ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Καταμερισμός φορτίου σε εξυπηρετητές web

Web server Load-balancing

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σπυρίδων Σ. Παπαδάκης

Επιβλέπων : Νεκτάριος Κοζύρης
Επ. Καθηγητής Ε.Μ.Π

Αθήνα, Νοέμβριος 2007

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Καταμερισμός φορτίου σε εξυπηρετητές web

Web server Load-balancing

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σπυρίδων Σ. Παπαδάκης

Επιβλέπων : Νεκτάριος Κοζύρης
Επ. Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 23 Νοεμβρίου 2007

.....	.....	.....
Νεκτάριος Κοζύρης	Παναγιώτης Τσανάκας	Νικόλαος Παπασπύρου
Επ. Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Επ. Καθηγητής Ε.Μ.Π

Αθήνα Νοέμβριος 2007

.....
Σπυρίδων Σ. Παπαδάκης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

ΠΕΡΙΛΗΨΗ

Η ραγδαία ανάπτυξη στους προσωπικούς υπολογιστές προσφέρει στον τελικό χρήστη χαμηλού κόστους συσκευές, επιπλέον η μεγάλη βελτίωση και εξάπλωση στα δίκτυα επικοινωνιών τα τελευταία χρόνια, παρέχει την δυνατότητα πρόσβασης σε ένα τεράστιο, παγκόσμιο, όγκο πληροφοριών και υπηρεσιών σε εκατομμύρια κόσμο. Στις μέρες μας παρατηρείται τρομακτική ανάπτυξη του παγκόσμιου ιστού (www.), η οποία φτάνει το 60% ετησίως και αυξάνεται καθημερινά ολοένα και περισσότερο, στρέφοντας όλο και περισσότερες εταιρείες στην χρήση web-based εφαρμογών (πωλήσεις, ηλεκτρονικό ταχυδρομείο, διαφήμιση, ενημέρωση, διασκέδαση) για την καλύτερη και πιο αποδοτική εξυπηρέτηση των πελατών τους. Αποτέλεσμα της ανωτέρω ανάπτυξης είναι η εξίσου μεγάλη αύξηση του φορτίου εργασίας των εξυπηρετητών (web servers) κάθε εταιρείας, πρόβλημα που γίνεται εντονότερο σε πολύ γνωστούς παροχείς υπηρεσιών όπως www.Google.com, www.Microsoft.com, www.in.gr, www.Cisco.com, www.msn.com, οι οποίοι δέχονται καθημερινά εκατομμύρια επισκέψεις για αναζήτηση και παροχή πληροφοριών.

Η ραγδαία αυτή αύξηση του φορτίου εργασίας στους εξυπηρετητές web κατέστησε αναγκαία τη δημιουργία clusters (πολλών εξυπηρετητών συνδεμένων μεταξύ τους) ούτως ώστε οι εταιρείες να μπορέσουν να αντεπεξέλθουν σε αυτή την αύξηση και να παρέχουν πιο ποιοτικές υπηρεσίες στους χρήστες (χωρίς καθυστέρηση στον χρόνο αναζήτησης πληροφοριών και χωρίς την εμφάνιση μηνυμάτων όπως “The page is temporary not available”, “Server too busy”, “file was not found”). Για να πραγματοποιηθεί όμως κάτι τέτοιο απαιτούνται μέθοδοι για αποδοτικό καταμερισμό της κίνησης στους εξυπηρετητές μέσα στο cluster.

Σκοπός μας, μέσω αυτής της εργασίας, είναι η έρευνα, καταγραφή και ανάλυση γνωστών μεθόδων καταμερισμού φορτίου σε εξυπηρετητές web

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Εξυπηρετητές, καταμερισμός φορτίου, αλγόριθμοι καταμερισμού φορτίου, cluster.

ABSTRACT

The rapid growth of personal computers has offered the end user very cheep Personal Computer devices. Moreover the improvement and spread of communication networks in the last few years has given the ability to millions of people to enter a huge worldwide volume of data and services. In our days a rapid growth of the world wide web (www.) is observed, that reaches the 60% per year and is getting bigger every day, making big companies to use web-based application (sales, e-mail, advertisement, posting up, entertainment) for better and more efficient service of their clients. A result of the above is the equal growth of the load on the web servers of each company, a problem that is more intense at sites of popular companies such as www.Google.com, www.Microsoft.com, www.in.gr, www.Cisco.com, www.msn.com, that have millions of visits every day for information searching.

This rapid growth of the load on the web servers, brought the need of forming clusters (many servers combined together), so that the companies will be able to overcome the extra volume of requests and be able to provide more qualitative service to the clients (without time delay in searching and messages such as “The page is temporary not available”, “Server too busy”, “file was not found”). In order to accomplish such a system there is the need of creating algorithms for load balancing the requests in the cluster.

Our objective throughout this project is the research, record and analysis of known methods for web server load balancing.

KEY WORDS

Servers, load balancing, load balancing algorithms, cluster.

ΠΕΡΙΕΧΟΜΕΝΑ

Σελίδα

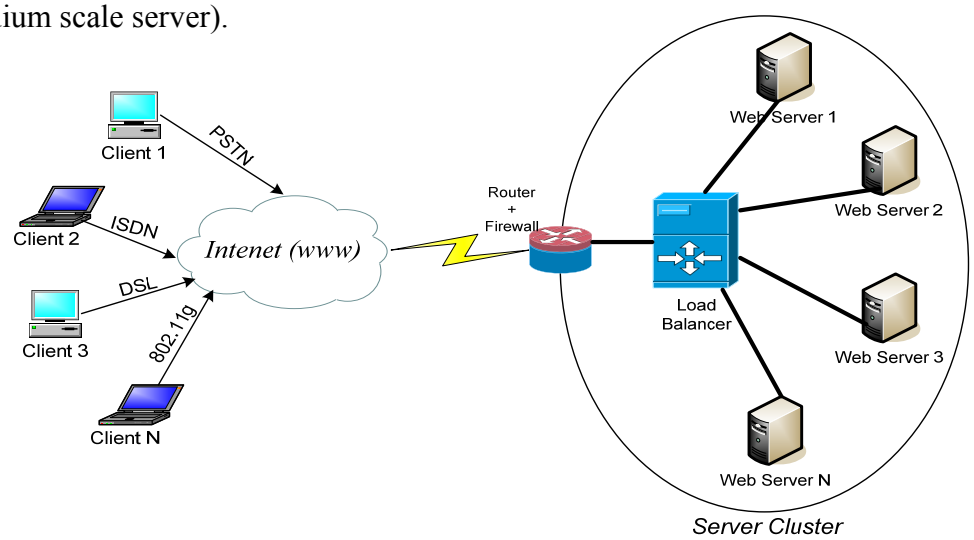
ΠΙΝΑΚΕΣ ΚΑΙ ΣΧΗΜΑΤΑ

ΚΕΦΑΛΑΙΟ 1ο ΕΙΣΑΓΩΓΗ

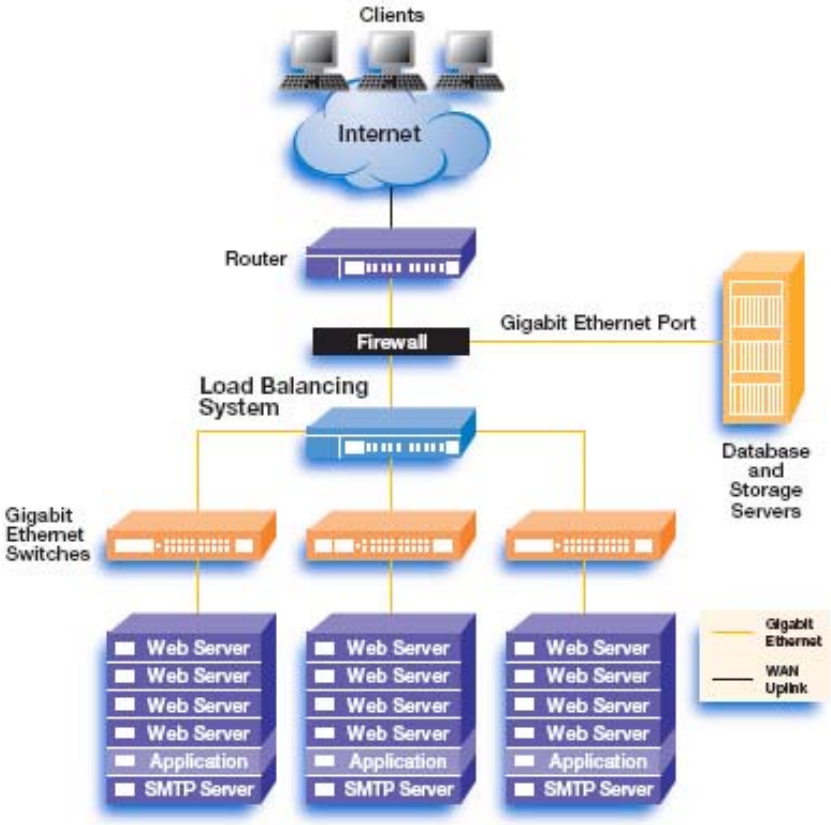
Η ποιοτική παροχή υπηρεσιών που μπορεί να προσφέρει ένας εξυπηρετητής web (στην συνέχεια server) εξαρτάται από δύο παράγοντες, ταχύτητα δικτύου και χρόνος ανταπόκρισης εξυπηρετητή. Η ταχύτητα δικτύου κυρίως εξαρτάται από την ποιότητα και τον τύπο σύνδεσης (PSTN, ISDN, DSL, ασύρματη) του χρήστη (client) στο δίκτυο, όμως από την άλλη ο χρόνος ανταπόκρισης του εξυπηρετητή εξαρτάται από τους πόρους: ταχύτητα ρολογιού επεξεργαστή, μέγεθος μνήμης RAM, απόδοση συσκευών εισόδου/ εξόδου.

Στην περίπτωση χρήσης ενός μόνο server, όταν οι πόροι του μηχανήματος εξαντληθούν μπορούμε να το αναβαθμίσουμε και να εγκαταστήσουμε επεξεργαστή με μεγαλύτερο ρολόι χρονισμού, και περισσότερη μνήμη RAM, αλλά σύντομα και αυτοί οι πόροι θα εξαντληθούν και θα χρειαστεί νέα αναβάθμιση, διαδικασία πολύπλοκη και επιπλέον πολύ δαπανηρή γιατί εκτός από το κόστος της αναβάθμισης είναι αναγκαία και η διακοπή της λειτουργίας του server, που συνεπάγεται με τερματισμό της λειτουργείς τις ιστοσελίδας μας και άρα μείωση των εσόδων μας (φανταστείτε τις συνέπειες σε ένα e-shop).

Αντίθετα μπορούμε να αυξήσουμε τον αριθμό των server οι οποίοι θα δουλεύουν παράλληλα, δημιουργώντας έτσι ένα cluster (σχήμα 1.1 και 1.2), και κάθε φορά που είναι αναγκαίο θα προσθέτουμε ένα ακόμα server (έτσι δεν χρειάζεται η χρήση πολύ ακριβών server, large scale server, αλλά μικρότερων, medium scale server).



Σχήμα 1.1 : Γενική Αναπαράσταση Καταμερισμού Φορτίου με χρήση Cluster



Σχήμα 1.2 :Αναπαράσταση Καταμερισμού Φορτίου με χρήση Cluster σε ποιο πολύπλοκο δίκτυο

Αυτή η τεχνική βελτιστοποίησης της παροχής υπηρεσιών προς τους χρήστες δεν είναι εύκολη διαδικασία και απαιτεί την χρήση εξελιγμένων πρωτοκόλλων και αλγορίθμων δημιουργώντας έτσι την ανάγκη εύρεσης μεθόδων καταμερισμού του φορτίου ανάμεσα στους servers μέσα στα cluster με όσο πιο αποδοτικό τρόπο γίνεται.

Σκοπός μας μέσα από αυτή την εργασία είναι να καταγράψουμε τις πιο γνωστές μεθόδους καταμερισμού φορτίου που χρησιμοποιούνται σήμερα.

1.1 Το Διαδικτυο

[2]Η Δικτύωση είναι ένα από πλέον ενδιαφέροντα και σημαντικά τεχνολογικά πεδία στις μέρες μας. Το Διαδίκτυο (Internet) διασυνδέει εκατομμύρια (και σύντομα δισεκατομμύρια) υπολογιστές, παρέχοντας μια παγκόσμια υποδο-

μή για επικοινωνία, αποθήκευση και υπολογιστικές εργασίες. Επιπλέον. Το διαδίκτυο στις μέρες μας ολοκληρώνεται με τεχνολογίες κινητής και ασύρματης επικοινωνίας, εισάγοντας μια εντυπωσιακή σειρά νέων εφαρμογών.

[2]Το δημόσιο διαδίκτυο είναι ένα παγκόσμιο δίκτυο υπολογιστών. Οι περισσότερες από αυτές τις υπολογιστικές συσκευές είναι παραδεισιακά επιτραπέζια PC, σταθμοί εργασίας UNIX και καλούμενοι εξυπηρετητές (servers), που αποθηκεύουν και μεταδίδουν πληροφορίες, όπως ιστοσελίδες και μηνύματα e-mail. Όλο και περισσότερα, μη παραδοσιακά τερματικά συστήματα διαδικτύου, όπως PDAs (Personal Digital Assistants), τηλεοράσεις, φορητοί υπολογιστές, αυτοκίνητα συνδέονται καθημερινά, τον Ιανουάριο 2002 υπήρχαν 100-500 εκατομμύρια τερματικά συστήματα που χρησιμοποιούσαν το διαδίκτυο και ο αριθμός συνεχίζει να αυξάνεται εκθετικά.

