των παλιών αιτήσεων

Σε αυτά τα πειράματα που ο ρυθμός εισαγωγής αιτήσεων στο σύστημα είναι μεγάλος, το μεγαλύτερο διάστημα καταναλώνεται ώστε να έχουμε έλεγχο για διπλότυπα μηνύματα. Αυτό φαίνεται ξεκάθαρα αν συγκρίνουμε το Σχήμα 6.1 με το Σχήμα 6.2. Το δεύτερο είναι μία αρχική υλοποίηση όπου τα διπλότυπα αποθηκεύονται σε ένα διάνυσμα (vector) της *STL* βιβλιοθήκης της *C++*. Με την αλλαγή της αποθήκευσης σε *Hash Map* έχουμε διπλασιασμό της ρυθμαπόδοσης.

Η παραπάνω συμπεριφορά παρατηρείται γιατί μεγαλώνει πολύ ο όγκος της συγκεκριμένης πληροφορίας. Αν το σύστημα εξυπηρετεί 500 αιτήσεις το δευτερόλεπτο και κρατάει ιστορικά για δέκα δευτερόλεπτα τότε ανά πάσα στιγμή υπάρχει 5000 αιτήσεις. Οπότε κάθε

φορά που φτάνει μία αίτηση πρέπει να διατρέξει αυτή τι δομή και να ελέγχει όλες τις παλιές. Όμως κάτι τέτοιο απαιτεί 2500000 ελέγχους το δευτερόλεπτο.

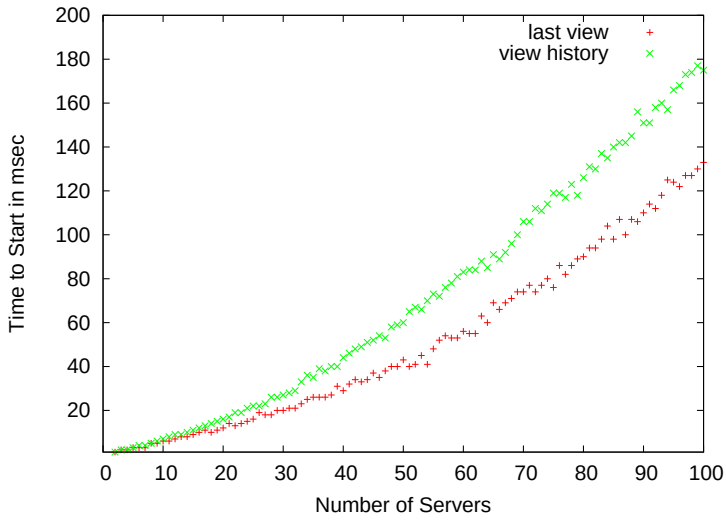


Σχήμα 6.3: Χρόνος απόκρισης Συστήματος

Στο Σχήμα 6.3 φαίνεται ο μέσος χρόνος εξυπηρέτησης των αιτήσεων σε σχέση με το πλήθος των διακομιστών που είναι μέλη της τρέχουσας σύνθεση. Ο μέσος χρόνος εξυπηρέτησης εξαρτάται από το πόσο φορτωμένο είναι το σύστημα, οπότε έχουν συμπεριληφθεί δύο γραφικές. Μία για χαμηλό φόρτο και άλλη μία για την περίπτωση υψηλού φορτίου στο σύστημα. Με τον όρο υψηλό φορτίο εννοείται φορτίο στο 80% της μέγιστης τιμής ρυθμαπόδοσης αιτήσεων που μπορεί να εξυπηρετήσει. Οι τιμές είναι αποτέλεσμα όλου του πλήθους των μετρήσεων. Έγιναν συνολικά 15000 αιτήσεις στην περίπτωση του υψηλού φορτίου. Ο αριθμός είναι τόσο μεγάλος γιατί αφενός χρειάστηκε μεγάλο πλήθος από αιτήσεις ώστε να φορτωθεί και αφετέρου ώστε η τυπική απόκλιση να μικρύνει. Έτσι η τυπική απόκλιση είναι στο 4% της μέσης τιμής. Στην περίπτωση του χαμηλού φορτίου έγιναν 10 μετρήσεις και η τυπική απόκλιση είναι από 2% μέχρι και 7%. Ο άξονας y εμφανίζεται σε λογαριθμική κλίμακα.