Τα τερματικά συστήματα προσπελαύνουν το διαδίκτυο μέσω παρόχων (ISP) υπηρεσιών διαδικτύου (ISP, Internet Service Providers), που περιλαμβάνουν οικιακούς ISP σαν τους AOL και MSN, πανεπιστημιακούς και εταιρικούς. Κάθε ISP είναι ένα δίκτυο δρομολογητών και ζεύξεων επικοινωνίας. Οι διάφοροι ISP παρέχουν μια ποικιλία τύπων προσπέλασης δικτύου προς τα τερματικά συστήματα, περιλαμβανομένου της προσπέλασης 56 Kbps μέσο τηλεφώνου και της οικιακής ευρυζωνικής προσπέλασης, όπως είναι το καλωδιακό μόντεμ ή το DSL. Τα τερματικά συστήματα, εκτελούν πρωτόκολλα, τα οποία ελέγχουν την αποστολή και τη λήψη των πληροφοριών μέσα στο διαδίκτυο. Το TCP (Transmission Control Protocol) και το IP (Internet Protocol, καθορίζει την μορφή των πακέτων που στέλνονται κα λαμβάνονται ανάμεσα σε δρομολογητές και τερματικά συστήματα) είναι δύο από τα σημαντικότερα πρωτόκολλα στο διαδίκτυο. Τα κύρια πρωτόκολλα του διαδικτύου καλούνται συλλογικά TCP/IP [2].Επιπλέον θα ήταν χρήσιμο να αναφέρουμε ότι σε γενικές γραμμές, υπάρχουν δύο είδη τεχνολογιών μετάδοση πληροφοριών στο διαδίκτυο:

[3]Τα **δίκτυα εκπομπής (broadcast networks)** έχουν ένα μόνο κανάλι επικοινωνίας το οποίο είναι κοινόχρηστο από όλες τις μηχανές του δικτύου. Τα σύντομα μηνύματα από στέλνονται από κάθε μηχάνημα, τα οποία συχνά ονομάζονται πακέτα (packets), λαμβάνονται από όλες τις άλλες μηχανές. Ένα πεδίο διεύθυνσης μέσα στο πακέτο προσδιορίζει τον επιθυμητό παραλήπτη. Το κάθε μηχάνημα, μόλις λάβει ένα πακέτο εξετάζει αυτό το πεδίο διεύθυνσης. Αν το

πακέτο προορίζεται για το μηχάνημα που το έλαβε. Αυτό προχωρά στην επεξεργασία του. Αν το πακέτο προορίζεται για κάποιο άλλο μηχάνημα, τότε απλώς παραβλέπεται.

Τα **δίκτυα από σημείο σε σημείο (point-to-point)** αποτελούνται από πολλές συνδέσεις ανάμεσα σε ζεύγη μηχανών. Για να φτάσει ένα πακέτο από την προέλευση στον προορισμό του σε αυτόν τον τύπο δικτύου, μπορεί να χρειαστεί να επισκεφτεί πρώτα μια ή περισσότερες ενδιάμεσες μηχανές. Συχνά υπάρχουν πολλά πιθανά δρομολόγια, με διαφορετικά μήκη, άρα και η ανεύρεση δρομολογίων είναι ένα σημαντικό ζήτημα στα δίκτυα από σημείο σε σημείο. Ένας γενικός κανόνας (αν και υπάρχουν πολλές εξαιρέσεις) είναι ότι τα μικρότερα, γεωγραφικά περιορισμένα δίκτυα συνήθως χρησιμοποιούν εκπομπή, ενώ τα μεγαλύτερα δίκτυα συνήθως είναι δίκτυα από σημείο σε σημείο. Η μετάδοση από σημείο σε σημείο με ένα αποστολέα και ένα παραλήπτη μερικές φορές ονομάζεται αποκλειστική διανομή (unicast) [3].

1.2 Το Μοντελο Αναφορας OSI

Πώς όμως συνδέονται όλα αυτά μεταξύ τους; Για να επικοινωνήσουν οι συσκευές πάνω σε ένα δίκτυο θα πρέπει να υπάρχουν κάποιοι κανόνες που πρέπει να ακολουθούν και θα πρέπει να είναι οι ίδιοι για όλες εις συσκευές για όλο τον κόσμο, κάτι σαν τους γραμματικούς και λεκτικούς κανόνες που έχει κάθε γλώσσα, αυτοί οι κανόνες ονομάζονται πρωτόκολλα[1]. Χωρίς αυτά θα ήταν αδύνατον να υπάρξει επικοινωνία μεταξύ συσκευών. Πάλι όμως το πρόβλημα δεν είχε λυθεί, θα έπρεπε με κάποιο τρόπο όλα αυτά τα πρωτόκολλα που κατά την διάρκεια μια διασύνδεσης είναι απαραίτητα να συνεργαστούν μεταξύ τους για ευκολότερη πιο γρήγορη και συγχρόνως ασφαλή διασύνδεση.

Καθώς η ανάγκη για δικτύωση γινόταν όλο και μεγαλύτερη έπρεπε να βρεθούν λύσεις. Η λύση ήρθε το 1978 όπου Διεθνής Οργανισμός Τυποποίησης (ISO) ανάπτυξε μια πρόταση που θα πραγματοποιούσε αυτό τον στόχο, η οποία πρόταση αναθεωρήθηκε το 1984 και έγινε παγκόσμιο πρότυπο γνωστό ως μοντέλο OSI (Open System Interconnection, Διασύνδεση Ανοικτών Συστημάτων),

[1] Στην Ιστοσελίδα: http://www.protocols.com/ μπορείτε να βρείτε αναλυτικά όλα τα πρωτόκολλα.

το οποίο τυποποιεί τα πρωτόκολλα που χρησιμοποιούνται για την διασύνδεση διαφόρων συστημάτων και πως αυτά επικοινωνούν μεταξύ τους.

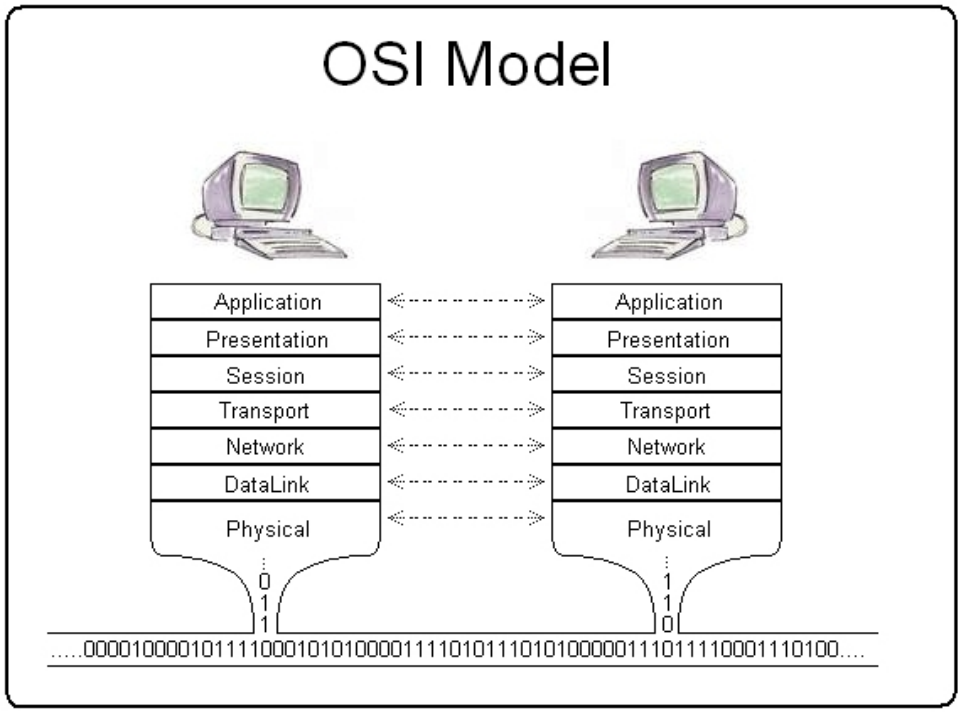
[2]"Το μοντέλο OSI διαιρεί τις λειτουργίες ενός πρωτοκόλλου σε μια σειρά από 7 επίπεδα. Κάθε επίπεδο χρησιμοποιεί μόνο τις λειτουργίες του κάτω επιπέδου και προσφέρει λειτουργικότητα στο πάνω επίπεδο. Ένα σύστημα που παρουσιάζει συμπεριφορά πρωτοκόλλου και που είναι χωρισμένο σε επίπεδα ονομάζεται στοίβα πρωτοκόλλων ή απλά στοίβα. Οι στοίβες κατασκευάζονται με υλικό είτε με λογισμικό. Τυπικά, τα κατώτερα επίπεδα κατασκευάζονται με υλικό, ενώ τα ανώτερα επίπεδα είναι εφαρμογές λογισμικού.

Το μοντέλο OSI είναι βασικά συνδεδεμένο με τον κλάδο των υπολογιστών και την δικτύωσή τους. Το κύριο χαρακτηριστικό του είναι η διεπαφή μεταξύ των επιπέδων, η οποία υπαγορεύει τις προδιαγραφές της αλληλεπίδρασης αυτών των επιπέδων. Αυτό σημαίνει ότι ένα επίπεδο δημιουργημένο από έναν κατασκευαστή μπορεί να συνεργαστεί με το διπλανό επίπεδο που έχει κατασκευάσει άλλος (με την προϋπόθεση ότι έχει γίνει αντιληπτή η προδιαγραφή σωστά). Αυτές οι προδιαγραφές είναι τυπικά γνωστές ως RFC (Request for Comments) στους σχετικούς με τα πρωτόκολλα TCP/IP, και είναι πρότυπα για τον οργανισμό Διεθνή Οργανισμό Τυποποίησης ISO.

Το μοντέλο OSI είναι μια ιεραρχική δομή επτά επιπέδων που καθορίζει τις απαιτήσεις για επικοινωνία δύο υπολογιστών μεταξύ τους και καθορίστηκε ως πρότυπο ISO 7498. Θεωρήθηκε ότι θα επέτρεπε την διασύνδεση μεταξύ διαφόρων συσκευών που προσέφεραν στην αγορά οι διάφοροι κατασκευαστές. Το μοντέλο επιτρέπει σε όλα τα στοιχεία ενός δικτύου να συλλειτουργούν ανεξάρτητα από το ποιος είναι ο κατασκευαστής τους. Περί τα τέλη της δεκαετίας 1980 ο ISO συνιστούσε την εφαρμογή του μοντέλου OSI ως δικτυακού προτύπου.

Στο παρακάτω σχήμα παρουσιάζουμε το μοντέλο OSI και πως τα διάφορα επίπεδα συνδέονται μεταξύ τους.

[2] από http://el.wikipedia.org



Σχήμα 1.3 : Επίπεδα OSI

1.2.1 ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΕΠΙΠΕΔΩΝ OSI[3]

Επίπεδο 1: Φυσικό (Physical)

Το φυσικό επίπεδο ορίζει όλες τις ηλεκτρικές και φυσικές προδιαγραφές των συσκευών. Σ' αυτές περιλαμβάνονται οι σχηματισμοί των ακίδων, οι επιτρεπτές τάσεις, οι προδιαγραφές των καλωδίων. Συσκευές φυσικού επιπέδου είναι οι διανεμητές (hub), οι αναμεταδότες (repeater), οι κάρτες δικτύου (network card), οι προσαρμοστές (adaptor) αρτηρίας (bus). Οι κυριότερες λειτουργίες και υπηρεσίες του φυσικού επιπέδου είναι:

- Έναρξη και περαίωση της ηλεκτρικής σύνδεσης μίας επικοινωνιακής συσκευής.
- Συμμετοχή σε διαδικασίες όπου οι επικοινωνιακές συσκευές εξυπηρετούν αποτελεσματικά πολλούς χρήστες. Επιλύονται προβλήματα προτεραιότητας πρόσβασης και ελέγχου ροής δεδομένων.
- Διαμόρφωση και αποδιαμόρφωση των ψηφιακών δεδομένων κατά την μετάδοση από συσκευή σε συσκευή. Για παράδειγμα, τα ψηφιακά ηλεκτρικά σήματα μπορεί να ταξιδέψουν ως αναλογικά σε χάλκινο καλώδι-

[3] από http://el.wikipedia.org

ο, μετά σε οπτική ίνα, μετά να μεταδοθούν από ράδιοζεύξη ή δορυφορικά, να φθάσουν πάλι αναλογικά σε χάλκινο καλώδιο, και να γίνουν ψηφιακά στον παραλήπτη.

Επίπεδο 2: Διασύνδεσης Δεδομένων (Data Link)

Το επίπεδο διασύνδεσης δεδομένων παρέχει τα λειτουργικά και διαδικαστικά μέσα για την μεταφορά δεδομένων από την μια συσκευή του δικτύου στην άλλη, και για τον έλεγχο και την πιθανή διόρθωση σφαλμάτων που συμβαίνουν στο φυσικό επίπεδο. Το πιο γνωστό παράδειγμα είναι το Ethernet.

Επίπεδο 3: Δικτύου (Network)

Το επίπεδο δικτύου παρέχει τα λειτουργικά και διαδικαστικά μέσα για την μεταφορά στοιχειοσειρών δεδομένων μεταβλητού μήκους από μια προέλευση σε ένα προορισμό, μέσα από ένα ή περισσότερα δίκτυα, ενώ διατηρεί την ποιότητα εξυπηρέτησης που απαιτεί το επίπεδο μεταφοράς. Το επίπεδο δικτύου εκτελεί λειτουργίες δρομολόγησης, με πιθανές τμηματοποιήσεις / αποτμηματοποιήσεις, και αναφέρει σφάλματα σχετικά με την παράδοση των πακέτων. Οι δρομολογητές (routers) λειτουργούν στο επίπεδο αυτό, και στέλνοντας δεδομένα σε διασυνδεδεμένα δίκτυα έκαναν το Διαδίκτυο πραγματικότητα. Το καλύτερο παράδειγμα πρωτοκόλλου δικτύου είναι το Πρωτόκολλο Διαδικτύου (Internet Protocol, IP).

Επίπεδο 4: Μεταφοράς (Transport)

Το επίπεδο μεταφοράς διεκπεραιώνει την μεταφορά των δεδομένων από χρήστη σε χρήστη, απαλλάσσοντας έτσι τα ανώτερα επίπεδα από κάθε φροντίδα να προσφέρουν αξιόπιστη και οικονομική μεταφορά δεδομένων. Το επίπεδο μεταφοράς ελέγχει την αξιοπιστία ενός χρησιμοποιούμενου καναλιού με έλεγχο ροής (flow control), τμηματοποίηση και αποτμηματοποίηση (segmentation / desegmentation), και έλεγχο σφαλμάτων (error control). Ορισμένα πρωτόκολλα καταγράφουν καταστάσεις και συνδέσεις, οπότε κρατούν λογαριασμό των πακέτων και επανεκπέμπουν αυτά που δεν παραλήφθηκαν σωστά. Τα διάφορα

πρωτόκολλα σχηματίζουν διαφορετικά τα πακέτα πληροφοριών. Το καλύτερο παράδειγμα πρωτοκόλλου μεταφοράς είναι το TCP (Transmission Control Protocol, πρωτόκολλο ελέγχου μετάδοσης).