Όσα αυξάνει το πλήθος των διακομιστών αυξάνει εκθετικά και ο χρόνος απόκρισης. Παρατηρείται όμως μεγάλη καθυστέρηση στην εξυπηρέτηση στην περίπτωση όπου στο σύστημα υπάρχει υψηλό φορτίο. Όταν υπάρχουν 16 διακομιστές και δεν έχει καθόλου φορτίο ο χρόνος απόκρισης είναι γύρω στα 2ms ενώ αν είναι φορτωμένο έχει χρόνο απόκρισης 138ms.

Στο Σχήμα 6.4 εξετάζεται ο χρόνος που χρειάζεται το σύστημα στην περίπτωση εισαγωγής ενός διακομιστή. Στον άξονα x φαίνεται το πλήθος των διακομιστών που ήδη υπάρχουν και στον άξονά y ο χρόνος ώστε να είναι πάλι σε σταθερή κατάσταση το σύστημα. Στην Ενότητα 5.4.3 επισημάνθηκε πως έχουμε δύο επιλογές όσον αφορά το ιστορικό των συνθέσεων. Στην περίπτωση που θέλουμε να έχουμε πλήρης εικόνα πρέπει κάθε καινούργιος διακομιστής να μαθαίνει όλες τις προηγούμενες κατά της εκκίνηση του. Σε διαφορετική περίπτωση, αρκεί να μάθει την τελευταία. Στο Σχήμα 6.4 φαίνεται ο χρόνος που χρειάζεται



Σχήμα 6.4: Χρόνος Εκκίνησης Διακομιστή

για εκκίνηση σε κάθε περίπτωση.

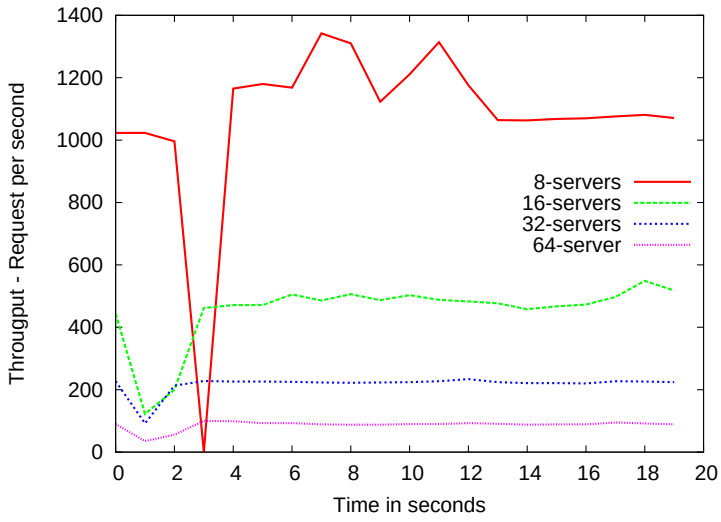
Επειδή τα πειράματα μεταξύ τους είχαν μεγάλες τυπικές αποκλίσεις, χρειάστηκε το κάθε πείραμα να εκτελεστεί 5 φορές και να πάρουμε το μέσο όρο των πειραμάτων. Στην περίπτωση που κρατάμε ιστορικό των συνθέσεων η τυπική απόκλιση είναι από 7% μέχρι 11% της μέσης τιμής, ενώ στην περίπτωση που δεν κρατάμε ιστορικό η απόκλιση είναι μεγαλύτερη όσο έχουμε μικρό πλήθος διακομιστών και μικραίνει όσο μεγαλώνει ο αριθμός των διακομιστών. Συγκεκριμένα στην περίπτωση των τεσσάρων διακομιστών που παρουσιάζεται η μέγιστη απόκλιση είναι στο 17% ενώ από τους πενήντα διακομιστές και μετά κυμαίνεται από 2% μέχρι 7%.

Όπως και σε προηγούμενα πειράματα έτσι και σε αυτό όσο αυξάνει το πλήθος των διακομιστών τόσο αυξάνει εκθετικά ο χρόνος εκκίνησης ενός καινούργιου. Επίσης παρατηρούμε πως στο σύστημα αν δεν θέλουμε να έχουμε ιστορικό τότε ο κάθε χρόνος είναι πολύ καλύτερος και συγκεκριμένα παρουσιάζει μία βελτίωση της τάξης του 30%.

6.2.2 Περιπτώσεις Αποτυχιών

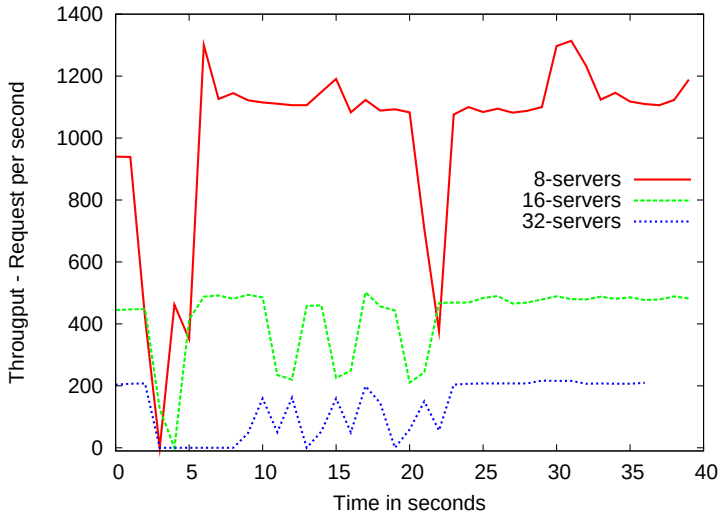
Σε αυτή την υποενότητα έχουμε μία σειρά πειραμάτων όπου παρουσιάζονται αποτυχίες στους διακομιστές τις υπηρεσίας χώρου ονομάτων. Συγκεκριμένα υπάρχουν δύο περιπτώσεις, ο διακομιστής που απέτυχε να είναι είτε εφεδρικός είτε επικεφαλής. Τα πειράματα υποθέτουν μία σειρά αιτήσεων που φτάνει στο σύστημα ικανή ώστε να το φορτώσει στο μέγιστο. Ο άξονας των x είναι ο χρόνος και στον άξονα των y έχουμε τη ρυθμαπόδοση που παρατηρείται, ενώ έχει ληφθεί και ως παράμετρος και το πλήθος των διακομιστών.

Στο Σχήμα 6.5 βλέπουμε την περίπτωση αποτυχίας ενός εφεδρικού διακομιστή. Αυτή την αποτυχία την αντιλαμβάνεται ο επικεφαλής όταν προσπαθεί να του στείλει ένα νέο αίτημα προς εξυπηρέτηση. Τότε σταματάει την λειτουργία εξυπηρέτησης αιτημάτων και ξεκινάει ένα στιγμιότυπο του Paxos. Από την στιγμή που κανείς άλλος δεν ανιχνεύει την αποτυχία είναι ο μόνος που προτείνει μία νέα σύνθεση. Αρα αυτό που χρειάζεται



Σχήμα 6.5: Αποτυχία Εφεδρικού Διακομιστή

είναι να επικοινωνήσει με όλους τους διακομιστές τρεις φορές, μία για κάθε φάση, και να αποφασίσουν να αφαιρέσουν τον προηγούμενο από τη σύνθεση. Παρατηρείται μία στιγμιαία πτώση της ρυθμαπόδοσης, μέχρι το σύστημα να έρθει πάλι σε σταθερή κατάσταση.



Σχήμα 6.6: Αποτυχία Επικεφαλής Διακομιστή

Στο Σχήμα 6.6 φαίνεται η περίπτωση αποτυχίας του επικεφαλής. Αυτή την αποτυχία την αντιλαμβάνονται όλοι οι εφεδρικοί όταν αντιληφθούν πως αποτυγχάνει το μήνυμα τύπου heartbeat που στέλνουν κάθε ένα δευτερόλεπτο. Έτσι προσπαθούν όλοι να προτείνουν μία νέα σύνθεση, και τα μηνύματα που ανταλλάσσονται είναι πολύ περισσότερα μέχρι να συμφωνηθεί η νέα σύνθεση.