Επίπεδο 5: Συνόδου (Session)

Το επίπεδο συνόδου ελέγχει τις συνόδους (δηλαδή τους διαλόγους) μεταξύ δύο υπολογιστών, του Α και του Β. Ξεκινά, διαχειρίζεται και τερματίζει την σύνδεση μεταξύ μιας τοπικής και μιας απομακρυσμένης (remote) εφαρμογής. Αντιμετωπίζει λειτουργίες FDX (full duplex, οι Α και Β μιλούν ταυτόχρονα από δύο κανάλια) ή HDX (half-duplex, μιλάει ο Α και μετά απαντάει ο Β από το ένα διαθέσιμο κανάλι), και έχει διαδικασίες αποθήκευσης κατάστασης (check point) αναβολής, περαίωσης λειτουργίας, και επανεκκίνησης. Αυτό το επίπεδο είναι υπεύθυνο για το ομαλό κλείσιμο της συνόδου, (που είναι ιδιότητα του TCP), και επίσης για αποθήκευση κατάστασης και ανάκτηση (recovery).

Επίπεδο 6: Παρουσίασης (Presentation)

Το επίπεδο παρουσίασης μετασχηματίζει τα δεδομένα σε τυπική μορφή που την αναμένει το επίπεδο εφαρμογών. Στο επίπεδο αυτό γίνεται στα δεδομένα κρυπτογράφηση, συμπίεση, κωδικοποίηση MIME, και όποια άλλη διαμόρφωση απαιτεί η μορφή δεδομένων ή ο σχεδιαστής του πρωτοκόλλου. Ως παραδείγματα αναφέρουμε την μετατροπή αρχείων από κώδικα EBCDIC σε κώδικα ASCII, και την μετατροπή της δομής των δεδομένων σε μορφή XML ή το αντίστροφο (π.χ. από XML σε έγγραφο DOC).

Επίπεδο 7: Εφαρμογών (Application)

Το επίπεδο εφαρμογών παρέχει στον χρήστη έναν τρόπο να προσπελάσει μέσω μιας εφαρμογής τις πληροφορίες ενός δικτύου. Αυτό το επίπεδο είναι η κύρια διεπαφή του χρήστη με την εφαρμογή, και συνεπώς με το δίκτυο. Στο επίπεδο αυτό γίνεται η διαχείριση των κατανεμημένων εφαρμογών και η αποστολή των ηλεκτρονικών επιστολών (e-mail).

Στον παρακάτω πίνακα φαίνονται συνοπτικά όλα τα χαρακτηριστικά των επιπέδων και τα πρωτόκολλα κάθε επιπέδου.

	Τι μεταφέρεται	Επίπεδο	Λειτουργίες	Πρωτόκολλα
Επίπεδα Λογισμικού	Δεδομένα	7. Επίπεδο εφαρμογών	Διαδικασίες δικτύου προς εφαρμογές	DNS; FTP; TFTP; BOOTP; SNMP;RLOGIN; SMTP; MIME; NFS; FINGER; TELNET; NCP; APPC; AFP; SMB
		6. Επίπεδο παρουσίασης	Παρουσίαση δεδομένων και κρυπτογράφηση	
		5. Επίπεδο συνόδου	Επικοινωνία υπολογιστών, συγχρονισμός	NetBIOS Names Pipes Mail Slots RPC
	Τμήματα (Segments)	**4. Επίπεδο μεταφοράς**	Διασφάλιση συνδέσεων, αξιοπιστία. **Από εδώ και πάνω δουλεύουν οι load balancer συσκευές.**	TCP, ARP, RARP; SPX, NWLink NetBIOS / NetBEUI ,ATP
Επίπεδα Υλικού	Πακέτα (Packets)	3. Επίπεδο δικτύου	Δρομολόγηση πακέτων, λογικές διευθύνσεις (IP)	IP; ARP; RARP, ICMP; RIP; OSFP; IGMP; IPX ,NWLink NetBEUI ,OSI DDP ,DECnet
	Πλαίσια (Frames)	2. Επίπεδο διασύνδεσης δεδομένων	Φυσικές διευθύνσεις (MAC & LLC)	802.1 OSI Model 802.2 Logical Link Control ,802.3 CSMA/CD (Ethernet) ,802.4 Token Bus (ARCnet) ,802.5 Token Ring ,802.12 Demand Priority
	Δυαδικά ψηφία (Bits)	1. Φυσικό επίπεδο	Μετάδοση ψηφίων στο κανάλι επικοινωνίας	IEEE 802 IEEE 802.2 ISO 2110 ISDN

Πίνακας 1.1 :Χαρακτηριστικά κάθε επιπέδου OSI

1.3 Το Μοντελο TCP/IP[4]

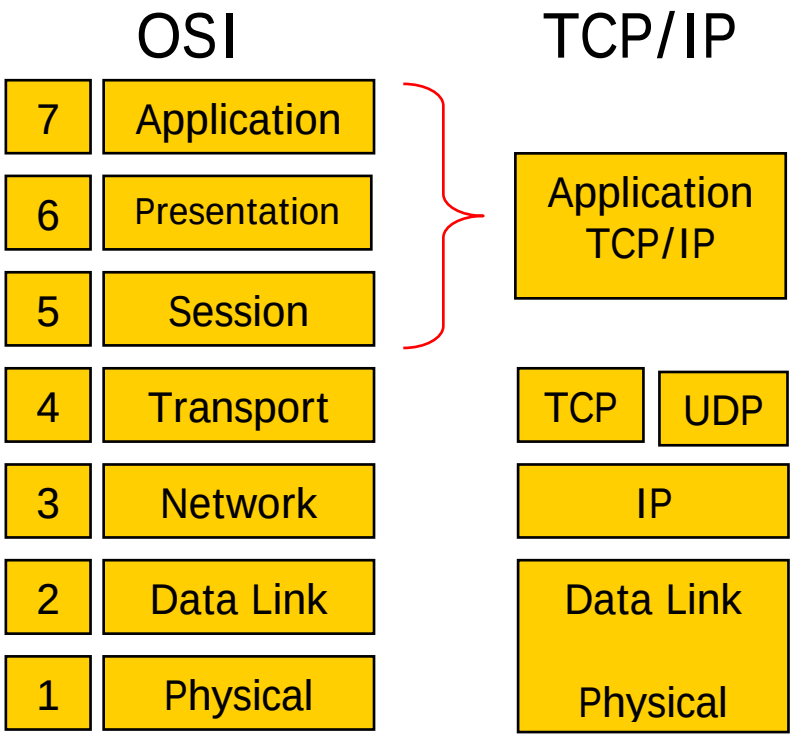
Το TCP/IP ή και Σουίτα Πρωτοκόλλων Διαδικτύου (Internet protocol suite) είναι μια συλλογή πρωτοκόλλων επικοινωνίας στα οποία βασίζεται το Διαδίκτυο αλλά και μεγάλο ποσοστό των εμπορικών δικτύων. Η ονομασία TCP/IP προέρχεται από τις συντομογραφίες των δυο κυριότερων πρωτοκόλλων που περιέχει : το TCP ή transmission control Protocol (Πρωτόκολλο Ελέγχου Μετάδοσης) και το IP ή Internet Protocol (Πρωτόκολλο Διαδικτύου).

Αυτή η συλλογή πρωτοκόλλων, όπως και πολλές άλλες άλλωστε, είναι οργανωμένη σε στρώματα ή επίπεδα (layers) όπως και το μοντέλο OSI. Το καθένα τους απαντά σε συγκεκριμένα προβλήματα μεταφοράς δεδομένων και παρέχει μια καθορισμένη υπηρεσία στα υψηλότερα στρώματα. Τα ανώτερα επίπε-

[4] από http://el.wikipedia.org

δα είναι πιο κοντά στη λογική του χρήστη και εξετάζουν πιο αφηρημένα δεδομένα, στηριζόμενα σε πρωτόκολλα χαμηλότερων στρωμάτων για να μεταφράσουν δεδομένα σε μορφές που μπορούν να διαβιβαστούν με φυσικά μέσα.

Στο παρακάτω σχήμα παρουσιάζονται οι διαφορές στα επίπεδα ανάμεσα στο μοντέλο OSI και το TCP/IP:



Σχήμα 1.4 :Διαφορές μοντέλων OSI και TCP/IP

Όπως παρουσιάζεται τα τρία ανώτερα στρώματα του μοντέλου OSI (Εφαρμογής, Παρουσίασης και Συνόδου) αποτελούν ένα ενιαίο στρώμα στο TCP/IP, το επίπεδο Εφαρμογής. Τα χαρακτηριστικά του στρώματος Συνόδου αναλαμβάνονται από τις ίδιες εφαρμογές ή απλά αγνοούνται.

1.3.1 ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΕΠΙΠΕΔΩΝ TCP/IP

Επίπεδο : Εφαρμογής

Το στρώμα εφαρμογής χρησιμοποιείται από την πλειοψηφία των προγραμμάτων δικτύου. Το πρόγραμμα παραδίδει τα δεδομένα σε μια μορφή που ορίζει τα ίδιο. Εφ' όσον το TCP/IP δεν παρέχει στρώματα μεταξύ των στρωμάτων εφαρμογής και μεταφοράς, όλες οι λειτουργίες παρουσίασης και συνεδρίας (βλέπε μοντέλο OSI) πρέπει να υλοποιηθούν σ' αυτό το επίπεδο. Αυτή η διαδικασία διευκολύνεται με την χρήση βιβλιοθηκών.

Επίπεδο : Μεταφοράς

Το στρώμα μεταφοράς είναι υπεύθυνο για την μεταφορά μηνυμάτων, ανεξαρτήτως του υποκείμενου δικτύου, με έλεγχο σφαλμάτων (error control), κατάτμηση (fragmentation) και ρύθμιση ροής (flow control). Η μετάδοση μηνυμάτων μεταξύ δυο οντοτήτων μπορεί να κατηγοριοποιηθεί ως εξής:

- Connection oriented, π.χ. TCP
- Connection less, π.χ. UDP

Η λειτουργία του στρώματος αυτού μπορεί να συγκριθεί με αυτή οποιουδήποτε μηχανισμού/ μέσου μεταφοράς, π.χ. ένα όχημα που πρέπει να εξασφαλίζει την πλήρη και ασφαλή διακίνηση του φορτίου του. Το στρώμα μεταφοράς παρέχει αυτή την υπηρεσία σύνδεσης εφαρμογών μεταξύ τους, κάνοντας χρήση θυρών (ports). Καθώς το IP προσφέρει μόνο παράδοση όσο το δυνατόν καλύτερα (best effort delivery), το στρώμα μεταφοράς είναι το πρώτο επίπεδο όπου λαμβάνεται υπόψη το θέμα της αξιοπιστίας. Παραδείγματος χάρη, σε μια προσπάθεια αξιόπιστης μετακίνησης δεδομένων, το TCP που είναι ένα connection oriented πρωτόκολλο, έχει τα εξής χαρακτηριστικά:

- Τα δεδομένα έρχονται στην ίδια σειρά με την οποία στάλθηκαν
- Ελάχιστος έλεγχος σφαλμάτων
- Ανεπιθύμητα αντίγραφα απορρίπτονται
- Χαμένα / απορριμμένα πακέτα ξαναστέλνονται

- Έλεγχος κυκλοφοριακής συμφόρησης (congestion control)

Τα πρωτόκολλα δυναμικής δρομολόγησης (dynamic routing), που κανονικά θα έπρεπε να βρίσκονται σε αυτό το στρώμα του TCP/IP (αφού λειτουργούν πάνω από το IP) αντιμετωπίζονται συχνά ως τμήματα του επίπεδου δικτύου (π.χ. το OSPF).

Επίπεδο : Δικτύου

Ο σκοπός του στρώματος δικτύου είχε αρχικά καθοριστεί ως η μεταφορά πακέτων μέσω ενός ενιαίου δικτύου. Με την εμφάνιση πιο σύνθετων μορφών δικτύων, προστέθηκαν επιπλέον χαρακτηριστικά στο στρώμα αυτό, έτσι ώστε ο ρόλος του να είναι πια η διακίνηση δεδομένων από το δίκτυο πηγή στο δίκτυο προορισμού. Αυτό προϋποθέτει συνήθως την δρομολόγηση πακέτων διαμέσου ενός δικτύου δικτύων (inter network) ή διαδικτύου (με μικρά γράμματα). Στην σουίτα πρωτοκόλλων Διαδικτύου, το IP μεταφέρει τα πακέτα δεδομένων από την πηγή, στον προορισμό. Το IP μπορεί να εξυπηρετήσει διάφορα πρωτόκολλα ανωτέρων επιπέδων (upper layer protocols) ˙ το καθένα τους προσδιορίζεται με έναν αποκλειστικό αριθμό πρωτοκόλλου: π.χ. το ICMP και το IGMP έχουν τους αριθμούς 1 και 2 αντίστοιχα.

Επίπεδο : Συνδέσμου

Το στρώμα αυτό, ρόλος του είναι η διακίνηση πακέτων του επιπέδου δικτύου μεταξύ δυο οντοτήτων, δεν είναι στην ακρίβεια μέρος της σουίτας πρωτοκόλλων Διαδικτύου, διότι το IP λειτουργεί με διάφορα στρώματα συνδέσμου. Η διαδικασία διαβίβασης πακέτων σε ένα συγκεκριμένο επίπεδο συνδέσμου μπορεί να ελέγχεται είτε από τον οδηγό του interface, είτε το firmware σύνολο εξειδικευμένων κυκλωμάτων (chip sets), είτε τέλος από ένα συνδυασμό τον προ-αναφερθέντων. Αυτά θα εκτελέσουν τις λειτουργίες σύνδεσης δεδομένων (data link), όπως π.χ. την πρόσθεση επικεφαλίδας (packet header) πριν την αποστολή, την ίδια τη διαβίβαση του πλαισίου (frame) με τη χρήση ενός φυσικού μέσου.

Για συνδέσεις μέσω μόντεμ (σε γραμμή τηλεφώνου), τα πακέτα IP μεταφέρονται συνήθως χρησιμοποιώντας το PPP. Σε ευρυζωνικές συνδέσεις (π.χ.

ADSL) συναντάμε το PPPoE. Σε τοπικά δίκτυα, τα πρωτόκολλα Ethernet ή IEEE 802.11 (για ενσύρματα ή ασύρματα δίκτυα αντίστοιχα) είναι πιο κοινά. Για δίκτυα ευρείας περιοχής (WAN) χρησιμοποιούνται συχνά το PPP πάνω σε γραμμές T carrier ή E carrier, το Frame relay το ATM ή το Packet over SONET/SDH (POS).