Η πιο ξεκάθαρη γραφική παράσταση του Σχήματος 6.6 είναι στην περίπτωση όπου υπάρχουν οχτώ διακομιστές χώρου ονομάτων. Στην πρώτη πτώση της απόδοσης φαίνεται η περίπτωση που ανιχνεύθηκε η αποτυχία του επικεφαλής και συμφωνήθηκε η νέα σύνθεση.

Αυτό που δεν είναι αμέσως ξεκάθαρο είναι η δεύτερη πτώση που φαίνεται στο Σχήμα 6.6. Το πρωτόκολλο Paxos προβλέπει πως κάθε φορά που ένας διακομιστής λάβει μία απάντηση reject τότε κοιμάται για τυχαίο χρονικό διάστημα και ξαναπροσπαθεί στη συνέχεια. Η συμπεριφορά που παρατηρείται είναι αποτέλεσμα αυτής της συμπεριφοράς του Πρωτοκόλλου.

Συγκεκριμένα αρχικά προσπάθησε ένας διακομιστής να προτείνει σύνθεση χωρίς τον επικεφαλής και πέρασε το μήνυμα τύπου *prepare*. Όμως μέχρι να στείλει τα μήνυμα τύπου *accept* κάποιος άλλος διακομιστής προσπάθησε να ξεκινήσει στιγμιότυπο για νέα σύνθεση. Ο δεύτερος διάλεξε μεγαλύτερο προτεινόμενο αριθμό οπότε το δέχτηκαν κάποιοι διακομιστές. Όταν όμως ο πρώτος προσπάθησε να στείλει τα μηνύματα τύπου accept τότε έλαβε μήνυμα *reject*. Σύμφωνα με τον αλγόριθμο περίμενε τυχαίο χρονικό διάστημα και μετά ξαναπροσπάθησε να προτείνει την αλλαγή. Στο ενδιάμεσο οι υπόλοιποι διακομιστές είχαν συμφωνήσει σε κάποια σύνθεση, έτσι αποφασίζει να προτείνει μία με μεγαλύτερο αναγνωριστικό. Το αποτέλεσμα της παραπάνω διαδικασία είναι η δεύτερη πτώση της ρυθμαπόδοσης.

Με βάση την παραπάνω παρατήρηση είναι εμφανές τι γίνεται και στις περιπτώσεις που έχουμε περισσότερους διακομιστές. Συγκεκριμένα ξεκινάνε να προτείνουν νέα σύνθεση σχεδόν όλοι και είναι πολύ πιθανόν αρκετοί να λάβουν μηνύματα reject, έτσι κοιμούνται και ξυπνάνε αργότερα. Όταν όμως θα ξυπνήσουν πάλι μπορεί να λάβουν μηνύματα reject. Αυτό θα συνεχίζεται συνέχεια μέχρι να μην λάβει κανείς μήνυμα reject. Έτσι παρουσιάζεται η περίπτωση, μίας σύνθεση 64 διακομιστών, η οποία για να ανάκαμψη από την αποτυχία ενός επικεφαλής χρειάζεται περίπου 20 δευτερόλεπτα. Όταν παρουσιάζεται το πρόβλημα εμφανίζεται και ένα διάστημα περίπου 8 δευτερολέπτων όπου δεν μπορεί να εξυπηρετήσει καθόλου αιτήσεις.

6.3 Απομακρυσμένοι Διακομιστές

Σε αυτή την ενότητα εξετάζεται η συμπεριφορά του συστήματος στην περίπτωση που οι διακομιστές του χώρου ονομάτων είναι διασκορπισμένοι σε διαφορετικούς οργανισμούς. Σε αυτή την περίπτωση ο χρόνος διάδοσης μεταξύ δύο διαφορετικών οργανισμών δεν είναι αμελητέος αλλά εξαρτάται ως επί το πλείστον από την απόσταση που υπάρχει ανάμεσα στους οργανισμούς.