Το στρώμα συνδέσμου είναι επίσης το επίπεδο όπου τα πακέτα μπορούν να αναχαιτιστούν για να σταλθούν σ' ένα ιδεατό ιδιωτικό δίκτυο (Virtual Private network, VPN). Σ αυτήν την περίπτωση, τα δεδομένα του επιπέδου αυτού αντιμετωπίζονται ως δεδομένα εφαρμογής, και "ξανακατεβαίνουν" την στοίβα πρωτοκόλλων Διαδικτύου για να σταλθούν. Στη λαμβάνουσα πλευρά, τα δεδομένα ανεβαίνουν δυο φορές την στοίβα (μια για το VPN και μια δεύτερη για τη δρομολόγηση).

Το φυσικό επίπεδο, που αποτελείται από τα φυσικά στοιχεία του δικτύου (π.χ. hubs, repeaters, καλώδια δικτύου, οπτικές ίνες, ομοαξονικά καλώδια, κάρτες δικτύων) και τις προδιαγραφές χαμηλού επιπέδου των σημάτων (τάση, συχνότητα, κλπ.), θεωρείται συχνά ως μέρος του στρώματος συνδέσμου.

1.4 ΟΡΙΣΜΟΙ ΔΙΚΤΥΩΝ

1.4.1 ΔΙΕΥΘΥΝΣΗΟΔΟΤΗΣΗ (IP ADDRESS)[5]

Μία διεύθυνση IP (IP address – Internet protocol address), είναι ένας μοναδικός αριθμός που χρησιμοποιείται από συσκευές για τη μεταξύ τους αναγνώριση και συνεννόηση σε ένα δίκτυο υπολογιστών που χρησιμοποιεί το Internet Protocol standard. Κάθε συσκευή που ανήκει στο δίκτυο - όπως επίσης δρομολογητές (routers), υπολογιστές, time servers, εκτυπωτές, μηχανές για fax μέσω Internet, και ορισμένα τηλέφωνα - πρέπει να έχει τη δική της μοναδική διεύθυνση. Μία διεύθυνση IP μπορεί να θεωρηθεί το αντίστοιχο μιας διεύθυνσης κατοικίας ή ενός αριθμού τηλεφώνου (σύγκριση με VoIP) για έναν υπολογιστή ή άλλη συσκευή δικτύου στο Διαδίκτυο. Όπως κάθε διεύθυνση κατοικίας

[5] από http://el.wikipedia.org

και αριθμός τηλεφώνου αντιστοιχούν σε ένα και μοναδικό κτίριο ή τηλέφωνο, μια IP address χρησιμοποιείται για τη μοναδική αναγνώριση ενός υπολογιστή ή άλλης συσκευής που συνδέεται στο δίκτυο.

Μια διεύθυνση IP μπορεί να "μοιράζεται" σε πολλές συσκευές-πελάτες είτε επειδή αυτές είναι μέρος ενός hosting web server environment, είτε λόγω ενός proxy server(π.χ. ενός παρόχου Υπηρεσιών Διαδικτύου (ISP) ή μιας υπηρεσίας για εξασφάλιση ανωνυμίας - anonymous service) που λειτουργού ως μεσολαβητές. Στην τελευταία περίπτωση (χρήση διακομιστή μεσολάβησης) η πραγματική διεύθυνση IP μπορεί να αποκρύπτεται από το διακομιστή που δέχεται αίτηση. Η αναλογία στα τηλεφωνικά συστήματα θα ήταν η χρήση διεθνών ή τοπικών αριθμών κλήσης (proxy) και επεκτάσεων (shared).

Αυτή την στιγμή η έκδοση που χρησιμοποιείται είναι Το IPv4 χρησιμοποιεί διευθύνσεις των 32-bit (4 byte), που περιορίζουν το πλήθος διευθύνσεων σε 4.294.967.296 (232) πιθανές μοναδικές διευθύνσεις. Εντούτοις, πολλές παρακρατούνται για ειδικούς λόγους, όπως για χρήση σε ιδιωτικά δίκτυα (~18 εκατομμύρια) ή διευθύνσεις multicast (~1 εκατομμύριο). Κατά αυτόν τον τρόπο, μειώνεται ο αριθμός που μπορεί να διατεθεί για δημόσιες διευθύνσεις Διαδικτύου και, καθώς ο αριθμός διαθέσιμων διευθύνσεων καταναλώνεται, η έλλειψη εμφανίζεται να είναι αναπόφευκτη μακροπρόθεσμα. Αυτός ο περιορισμός έχει βοηθήσει στη στροφή προς το IPv6, που είναι αυτήν την περίοδο σε αρχικά στάδια επέκτασης και ο μόνος υποψήφιος αντικαταστάτης του IPv4.

1.4.2 ROUTER

Δρομολογητής (router) είναι μια συσκευή, η οποία αναλαμβάνει την αποστολή και λήψη πακέτων πληροφοριών (δρομολόγηση) μεταξύ ενός ή περισσοτέρων διακομιστών (server) και πελατών (client), σε ένα δίκτυο. Η δρομολόγηση γίνεται με βάση διάφορα κριτήρια, ανάμεσα σε διαφορετικές πιθανές διαδρομές.

Κάθε δρομολογητής τρέχει ένα η περισσότερα πρωτόκολλα δρομολόγησης. Με βάση αυτά τα πρωτόκολλα ο δρομολογητής καθορίζει ποιος η ποίοι δρομολογητές είναι οι καλύτεροι κάθε χρονική περίοδο και δρομολογεί τα πακέτα προς

αυτούς. Ορισμένα πολύ γνωστά πρωτόκολλα δρομολόγησης είναι τα εξής: RIP, OSPF, BGP.

1.4.3 MAC ADDRESS

Μια διεύθυνση Media Access Control (διεύθυνση MAC) είναι ένας δεκαεξαδικός σειριακός αριθμός ο οποίος είναι μοναδικός για κάθε δικτυακή συσκευή (κάρτα δικτύου) σε όλο τον πλανήτη. Ο αριθμός έχει τη μορφή xx:xx:xx:xx:xx:xx, για παράδειγμα 0A:12:A1:B2:AE:04. Η διεύθυνση MAC χρησιμοποιείται για την επικοινωνία μεταξύ των δικτυακών συσκευών. Σε κάθε επικοινωνία οποιασδήποτε δικτυακής συσκευής με μια άλλη, ο αριθμός αυτός αποκαλύπτεται από τον αποστολέα (source) στον παραλήπτη (destination).

1.4.4 ETHERNET

Είναι ένας ευρέως διαδεδομένος τρόπος διασύνδεσης μεταξύ υπολογιστών το οποίο χρησιμοποιεί ως μέσο μεταφοράς το καλώδιο (ομοαξονικό twisted pair ή οπτική ίνα) και μια κάρτα η οποία αναλαμβάνει τον ρόλο της διασύνδεσης και ορθής μεταφοράς των πακέτων δεδομένων που αποστέλλονται ανάμεσα στους υπολογιστές και τις συσκευές του δικτύου.

1.4.5 ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ

Χρόνος Απόκρισης (response time), ο οποίος σε ένα real-time ή interactive σύστημα, είναι ο χρόνος από την άφιξη μίας απαίτησης (request) μέχρι την ικανοποίησή της από το σύστημα.

1.4.6 LAN:(LOCAL AREA NETWORK)

Είναι μία ομάδα υπολογιστών και σχετικών συσκευών οι οποίες είναι συνδεδεμένες μεταξύ τους, επικοινωνούν και μπορούν να ανταλλάσσουν πληροφορίες ή πόρους του δικτύου (π.χ. εκτυπωτές που μπορεί να είναι συνδεδεμένοι επάνω στο δίκτυο)

1.4.7 MAN:(METROPOLITAN AREA NETWORK)

Είναι δίκτυα υψηλής ταχύτητας τα οποία περιορίζονται σε υπολογιστές οι οποίοι είναι συνδεδεμένοι σε 1.4.6 αποστάσεις όχι μεγαλύτερες από 80 χιλιόμετρα.

1.4.8 WAN:(WIDE AREA NETWORK)

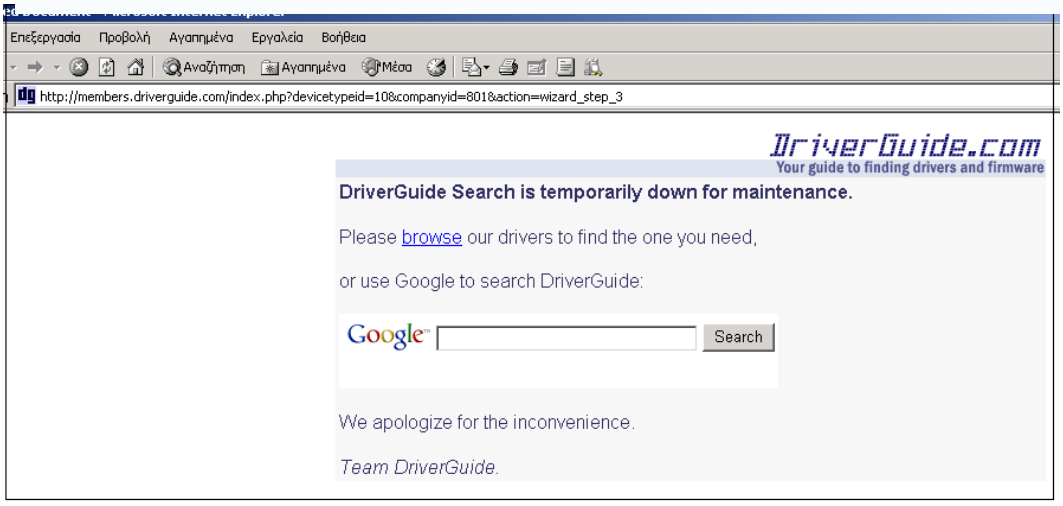
Είναι τα δίκτυα τα οποία συνδέουν χρήστες μεταξύ μεγάλων αποστάσεων οι οποίες πολλές φορές ξεπερνούν τα γεωγραφικά όρια των χωρών. Το Internet π.χ. είναι ένα WAN.

ΚΕΦΑΛΑΙΟ 2ο CLUSTERS & LOAD BALANCING

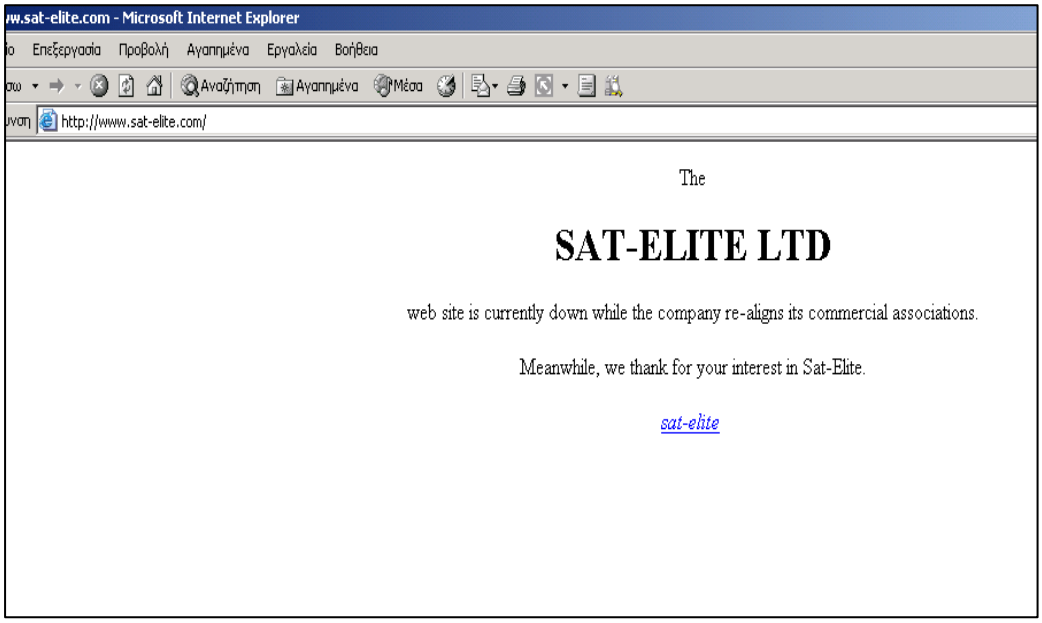
Στο παρελθόν ο μόνος τρόπος για να βελτιώσουμε τους πόρους σε ένα server ήταν η σταδιακή αναβάθμιση του, που κατέληγε στο τέλος στην αντικατάσταση του με ένα νέο λύση καθόλου μακροπρόθεσμη. Αναζητώντας μια πιο μακροπρόθεσμη λύση για ένα σύστημα με υψηλότερη διαθεσιμότητα (availability), μεγαλύτερη ευχρηστία (manageability, flexibility) και κλιμάκωση (scalability), δημιουργήθηκε ένα νέο πρότυπο, η χρήση clusters (ή server farm).

Cluster είναι μια ομάδα από ανεξάρτητους server οι οποίοι επικοινωνούν μεταξύ τους μέσο ενός δικτύου υψηλής ταχύτητας δημιουργώντας την αίσθηση ότι ενεργούν σαν ένας ανεξάρτητος server. Συνήθως κάθε server έχει τον δικό του δίσκο και μνήμη, υπάρχουν όμως περιπτώσεις όπου ένα cluster μπορεί να κάνει χρήση κοινών δίσκων και μνήμης.

Σε περίπτωση βλάβης σε ένα από τους server που αποτελούν το cluster το σύστημα μας συνεχίζει να είναι on-line αφού την δουλεία του ενός αναλαμβάνουν οι υπόλοιποι server μέσα στο cluster μέχρι να διορθωθεί η βλάβη. Επίσης στην περίπτωση που είναι αναγκαία η αναβάθμιση απλά αυξάνουμε τον αριθμό των server αυξάνοντας έτσι την συνολική ισχύ του cluster αυξάνοντας σημαντικά τον χρόνο απόκρισης (server reply) του συστήματος σε μια αίτηση χρήστη (client request). Έτσι σε δίκτυα με χρήση cluster θα είναι κάπως δύσκολο να δούμε μηνύματα όπως τα ακόλουθα :



Σχήμα 2.1 :Απεικόνιση σελίδας web σε περίοδο εργασιών



Σχήμα 2.2 :Απεικόνιση σελίδας web σε περίοδο εργασιών

Πως όμως μοιράζονται οι αιτήσεις μέσα στο cluster; Αυτό το πετυχαίνουμε με μια τεχνική που ονομάζεται καταμερισμός φορτίου (από εδώ και μετά load balancing) μοιράζοντας έτσι τις αιτήσεις μεταξύ των servers που αποτελούν το cluster. Η τεχνική αυτή εφαρμόζεται στον load balancer ή Dispatcher συσκευή που ενώνει το cluster με το δίκτυο και ο οποίος από την μία "καταλαβαίνει" πολλά υψηλού επιπέδου πρωτόκολλα ώστε να μπορεί να επικοινωνεί με τους server "έξυπνα" και από την άλλη "καταλαβαίνει" τα πρωτόκολλα επικοινωνίας ώστε να μπορεί να αλληλεπιδρά με το διαδίκτυο [5] έτσι ανάλογα με τον αλγόριθμο που χρησιμοποιεί διαχειρίζεται τις εισερχόμενες αιτήσεις και ελέγχει την κίνηση ανάμεσα στους servers. Ο load balancer συνήθως αποτελείται από ένα server (υπάρχουν και περιπτώσεις που τον αντικαθιστά και κάποιος router ο οποίος έχει το ανάλογο software) ο οποίος είναι ρυθμισμένος να δρα ως Virtual Server (έχοντας σαν IP διεύθυνση μια VIP διεύθυνση δηλαδή μια εικονική διεύθυνση) και στην συνέχεια είναι συνδεμένος με τους υπόλοιπους server μέσα στο cluster οι οποίοι έχουν ο κάθε ένας την δικιά του IP διεύθυνση και τις οποίες γνωρίζει ο load balancer. Έτσι όταν έρθει μια αίτησης ο load balancer "ξέρει" που να την προωθήσει. Επιπλέον ο Load balancer εποπτεύει συνεχώς

το cluster και σε περίπτωση βλάβης κάποιου server επιλέγει κάποιον άλλο για να συνεχίσει.

Συνοψίζοντας μπορούμε να πούμε ότι ο load balancer

- " Μοιράζει την εισερχόμενη στο cluster κίνηση (request) σε μικρότερες και αποφασίζει ποιος server θα απαντήσει σε κάθε μια.
- Παρακολουθεί τους διαθέσιμους server μέσα στο cluster και βεβαιώνει ότι ανταποκρίνονται σωστά, αν όχι τους βγάζει εκτός.
- Παρέχει διαθεσιμότητα παρέχοντας περισσότερες από μία μονάδες σε περίπτωση βλάβης.
- Παρέχει πιστοποίηση εισερχομένων και εξερχόμενων δεδομένων κάνοντας πράξεις όπως διάβασμα URLs, ανακοπή πορείας cookies." [4]

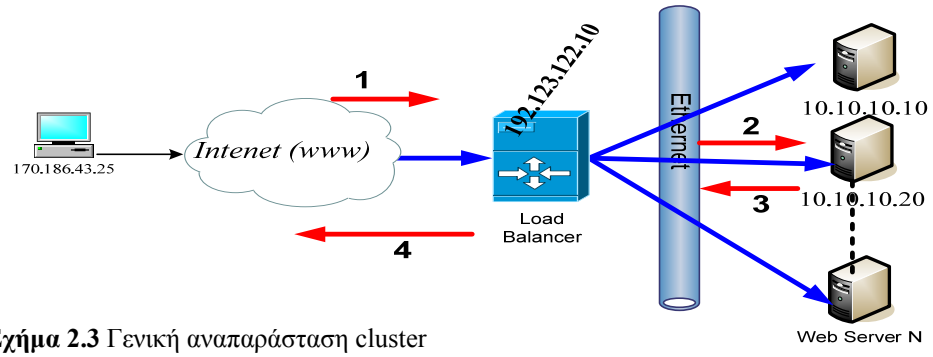
2.1 ΤΡΟΠΟΣ ΕΠΙΚΟΙΝΩΝΙΑΣ

Ας εξετάσουμε πως αυτά τα δύο συνδέονται μεταξύ τους. Όπως φαίνεται στο σχήμα 1.1 ο load balancer βρίσκεται μετά από τον router ο οποίος μα συνδέει με το διαδίκτυο και πριν από τους server οι οποίοι μπορεί να είναι συνδεμένοι απευθείας στον load balancer η μέσο κάποιου switch. Το δίκτυο από τον load balancer και μετά μέχρι και την τελευταία συσκευή που είναι συνδεμένη εμφανίζεται σαν ένας μεμονωμένος server και έτσι όπως κάθε server θα πρέπει να έχει την δικιά του IP διεύθυνση την οποία ονομάζουμε φανταστική διεύθυνση (Virtual IP – VIP) και την οποία ρυθμίζουμε μέσα στον load balancer.

Έστω ότι το δίκτυο του σχήματος είναι πραγματικό και έχει τις ρυθμίσεις του παρακάτω πίνακα και ονομάζεται www.spapada.com.

VIP Διεύθυνση	Αριθμός Server στο Cluster	IP Διεύθυνση
192.123.122.10	1	10.10.10.10
	2	10.10.10.20
	3	10.10.10.30
	4	10.10.10.40

Πίνακας 2.1 Ρυθμίσεις δικτύου



Σχήμα 2.3 Γενική αναπαράσταση cluster

	IP Διεύθυνση Αποστολέα	IP Διεύθυνση Προορισμού
Βήμα 1	170.186.43.25	192.123.122.10
Βήμα 2	192.123.122.20	10.10.10.20
Βήμα 3	10.10.10.10	192.123.122.10
Βήμα 4	192.123.122.10	170.186.43.25

Πίνακας 2.2 Network address translation (NAT πίνακας)

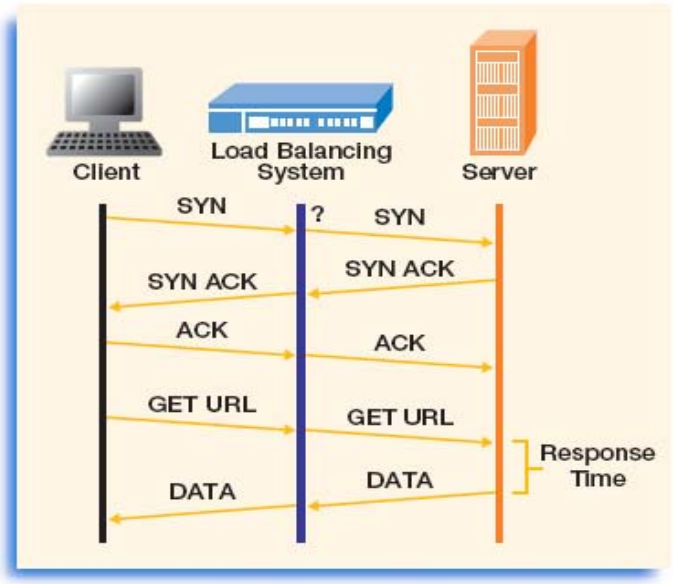
Βήμα 1ο ο load balancer λαμβάνει την αίτηση από τον client που προορίζεται για την ιστοσελίδα www.spapada.com και αντιστοιχεί στην IP διεύθυνση 192.123.122.10,

Βήμα 2ο αντί να εξυπηρετήσει ο load balancer την αίτηση, αντιγράφει το πακέτο (αίτηση) και ο load balancer γίνεται αποστολέας προς κάποιον server που επιλέγει μέσα από το cluster.

Βήμα 3ο ο server εξυπηρετεί την αίτηση και απαντάει πίσω στον load balancer.

Βήμα 4ο ο load balancer επιστρέφει την απάντηση πίσω στον client.

Έτσι ο χρήστης έχει την εντύπωση ότι του απάντησε κάποιος server αλλά η μόνη επικοινωνία που είχε ήταν με τον load balancer. Αυτός ο τρόπος επικοινωνίας είναι ο πιο συχνός, υπάρχει όμως και τεχνική όπου ο real server δεν απαντάει στην αίτηση μέσο του load balancer αλλά απευθείας στον χρήστη (DSR-Direct Server Return).



Σχήμα 2.4 Ανταλλαγή μηνυμάτων μεταξύ client και server

Πιο αναλυτικά ο load balancer καταγράφει τον χρόνο απόκρισης του server σε κάθε συναλλαγή και δημιουργεί ένα πίνακα. Στο παραπάνω σχήμα περιγράφεται η κίνηση μεταξύ κάποιου client και ενός server/load balancer (ο οποίος ουσιαστικά αντιπροσωπεύει μια ιστοσελίδα), ο client προσπαθεί να επικοινωνήσει με την ιστοσελίδα πληκτρολογώντας την VIP διεύθυνση της συγκεκριμένης ιστοσελίδας. Ο load balancer εκτελεί NAT για αποκομίσει την αίτηση συγχρονισμού (SYN) και να δημιουργήσει μια ακολουθία TCP[6] μεταξύ του load balancer και του client. Έπειτα ο load balancer δημιουργεί μια δεύτερη TCP επικοινωνία με κάποιον server (εξαρτάται από τον αλγόριθμο καταμερισμού φορτίου που θα υπάρχει) μέσα στο cluster και προωθεί την αίτηση συγχρονισμού (SYN). Ο server απαντάει με μια νέα TCP σύνδεση και στέλνει πίσω μια απάντηση συγχρονισμού (SYN/ACK) στο load balancer, αυτό επαναλαμβάνεται για να υπάρχει επικοινωνία και συγχρόνως μεταφέρονται τα πακέτα δεδομένων του client προς τον server και αντίθετα.

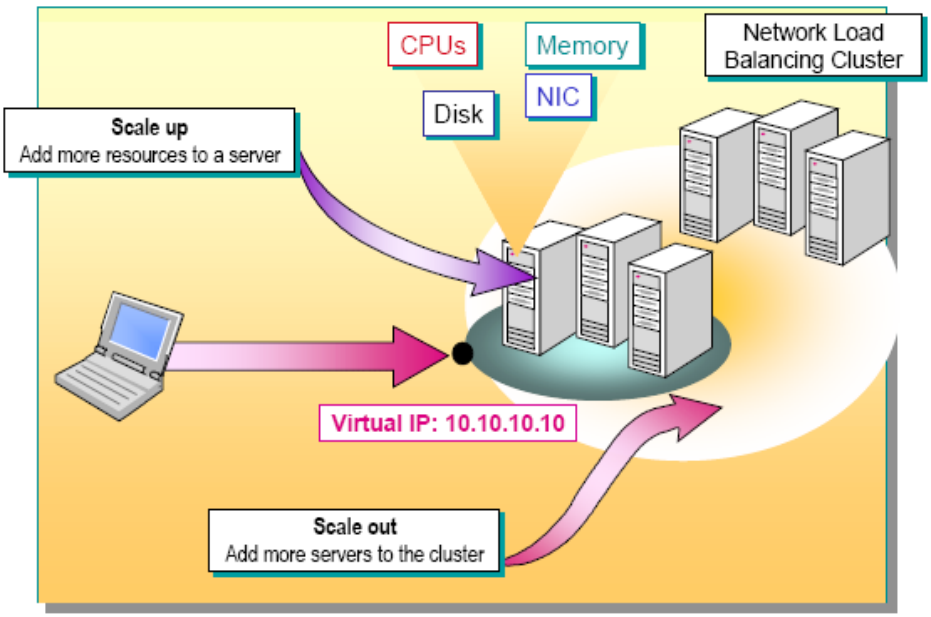
[6] TCP = Transmission Control Protocol

2.2 ΓΙΑΤΙ ΝΑ ΧΡΗΣΙΜΟΠΟΙΗΣΟΥΜΕ LOAD BALANCER

Κάνοντας χρήση κάποιου load balancer στο δίκτυο θα καταφέρουμε να έχουμε:

Κλιμάκωση (scalability).

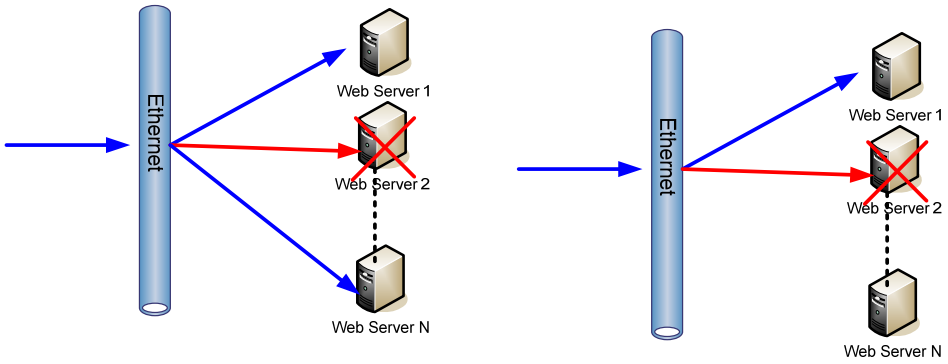
[5]Λόγω ότι ο load balancer κατανέμει τις αιτήσεις πελατών (client request) σε όλους τους servers του cluster είναι ο μόνος που πρέπει να έχει συνολική υπολογιστική ισχύ πολύ μεγαλύτερη από τους υπόλοιπους. Αν ο αλγόριθμος που χρησιμοποιεί είναι ο πιο αποδοτικός, τότε θα μοιράζει τις αιτήσεις όμοια σε όλους τους servers. Αυτό σημαίνει ότι για να αυξήσουμε την υπολογιστική ισχύ του δικτύου μας δεν χρειάζεται να έχουμε μεγάλης ισχύος servers, αλλά μικρότερης ισχύος και κάθε φορά που θέλουμε περισσότερο απλά προσθέτουμε ακόμα ένα η περισσότερους (scaling out), λύση πολύ πιο οικονομική. Επιπλέον μπορούμε να αυξήσουμε την ισχύει του κατά μέρους server απλά προσθέτοντας μνήμη, μεγαλύτερο δίσκο η κάποιον πιο γρήγορο επεξεργαστή (scale up)



Σχήμα 2.5 Κλιμάκωση, Scale out , Scale up

Διαθεσιμότητα (availability/Redundancy).