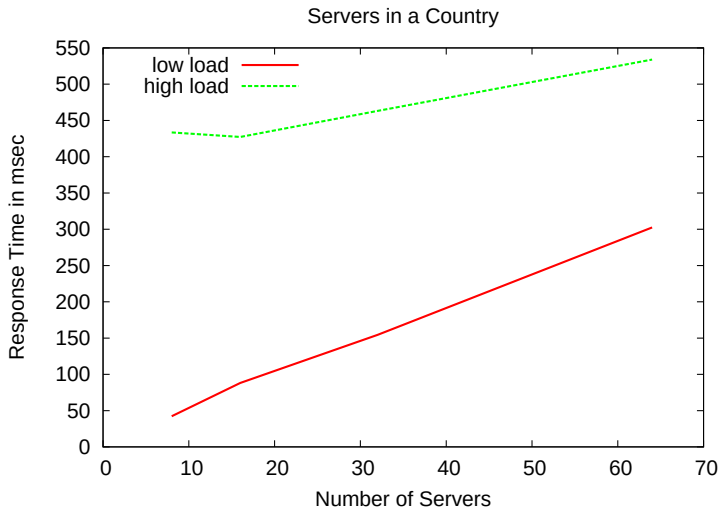
Υποθέτουμε ένα σενάριο όπου υπάρχουν τρεις οργανισμοί οι οποίοι διαθέτουν διακομιστές στην υπηρεσία χώρου ονομάτων. Αυτοί είναι εντός των ελληνικών συνόρων. Συγκεκριμένα υπάρχει ένας οργανισμός στην Αττική με το 50% των διακομιστών, ένας άλλος στη Θεσσαλονίκη με το 37,5% των διακομιστών και ένας τελευταίος με το 12,5% στα Ιωάννινα. Τα ποσοστά επιλέχθηκαν έτσι, ώστε να βγαίνουν ακέραιοι όταν το πλήθος των διακομιστών είναι δύναμη του δύο. Ο χρόνος διάδοσης από τα Ιωάννινα προς τους άλλους δύο οργανισμούς ορίστηκε στα 7ms ενώ μεταξύ Αθήνας - Θεσσαλονίκης στα 5ms.

Ένα δεύτερο σενάριο περιλαμβάνει διακομιστές που είναι τοποθετημένοι εντός μίας Ηπείρου. Πάλι υποθέτουμε τρεις οργανισμούς. Ο πιο απομακρυσμένος απέχει από τους άλλους δύο χρόνο ίσο με 60ms. Ένας τέτοιος θα μπορούσε να είναι ένας που βρίσκεται στην Ελλάδα ενώ οι άλλοι δύο βρίσκονται σε χώρες όπως Γαλλία, Γερμανία. Οι δύο τελευταίοι

έχουν μεταξύ τους χρόνο διάδοσης ίσο με 25ms. Πάλι η ποσόστωση των διακομιστών στους οργανισμούς γίνεται όπως πριν, 50% ,37.5% και 12,5% με την χώρα που απέχει 60ms να παίρνει το 12,5% των διακομιστών.

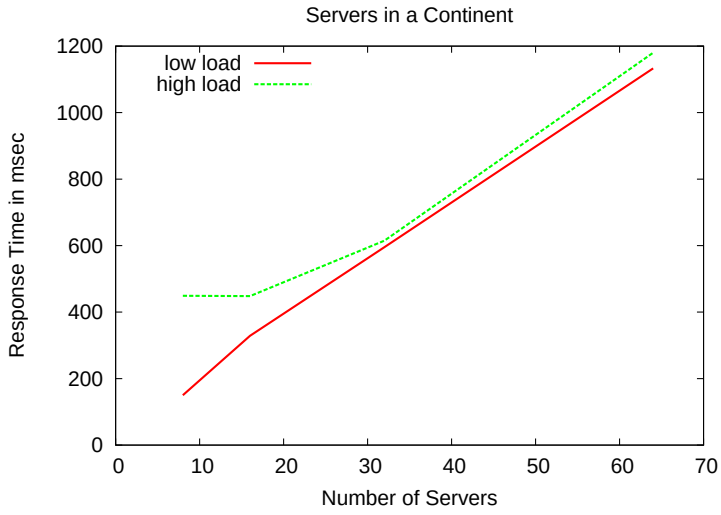
Για να εφαρμοστούν τα παραπάνω σενάρια χρειάστηκε να τροποποιηθούν οι μέγιστοι χρόνοι αναμονής (timeout) ώστε να προσαρμοστούν στη συνθήκες επικοινωνίας μέσω διαδικτύου. Οπότε ο μέγιστος χρόνος αναμονής ανάμεσα στους διακομιστές έγινε 2sec. Μπορεί να τετραπλασιάστηκε σε σχέση με πριν άλλα σε αναλογία με το χρόνο διάδοσης είναι πολύ μικρότερος.