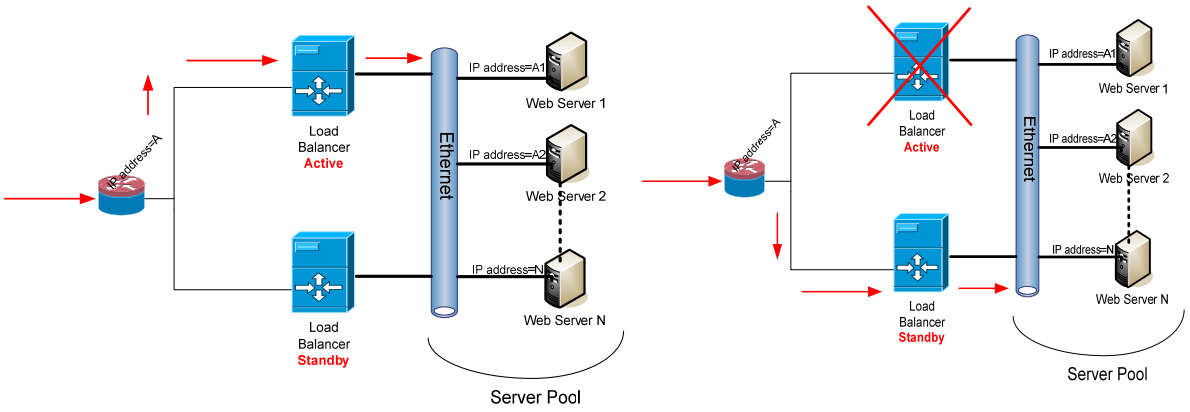
Ο load balancer συνεχώς ελέγχει την κατάσταση του κάθε server και των εφαρμογών που εκτελούνται κάθε στιγμή. Σε περίπτωση που κάποιος server δεν ανταποκριθεί, φανεί ότι δεν είναι on line η γενικά σε κάθε περίπτωση δυσλειτουργίας και βλάβης, θα βγει εκτός cluster αναλαμβάνοντας οι υπόλοιποι να απαντήσουν στις αιτήσεις μέχρι να λυθεί το πρόβλημα και επανέλθει στο cluster.



Σχήμα 2.6 Διαθεσιμότητα, επίπεδο server

Το παραπάνω σχήμα παρουσιάζει δύο περιπτώσεις (α) Όταν σταματήσει να λειτουργεί ο server 2, ο φόρτος εργασίας κατανέμεται ομοιόμορφα στους υπόλοιπους server του cluster, (β) Όταν σταματήσει να λειτουργεί ο server 2, ο φόρτος εργασίας μεταφέρεται όλος σε κάποιον άλλο server μέσα στο cluster.

Σε μεγάλα δίκτυα για ακόμη μεγαλύτερη διαθεσιμότητα υπάρχουν δύο load balancer που σε περίπτωση βλάβης του ενός να αναλάβει ο άλλος για να συνεχίσει την σωστή αποστολή αιτήσεων (Active-Standby Scenario).



Σχήμα 2.7 Διαθεσιμότητα, επίπεδο load balancer

το ποιο συνηθισμένο πρωτόκολλο είναι το Virtual Router Redundancy Protocol (VRRP), γενικότερα όλα τα σχετικά πρωτόκολλα τελειώνουν σε xxRP όπως Extreme Standby Router Protocol (ESRP) και Cisco's Hot Standby Routing Protocol (HSRP).

Ευχρηστία (manageability/flexibility).

[5] Αν κάποιος χρειαστεί να αναβάθμιση είτε στο λογισμικό είτε στο υλικό σίγουρα θα χρειαστεί να βγάλει εκτός λειτουργίας κάποιον η πολλές φορές κάποιους server. Είναι μια διαδικασία η οποία απαιτεί αρκετό χρόνο και η οποία συνήθως προγραμματίζεται σε ώρες μη αιχμής για να υπάρχει όσο το δυνατό λιγότερη δυσλειτουργία του ιστό χώρου. Είναι όμως εταιρίες οι οποίες πραγματικά δεν έχουν ώρες μη αιχμής , για παράδειγμα το Google το οποίο λειτουργεί παγκόσμια και πάντοτε κάπου υπάρχουν χιλιάδες χρηστές που το χρησιμοποιούν. Έτσι βάζοντας ένα load balancer μπροστά από τους servers δρομολογούμε τις αιτήσεις τους προς αναβάθμιση server κάπου αλλού μέσα στο cluster μέχρι το πέρας των εργασιών χωρίς να μειώσουμε την απόδοση της εταιρείας.

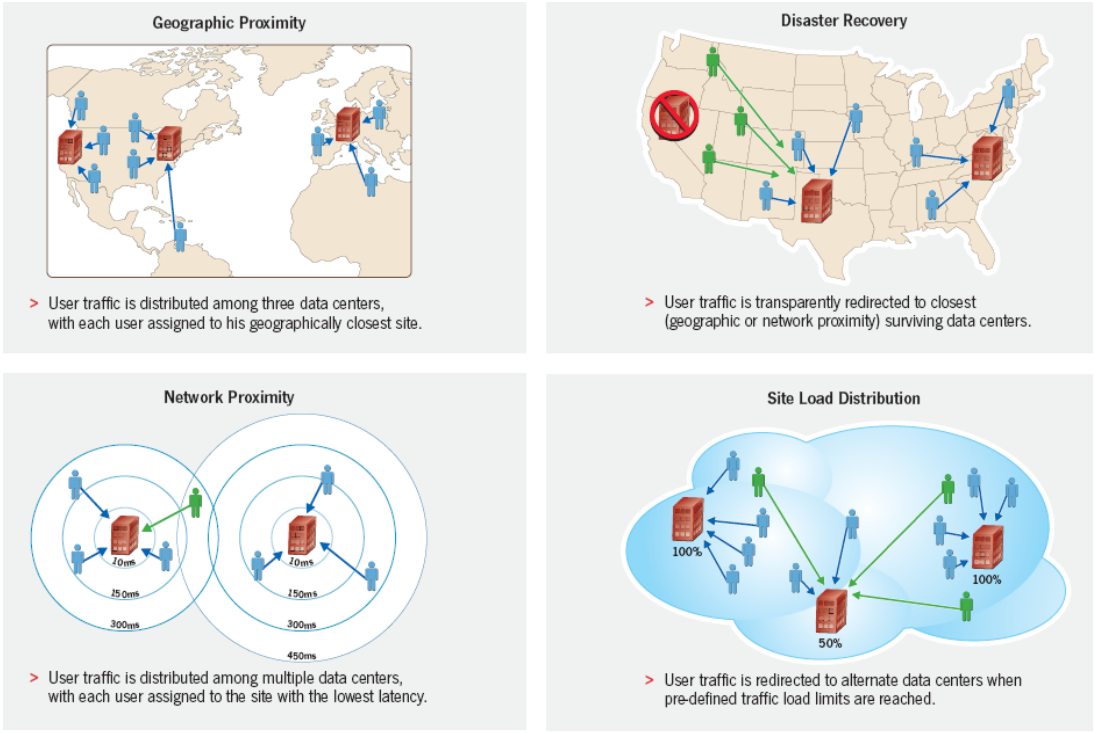
Βοηθούν στον διαχωρισμό (decoupling) υπηρεσιών σε όλους τους servers, έστω ότι έχουμε δέκα servers και θέλουμε να έχουμε δυο υπηρεσίες web (HTTP) στους δύο και File Transfer Protocol (FTP) στους υπόλοιπους οκτώ, αν δεν είχαμε load balancer θα έπρεπε να έχουμε DNS round-robin για τους δύο με HTTP και DNS round-robin στους υπόλοιπους οκτώ, το μεγαλύτερο πρόβλημα όμως θα δημιουργηθεί σε περίπτωση που αυξηθεί η ανάγκη για παράδειγμα web (HTTP), με την χρήση DNS round-robin θα πρέπει να ρυθμίσουμε από την αρχή το δίκτυο μας να στέλνει αιτήσεις σε περισσότερους από δυο server για web και στους υπόλοιπους FTP. Διαφορετικά ο load balancer δεν χρειάζεται καμία ρύθμιση, ανάλογα τις διαφορετικές ανάγκες δρομολογεί (ανάλογα τον αλγόριθμο) τις αιτήσεις σε όσους server και αν χρειαστεί.

Ασφάλεια (security).

Όπως αναφέραμε ο load balancer είναι πριν το cluster, έτσι στο διαδίκτυο φαίνεται μόνο η VIP και όχι η IP του κάθε server κάνοντας πιο δύσκολη την πρόσβαση μέσα στο cluster προστατεύοντας έτσι το δίκτυο από κάθε τύπου κακόβουλη επίθεση (ιούς, hackers) ανάλογα βέβαια τον τύπο του load balancer και των εφαρμογών που χρησιμοποιούνται.

2.3 GLOBAL LOAD BALANCING (GSLB)

Το global load balancing έχει ακριβός τις ίδιες αρχές και χρησιμοποιεί τους ίδιους αλγόριθμους καταμερισμού φορτίου με το load balancing με την μόνη διαφορά ότι δεν προορίζεται για τοπικά δίκτυα (Local area networks-LAN) αλλά για παγκόσμια δίκτυα (Wide area networks-WAN), το cluster δηλαδή το αποτελούν server οι οποίοι βρίσκονται σε διαφορετικές περιοχές ανά τον κόσμο. Όπως φαίνεται και στο παρακάτω σχήμα [14] η χρήση GSLB είναι εξίσου σημαντική όσο και η εφαρμογή load balancing σε ένα LAN δίκτυο



Σχήμα 2.8 Global Load balancing cases

Πάνω αριστερά : κάθε χρήστης εξυπηρετείτε από server που βρίσκεται στην γεωγραφική περιοχή του. Για παράδειγμα αν κάποιος πληκτρολογήσει www.google.com (αμερικάνικος server) από την Ελλάδα ο browser δεν θα τον δρομολογήσει εκεί αλλά στον Ελληνικό server www.google.gr.

Πάνω δεξιά : Σε περίπτωση που κάποιος server βγει εκτός λειτουργίας οι αιτήσεις δρομολογούνται στο κοντινότερο server του cluster. Για παράδειγμα αν κάποιος πληκτρολογήσει www.google.gr από την Ελλάδα και ο ελληνικός

server είναι εκτός λειτουργίας θα τον δρομολογήσει στο πιο κοντινό server η σε αυτόν τον server που μπορεί εκείνη την στιγμή να τον εξυπηρετήσει.

2.4 ΜΗΧΑΝΙΣΜΟΙ / ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ

Υπάρχουν πολλοί τρόποι για τον τρόπο με τον οποίο θα ενσωματώσουμε ένα load balancer μέσα στο δίκτυο μας. Μια γενική περιγραφή φαίνεται στο παρακάτω σχήμα [4]

IP Configuration	Return Path	Physical Connectivity
Flat-based	Bridge-path	One-armed
NAT-based	Route-path	Two-armed
	DSR	

Πίνακας 2.3 Μηχανισμοί Load balancing

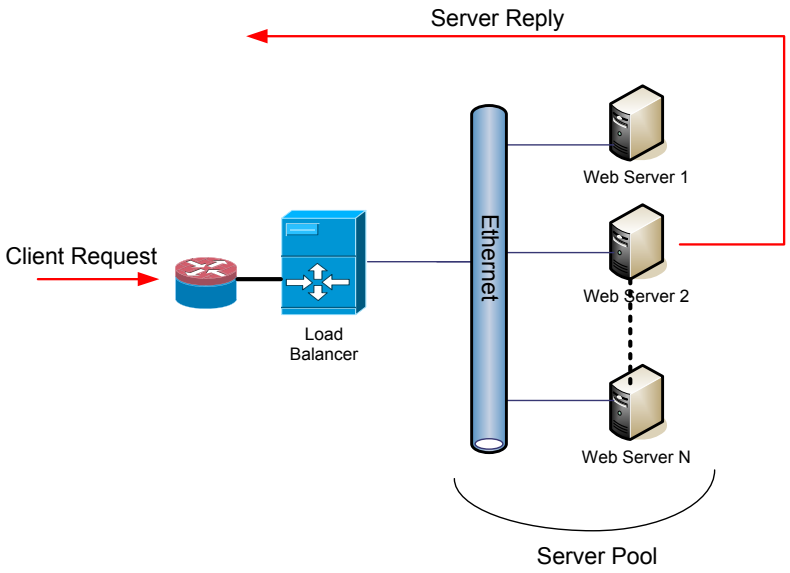
- Flat-based : Οι VIPs και οι real severs είναι στο ίδιο υπό δίκτυο (subnet).
- NAT-based : οι VIPs και οι real severs δεν είναι στο ίδιο υπό δίκτυο (subnet).
- Bridge-path : Ο load balancer ενεργεί σαν bridge (layer 2).
- Route-path : Ο load balancer ενεργεί σαν router (layer 3).
- DSR-Direct Server Return : Όταν ο server είναι ρυθμισμένος έτσι ώστε να παρακάμπτει τον load balancer.

Ανάλογα με πια τεχνολογία χρησιμοποιεί δρα ως switch και επεξεργάζεται μόνο τις εισερχόμενες πληροφορίες ή σαν network gateway οπότε επεξεργάζεται τις εισερχόμενες και τις εξερχόμενες πληροφορίες. Οι μηχανισμοί διαφοροποιούνται ανάλογα σε πιο στρώμα στο μοντέλο OSI (Open System Interconnection, Διασύνδεση Ανοικτών Συστημάτων) ενεργούν, έτσι ταξινομούνται στις παρακάτω κατηγορίες:

2.4.1 L4/2 CLUSTERING

[11] Στην περίπτωση που έχουμε L4/2 Clustering (Layer four switching with layer two packet forwarding), η IP διεύθυνση μοιράζεται από τον load

balancer και την ομάδα των servers (pool of servers) με την χρήση πρωτογενούς (primary) και δευτερογενούς (secondary) IP διεύθυνσης.



Σχήμα 2.9 L4/2 Clustering

Έστω ότι ένας χρήστης στέλνει μια αίτηση με IP διεύθυνση προορισμού Α, ο δρομολογητής θα στήλη το πακέτο στον load balancer και ανάλογα με πια μέθοδο καταμερισμού φορτίου χρησιμοποιεί αποφασίζει ποίος server θα λάβει το πακέτο. Έπειτα αλλάζετε η MAC διεύθυνση του πακέτου ίδια με αυτή του server που θα λάβει το πακέτο, και προωθείται . Τέλος ο επιλεγμένος server δέχεται την αίτηση, την επεξεργάζεται και απαντάει απευθείας στον χρήστη.