Στο πείραμα που φαίνεται στο Σχήμα 6.7 εξετάζεται ο χρόνος απόκρισης στην περίπτωση που έχουμε διακομιστές οι οποίοι βρίσκονται όντως της Ελλάδος όπως περιγράφηκε. Όπως και στην περίπτωση τον τοπικών έτσι και τώρα έγιναν πειράματα τόσο για περίπτωση χαμηλού όσο και για περίπτωση υψηλού φορτίου. Γενικώς παρατηρείται πως καθώς αυξάνει το πλήθος των διακομιστών αυξάνει γραμμικά και ο χρόνος απόκρισης στην περίπτωση χαμηλού φορτίου. Στην περίπτωση υψηλού φόρτου δεν παρατηρούμε τόσο μεγάλη αύξηση Αυτό έχει να κάνει με τον γεγονός πως ο χρόνος διάδοσης είναι και πάλι μικρός. Έτσι ό χρόνος που χρειάζεται για να επικοινωνήσει ο επικεφαλής με τους εφεδρικούς και να απαντήσει στον πελάτη υπερκαλύπτεται από το χρόνο που χρειάζεται για αναμονή μέχρι να αρχίσει η διαδικασία εξυπηρέτησης. Όσο όμως μεγαλώνει το πλήθος των διακομιστών τόσο προστίθεται χρόνος που χρειάζεται για επικοινωνία, οπότε μετά από λίγο παρατηρούμε αύξηση του μέσου χρόνου εξυπηρέτησης. Ακόμη και τώρα όμως αυτή η αύξηση δεν είναι τόση ώστε να καλύψει το χρόνο αναμονής για εξυπηρέτηση.



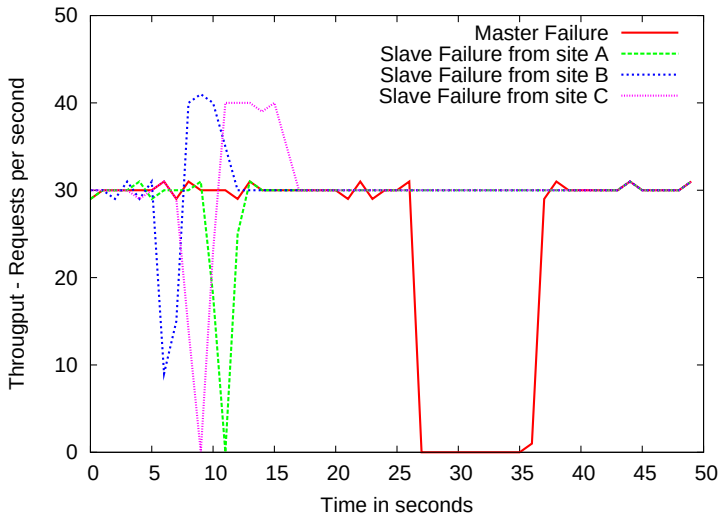
Σχήμα 6.7: Μέσος χρόνος εξυπηρέτησης για διακομιστές που βρίσκονται μέσα σε μία χώρα σε περίπτωση χαμηλού και υψηλού φορτίου

Στο Σχήμα 6.8 φαίνεται η περίπτωση διακομιστών οι οποίοι βρίσκονται σε μία Ήπειρο. Σε αυτή την περίπτωση ο χρόνος διάδοσης είναι πολύ μεγάλος. Στην περίπτωση που έχουμε χαμηλό φόρτο ο μέσος χρόνος εξυπηρέτησης αυξάνει γραμμικά σε σχέση με το πλήθος των διακομιστών καθώς ο επικεφαλής πρέπει να επικοινωνήσει με όλους ώστε να εξυπηρετήσει το αίτημα. Όσον αφορά την περίπτωση υψηλού φόρτου παρατηρείται συμπεριφορά ανάλογα



Σχήμα 6.8: Μέσος χρόνος εξυπηρέτησης για διακομιστές που βρίσκονται μέσα σε μία ήπειρο σε περίπτωση χαμηλού και υψηλού φορτίου