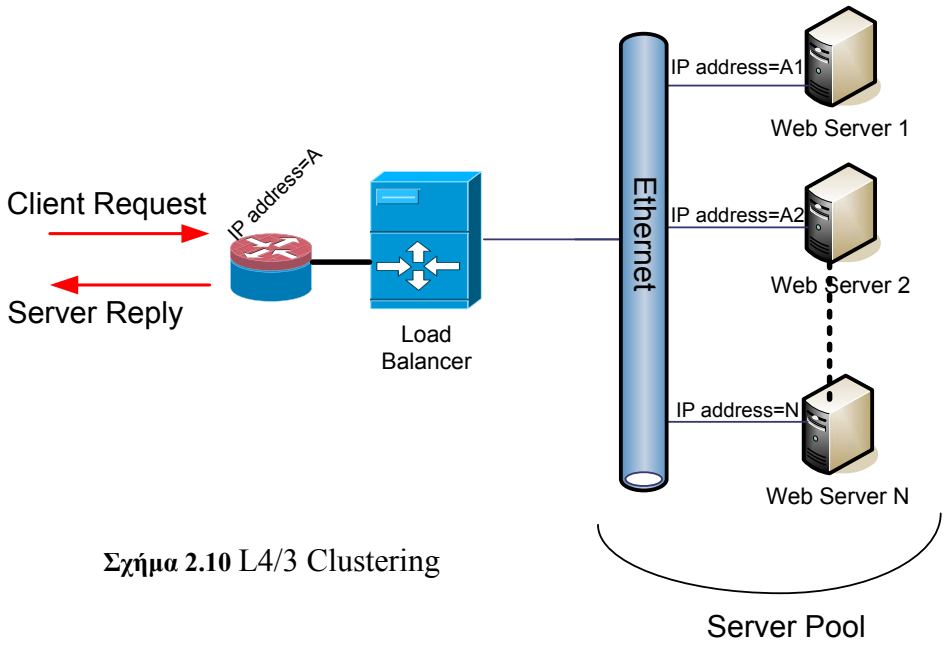
Μερικά παραδείγματα L4/2 Clustering είναι:

- ONE-IP : αναπτύχθηκε στα εργαστήρια της Bell.
- E-Network Dispatcher : IBM. 1996.

2.4.2 L4/3 CLUSTERING

[11] Η τοπολογία του δικτύου στην περίπτωση που έχουμε L4/2 Clustering (Layer four switching with layer three packet forwarding) είναι ακριβώς η

ίδια με πριν, η διαφορά είναι ότι εδώ οι servers έχουν διαφορετική διεύθυνση από τον load balancer του οποίου η διεύθυνση είναι η gateway για το διαδίκτυο.



Σχήμα 2.10 L4/3 Clustering

Έστω ότι ένας χρήστης στέλνει μια αίτηση με IP διεύθυνση προορισμού Α, ο δρομολογητής θα στήλη το πακέτο στον load balancer και ανάλογα με πια μέθοδο καταμερισμού φορτίου χρησιμοποιεί αποφασίζει ποίος server θα λάβει το πακέτο. Έστω ότι επιλέγεται ο server 1, τότε ο load balancer ξαναγράφει την IP διεύθυνση προορισμού σαν A1 και στέλνει το πακέτο στον server 1. Ο server 1 λαμβάνει το πακέτο, το επεξεργάζεται και απαντάει στον χρήστη μέσο του load balancer του οποίου η διεύθυνση είναι η gateway.

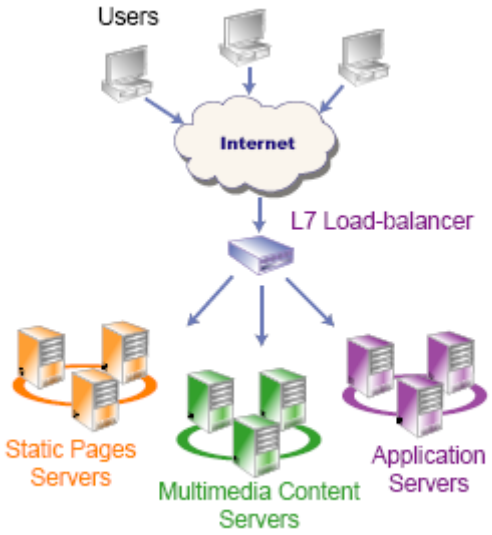
Ένα παράδειγμα L4/23Clustering είναι:

- Local Director : Cisco Systems.

2.4.3 L7 CLUSTERING

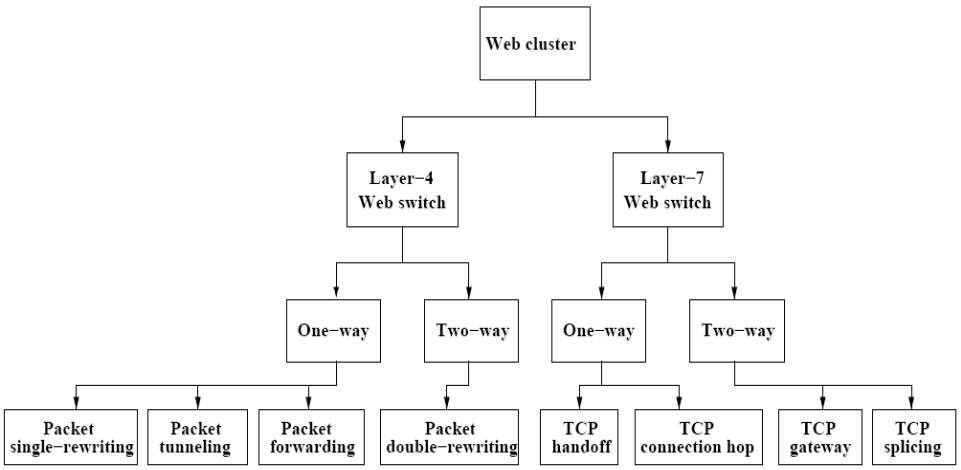
[25]Ο καταμερισμός φορτίου σε επίπεδο 4 (L4) είναι μια πολύ γρήγορη λύση για να διευθύνουμε servers οι οποίοι έχουν εφαρμογές όπως web service,

FTP, DNS, e-mail, και είναι πολύ αποτελεσματικό σε cluster στα οποία δεν έχουμε περαιτέρω ομαδοποιήσει κάποιος server. Σε περίπτωση όμως που χρειαστεί να δημιουργήσουμε ένα υπό cluster μέσα στο αρχικό για καλύτερη ευελιξία θα πρέπει να έχουμε L7 (Layer seven switching layer two packet forwarding) που μας επιτρέπει να ομαδοποιήσουμε εύκολα servers οι οποίοι εξυπηρετούν ίδιες λειτουργίες.



Σχήμα 2.11 Ομαδοποίηση servers Με την χρήση L7 clustering

[20] Οι τεχνικές του clustering που βασίζονται σε επίπεδο 4 και 7 (L4, L7) μπορούν να κατηγοριοποιηθούν περαιτέρω σύμφωνα με την κατεύθυνση μετακίνησης της πληροφορίας μεταξύ του client και του server που εξυπηρετεί και φαίνονται στο παρακάτω σχήμα [20]:



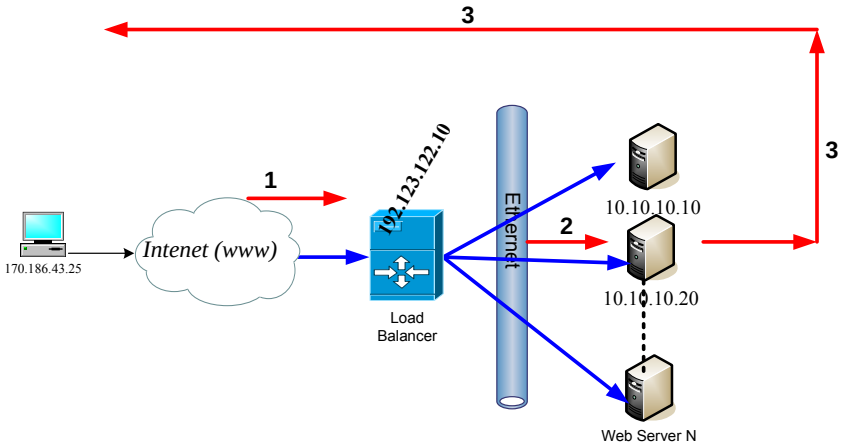
Σχήμα 2.12 Υπό κατηγορίες L4 και L7

	L4/2	L4/3	L7
Mechanism	Link-layer address translation	Network address translation	Content-based routing
Flows	Incoming only	Incoming/outgoing	Varies
Fault tolerance	Yes	Yes	Yes
Restrictions	Physical interface on every network with server incoming traffic passes through dispatcher	Dispatcher lies between client and server	All incoming traffic passes through dispatcher all outgoing as well, if caching
Performance	Thousands of c/s hundreds of Mb/s	Hundreds of chip/s tens of Mb/s	Tens of thousands of chip/s Gb/s
Bottleneck	Connection dispatching	Integrity code calculations	Connections dispatching, dispatcher complexity
Advantage	Simple	Flexible	Server specialization, caching

Πίνακας 2.4 Σύνοψη τεχνολογιών [11]

2.4.4 DIRECT SERVER RETURN – DSR

Η συγκεκριμένη τεχνική παρακάμπτει τον load balancer και ο real server απαντάει απευθείας στον client αυξάνοντας έτσι την απόδοση του load balancer αφού επεξεργάζεται το μισό από τις αιτήσεις (μόνο τις εισερχόμενες), πιο αναλυτικά περιγράφεται στο κεφάλαιο 3



Σχήμα 2.13 Direct server return

	IP Διεύθυνση Αποστολέα	IP Διεύθυνση Προορισμού
Βήμα 1	170.186.43.25	192.123.122.10
Βήμα 2	192.123.122.20	10.10.10.20
Βήμα 3	10.10.10.10	170.186.43.25

Πίνακας 2.5 DSR NAT table

2.4.5 SOFTWARE BASED LOAD BALANCING

Η συγκεκριμένη τεχνική βασίζεται στην εγκατάσταση κάποιου συγκεκριμένου λογισμικού, κατασκευασμένο αποκλειστικά για load balancing, σε κάθε server μέσα στο cluster.

2.4.6 HARDWARE BASED LOAD BALANCING

2.4.6.1 Server-based Load balancers

Βασίζεται στην χρήση κάποιου server / dispatcher ο οποίος χρησιμοποιεί κάποιο ειδικευμένο λειτουργικό όπου λειτουργεί κάποια εφαρμογή (software) καταμερισμού φορτίου.

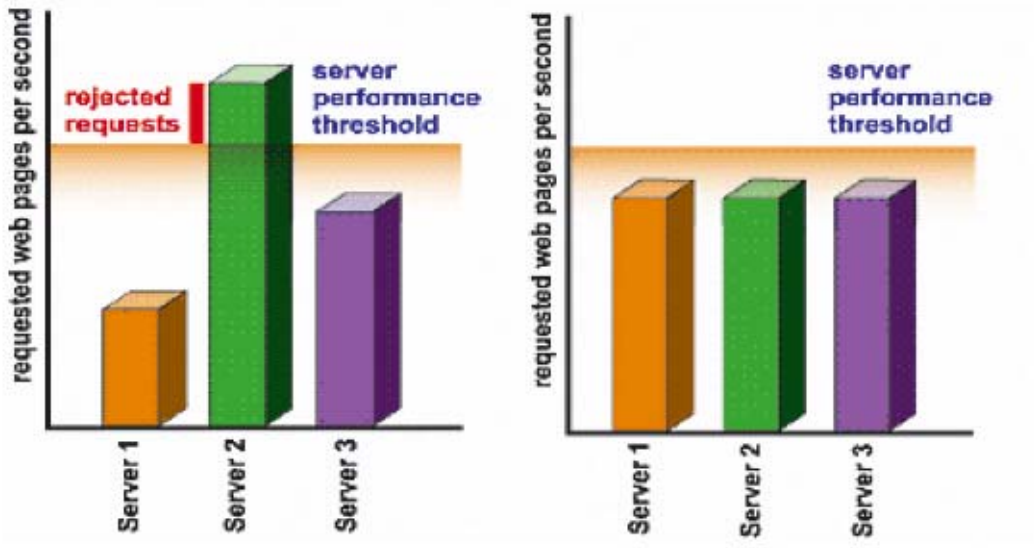
2.4.6.2 Switched-based Load balancers

Είναι πολύ απλή μέθοδος καταμερισμού φορτίου και δεν ενδείκνυται για μεγάλα δίκτυα. Γίνεται χρήση κάποιου switch όπου υπάρχει εγκατεστημένο κά-

ποιο ολοκληρωμένο κύκλωμα το οποίο το μόνο που κάνει είναι να αντιγράφει τα πακέτα (packet rewriting) και να δρομολογεί ξανά.

ΚΕΦΑΛΑΙΟ 3^{ο} ΑΛΓΟΡΙΘΜΟΙ

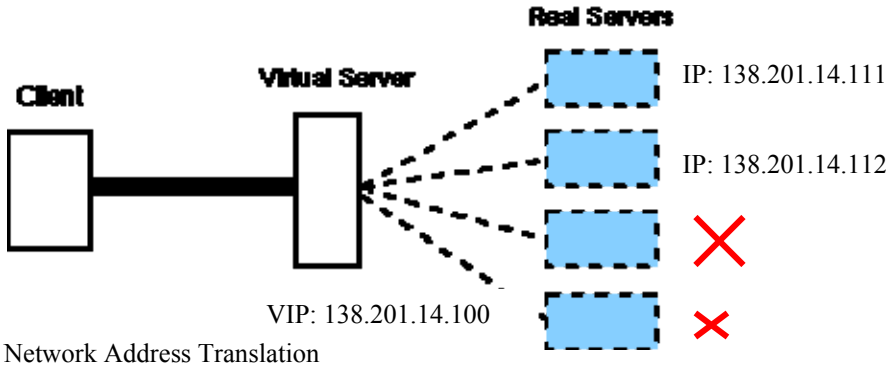
Οποιοδήποτε cluster η server farm στο οποίο δεν γίνεται κατάλληλος καταμερισμός φορτίου εργασίας (load balancing) στις αιτήσεις υπάρχει ο κίνδυνος να τις απορρίπτει γιατί είναι πολύ πιθανό κάποιος από τους server να δουλεύουν στο ανώτερο επίπεδο τις απόδοσης του και κάποιος άλλος να είναι σε πολύ χαμηλό επίπεδο απόδοσης (σχήμα 3.1). Έτσι θα πρέπει να επιλέξουμε τον κατάλληλο αλγόριθμο καταμερισμού φορτίου σύμφωνα με την αρχιτεκτονική του cluster μας και του δικτύου μας ο οποίος θα μας παρέχει στο δίκτυο την καλύτερη δυνατή λειτουργία και η οποία θα πρέπει να είναι όσο πιο κοντά γίνεται στο σχήμα 3.2. Παγκοσμίως χρησιμοποιούνται δεκάδες αλγόριθμοι καταμερισμού φορτίου, εμείς καταγράψαμε τους πιο γνωστούς και τους παρουσιάζουμε στις επόμενες σελίδες.