με το πλήθος των διακομιστών. Αρχικά όταν είναι μέχρι 16 οι διακομιστές, ο χρόνος αναμονής είναι μεγαλύτερος από το χρόνο που χρειάζεται το σύστημα για να εξυπηρετήσει την αίτηση. Έτσι μένει σταθερός ο μέσος χρόνος εξυπηρέτησης. Όταν όμως το πλήθος των διακομιστών είναι από 32 και πάνω τότε ο χρόνος αναμονής είναι ελάχιστος σε σχέση με τον χρόνο εξυπηρέτησης. Από αυτό το σημείο και έπειτα παρατηρούμε πως οι δύο περιπτώσεις σχεδόν συμπίπτουν



Σχήμα 6.9: Περιπτώσεις αποτυχιών στην περίπτωση που οι διακομιστές βρίσκονται μέσα σε μία χώρα

Στο Σχήμα 6.9 εξετάζονται οι περιπτώσεις αποτυχιών όταν οι διακομιστές είναι μέσα στην Ελλάδα. Το πλήθος των διακομιστών είναι οχτώ και κάθε πείραμα είναι ανεξάρτητο από τα άλλα, δηλαδή η δεύτερη αποτυχία δεν σημαίνει πως θα μείνουν έξι διακομιστές. Φαίνονται τρεις περιπτώσεις αποτυχιών εφεδρικού διακομιστή, μία για κάθε οργανισμό.

Αυτές μεταξύ τους δεν έχουν καμία διαφορά. Το σύστημα χρειάζεται περίπου ένα με δύο δευτερόλεπτα για να επανακάμψει.

Η περίπτωση που παρουσιάζει πάλι σχετικά μεγαλύτερη καθυστέρηση είναι η περίπτωση αποτυχίας του επικεφαλής Όπως φάνηκε και στην περίπτωση αποτυχίας του επικεφαλής σε τοπικό δίκτυο για το Σχήμα 6.6 υπάρχουν παραπάνω από ένας εφεδρικοί που ανιχνεύουν την αποτυχία του. Αυτοί θα προτείνουν νέα σύνθεση, όμως επειδή το πλήθος είναι μεγάλο είναι πολύ πιθανό να λάβουν μήνυμα reject. Τότε θα κοιμηθούν και θα ξεκινήσουν από την αρχή το πρωτόκολλο Paxos. Σε αντίθεση με το Σχήμα 6.6 το Σχήμα 6.9 παρουσιάζει μεγαλύτερο κενό διάστημα. Αυτό συμβαίνει λόγο της καθυστέρησης που υπάρχει ανάμεσα στους κόμβους. Επίσης δεν παρατηρείται δεύτερη πτώση της απόδοσης επειδή δεν προλαβαίνει το σύστημα να συμφωνήσει σε μία σύνθεση μέχρι να ξυπνήσουν οι υπόλοιποι εφεδρικοί.

Κεφαλαιο 7

Επιλογος

7.1 Επίλογος

7.1 Επίλογος

Σε αυτή την εργασία μελετήθηκαν τα ομοσπονδιακά συστήματα αρχείων. Σχεδιάστηκε το Σύστημα Orion το οποίο παρέχει ένα μηχανισμό ενοποίησης χώρου ονομάτων με βάση των πίνακα εξαγωγής των συστημάτων αρχείων.

Το σύστημα Orion παρέχει ανοχή σε σφάλματα με την διατήρηση πολλαπλών αντιγράφων των μεταδεδομένων του χώρου ονομάτων. Για να γίνει η ενημέρωση των μεταδεδομένων χρησιμοποιήθηκε ένα πρωτόκολλο ολοκλήρωσης δύο φάσεων. Τις περιπτώσεις αποτυχιών των αντιγράφων το σύστημα τις χειρίζεται χρησιμοποιώντας τον αλγόριθμο συμφωνίας Paxos.

Στα πλαίσια τις εργασίας έγινε πρότυπη υλοποίηση των παραπάνω μηχανισμών ώστε να χρησιμοποιηθούν στο σύστημα Orion. Τέλος έγινε μία πειραματική μελέτη για να αξιολογηθεί αν είναι εφικτή η χρήση του μηχανισμού ενοποίησης του ονοματοχώρου σε ομοσπονδιακά συστήματα αρχείων.

Τα πειραματικά αποτελέσματα είναι πολύ ενθαρρυντικά στην περίπτωση ενός οργανισμού όπου οι διακομιστές βρίσκονται πολύ κοντά και ο χρόνος διάδοσης είναι μικρός. Επιπλέον το σύστημα έχει καλή συμπεριφορά στην περίπτωση οργανισμών που βρίσκονται σε διαφορετικές αλλά κοντινές περιοχές, όπως για παράδειγμα εντός του ελλαδικού χώρου.

Όταν όμως το πλήθος των διακομιστών που αντιγράφουν την πληροφορία αυξηθεί πολύ και οι διακομιστές βρίσκονται κατανεμημένοι σε μίας μεγαλύτερης κλίμακας περιοχή, όπως είναι η περίπτωση της Ευρώπης, τότε το σύστημα έχει πολύ μεγάλο χρόνο απόκρισης και μικρή ρυθμαπόδοση.

Το παραπάνω συμβαίνει εξαιτίας της προώθησης των αιτημάτων από τον επικεφαλής σε όλους του εφεδρικούς.

Βιβλιογραφια

[1] A. M. A, R. Morris, T. M. Gil, and B. Chen. Ivy: a read/write peer-to-peer file system. In *USENIX symposium on operating systems design and implementation*, pages 31–44, Boston, MA, Dec. 2002.

[2] O. T. Anderson, L. Luan, C. Everhart, M. Pereira, R. Sarkar, and J. Xu. Global namespace for files. *IBM Systems Journal*, 43(4):702–722, 2004.

[3] Keep it simple: overcome information integration challenges with avaki data grid software. Technical report, Avaki Corporation, July 2003.

[4] C. Baru, R. Moore, A. R. A, and M. Wan. The sdsc storage resource broker. In *IBM CASCON*, Toronto, Canada, Nov. 1998.

[5] T. Chandra, R. Griesemer, and J. Redstone. Paxos made live - an engineering perspective. In *ACM Principles of Distributed Computing*, pages 398–407, Aug. 2007.

[6] G. David and L. Mills. Network time protocol (version 3) specification, implementation and analysis.

[7] J. R. Douceur and J. Howell. Distributed directory service in the farsite file system. In *USENIX Symposium on Operating Systems Design and Implementation*, pages 321–334, 2006.

[8] J. Gray. *Notes on Database Operating Systems*, pages 393–481. Springer-Verlag, 1978.

[9] L. Lamport. The part-time parliament. *ACM Transactions on Computer Systems*, 16(2):133–169, May 1998.

[10] L. Lamport. Paxos made simple. *ACM SIGACT News (Distributed Computing Column*, 32(4):51–58, Dec. 2001.

[11] D. Mazieres. Paxos made practical. Technical report, Computer Science Department, Stanford U, Jan. 2007.

[12] S. Radia, M. N. Nelson, and M. L. Powell. The spring name service. Technical report, Sun Microsystems Laboratories, Nov. 1993. SMLI TR-93-16.

[13] F. B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, pages 299–319, Dec. 1990.

[14] B. Sidebotham. Volumes: The andrew file system data structuring primitive. Technical report, The Information Technology Center, Carnegie-Mellon University, 1986. CMU-ITC-86-053.

[15] J. Stribling, Y. Sovran, I. Zhang, X. Pretzer, J. Li, M. F. Kaashoek, and R. Morris. Flexible, wide-area storage for distributed systems with wheelfs. In *Proc. of the 6th NSDI*, Apr 2009.

[16] J. Zhang and P. Honeyman. Nfsv4 replication for grid storage middleware. In *ACM Intl Workshop on Middleware for Grid Computing*, 2006.

Βιογραφικο

Ο Ευάγγελος Λάππας γεννήθηκε στον Αμαρούσιο Αττικής το 1984. Αποφοίτησε από το $8^ο$ Ενιαίο Λύκειο Αμαρουσίου. Οι βασικές σπουδές πραγματοποιήθηκαν στο Τμήμα Πληροφορικής του Πανεπιστημίου Ιωαννίνων από όπου αποφοίτησε το 2006. Συνέχισε για Μεταπτυχιακές σπουδές στο ίδιο Ίδρυμα και ασχολήθηκε με τον πεδίο των Ομοσπονδιακών συστημάτων Αρχείων.