Σχήμα 3.1 Διάγραμμα διαφοράς φόρτου εργασίας

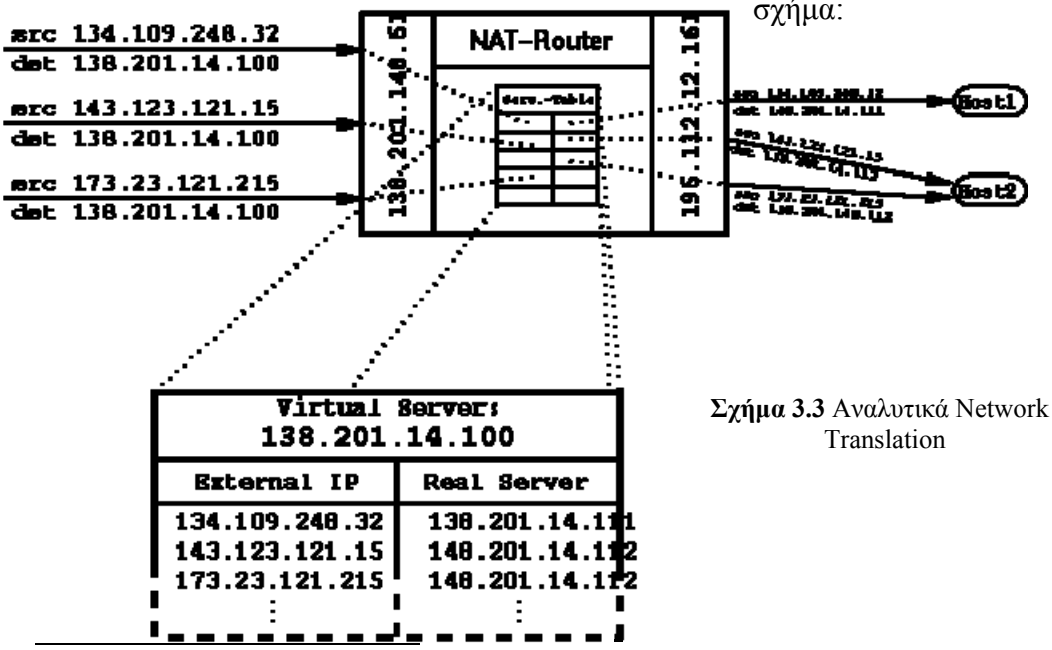
3.1 NETWORK ADDRESS TRANSLATION (NAT)

Η συγκεκριμένη τεχνική δημιουργήθηκε περισσότερο για εξοικονόμηση IP διευθύνσεων και στην συνέχεια κατά την ανάπτυξη των cluster έγινε ο η βασική μονάδα για τον καταμερισμού φορτίου εργασίας. Ο τρόπος λειτουργίας είναι απλός, ο load balancer (virtual server) μετονομάζει την VIP διεύθυνση που έχει σε μία IP διεύθυνση κάποιου πραγματικού server μέσα στο cluster. Έστω ότι έχουμε την τοπολογία του παρακάτω σχήματος[7] :



Σχήμα 3.2 Network Address Translation

Όπως φαίνεται λειτουργούν οι δύο από τους τέσσερις server, τώρα κάθε νέα αίτηση θα εξυπηρετείται πρώτα από τον virtual server ο οποίος θα προωθεί ξανά (ανάλογα με ποίον αλγόριθμο χρησιμοποιεί) τις αιτήσεις σε κάποιον από τους ενεργούς server, πιο αναλυτικά η διαδικασία παρουσιάζεται στο παρακάτω σχήμα:



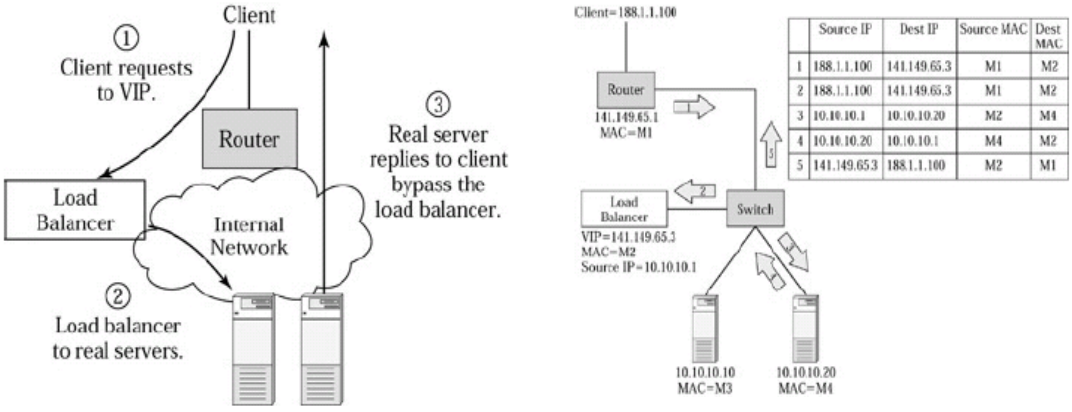
Σχήμα 3.3 Αναλυτικά Network Address Translation

[7] Από Ιστοσελίδα Michael Hasestain

Ο συγκεκριμένος αλγόριθμος ονομάζεται Destination NAT (NAT προορισμού η half-NAT) αφού ο load balancer μετονομάζει την VIP διεύθυνση με την IP διεύθυνση κάποιου server μέσα στο cluster.

[5]Υπάρχουν όμως περιπτώσεις όπου είναι αναγκαίο να παραβλεφθεί ο load balancer και ο πραγματικός server να απαντήσει απευθείας στον χρήστη (client), αυτό ονομάζεται source NAT (η Full-NAT). Εδώ ο load balancer αλλάζει την IP του αποστολέα σε όλα τα πακέτα / αιτήματα που λαμβάνει σε κάποια IP διεύθυνση που έχει ρυθμιστή από τον administrator η οποία ονομάζεται source IP (μπορεί να είναι ίδια με την VIP) πριν προωθήσει τις αιτήσεις στον real server. Όταν ο real server πάρει την αίτηση έχει σαν αποστολέα τον load balancer εξαιτίας της Source IP και μόλις ο real server εξυπηρετήσει θα στήλη την απάντηση στον load balancer ο οποίος απλά θα μεταφράσει την source IP στην πίσω στην IP του client για να λάβει την απάντηση, παρακάμπτοντας έτσι τον load balancer.

Το παρακάτω σχήμα [5] παρουσιάζει αναλυτικά όλα τα βήματα:



Σχήμα 3.4 Βήματα NAT, παράκαμψη load balancer

Μια τελευταία παραλλαγή είναι το enhanced NAT όπου ο load balancer "τρέχει" προοδευτικά πρωτοκολλά. Οι προηγούμενες εκδόσεις NAT άλλαζαν μόνο την IP διεύθυνση στο επικεφαλίδα (header) του πακέτου, όμως κάποια πρωτόκολλα περιέχουν διευθύνσεις και πληροφορίες για τις θύρες (port info) όπου πρέπει να γίνουν αλλαγές μαζί με το επικεφαλίδα (header) του πακέτου

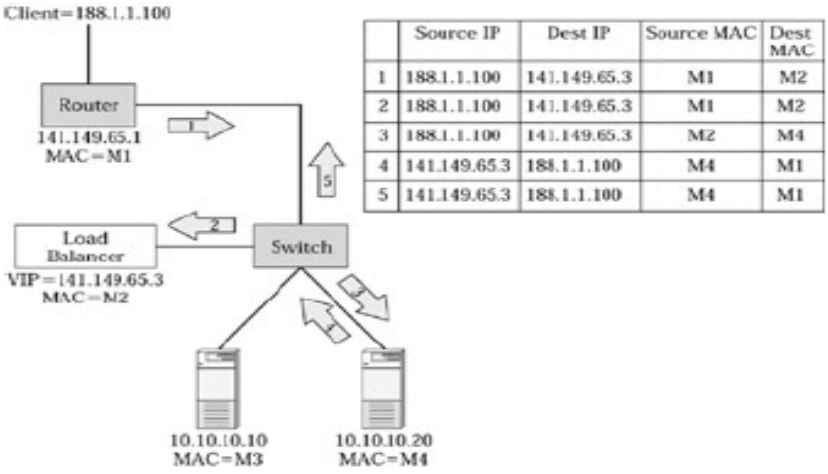
πράγμα το οποίο απαιτεί ένα πιο εμπεδωμένο (embedded) αλγόριθμο που να μπορεί να τα αντιμετωπίσει.

Με την χρήση NAT οι sever μέσα στο cluster μπορούν να χρησιμοποιούν όποιο λειτουργικό σύστημα θέλουν χωρίς καμία τροποποίηση, επιπλέον κάνουμε οικονομία στις IP διευθύνσεις αφού χρειαζόμαστε μόνο μια δημόσια (public IP) και όλες οι υπόλοιπες που αντιπροσωπεύουν τους πραγματικούς server είναι ιδιωτικές (private IP).

3.2 DIRECT SERVER RETURN (DSR)

Στις περισσότερες τεχνικές καταμερισμού φορτίου η απάντηση του client περνάει μέσα από τον load balancer προκαλώντας του συμφόρηση (bottleneck) ακόμα και σε περιπτώσεις που δεν είναι απαραίτητο αφού πρέπει να επεξεργαστεί συγχρόνως και τις εισερχόμενες και τις εξερχόμενες αιτήσεις . Αντίθετα ο Direct Server Return έχει το πλεονέκτημα ότι απαντά απευθείας στον client παρακάμπτοντας τον load balancer, μειώνοντας έτσι τον αριθμό των πακέτων που πρέπει να επεξεργαστεί στο μισό.

Για να γίνει αυτό, [5] ο load balancer είναι ρυθμισμένος έτσι ώστε μεταφράζει την MAC διεύθυνση (MAC address) του παραλήπτη στο εισερχόμενο πακέτο και η IP διεύθυνση του παραμένει ίδια με την VIP διεύθυνση του load balancer. Προκειμένου όμως η αίτησης να φτάσει στον παραλήπτη (ένα από τους server μέσα στο cluster) βασιζόμενη μόνο στην MAC διεύθυνση θα πρέπει όλοι οι server στο cluster να είναι στο ίδιο layer 2 domain όπως και ο load balancer. Όταν ο server λάβει το πακέτο βλέπει ότι η IP διεύθυνση παραλήπτη δεν είναι η δικιά του αλλά η VIP του load balancer, προκείμενου λοιπόν να μην τον απορρίψει ρυθμίζουμε σε κάθε server την loopback IP διεύθυνση να είναι ίδια με την VIP διεύθυνση του load balancer και έτσι το αποδέχεται σαν να ήταν αυτός ο αρχικός παραλήπτης.



	Source IP	Dest IP	Source MAC	Dest MAC
1	188.1.1.100	141.149.65.3	M1	M2
2	188.1.1.100	141.149.65.3	M1	M2
3	188.1.1.100	141.149.65.3	M2	M4
4	141.149.65.3	188.1.1.100	M4	M1
5	141.149.65.3	188.1.1.100	M4	M1

Σχήμα 3.5 Direct server return

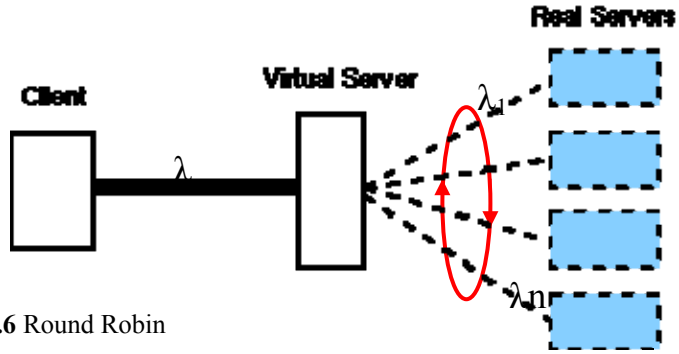
Για να γίνει πιο σαφές, το παρακάτω σχήμα [5] περιγράφει αναλυτικά όλα τα βήματα:

[5]Όπως φαίνεται ο load balancer δεν αλλάζει την IP με την VIP διεύθυνση αλλά αλλάζει την MAC προορισμού ίδια με την MAC του server που θα δεκτή την αίτηση. Καθώς το switch είναι λειτουργεί στο επίπεδο 2 (Layer 2) απλά προωθεί το πακέτο στον server με την επιλεγμένη MAC διεύθυνση, ο server δέχεται το πακέτο και η VIP διεύθυνση ορίζεται ως loopback IP. Όταν ο server απαντήσει στη αίτηση η VIP γίνεται διεύθυνση αποστολέα (source IP) και η διεύθυνση του client γίνεται διεύθυνση προορισμού (destination IP), πακέτο προωθείται από το switch στον router και έπειτα στον client χωρίς την ανάγκη χρήσης NAT και επιπλέον παρακάμψαμε επιτυχώς τον load balancer χωρίς να τον επιβαρύνουμε με επιπλέον εργασία.

3.3 ROUND ROBIN (RR)

Ο αλγόριθμος Round Robin είναι ο πιο απλός αλγόριθμος για καταμερισμό φορτίου σε εξυπηρετητές και οφείλεται στον πολύ απλό τρόπο λειτουργίας του, στέλνει κάθε αίτηση προς το cluster κυκλικά σε κάθε ένα server θεωρώντας τους ίσους σε υπολογιστική ισχύει αγνοώντας τον αριθμό των συνδέσεων και

τον χρόνο απόκρισης τους. Δηλαδή, σε ένα cluster με 3 server η πρώτη αίτηση θα πάει στον πρώτο server (ο administrator έχει ορίσει την αρίθμηση του κάθε server) δεύτερη στον δεύτερο server και η τρίτη αίτηση στον τρίτο server, και αν υπάρχουν περισσότερες αιτήσεις ο κύκλος ξεκινάει από την αρχή με τον ίδιο τρόπο επιλογής server. Είναι φανερό ότι ο Round Robin είναι κατάλληλος για ομογενή συστήματα όπου θα έχουμε λ1= λ2= λ3=…… =λn =λ / n (λ= αιτήσεις και n= συνολικός αριθμός server), επιπλέον αν οι αιτήσεις φτάνουν ομοιόμορφα και κάθε αίτηση επεξεργάζεται στο ίδιο χρονικό διάστημα τότε θα έχουμε τον καλύτερο καταμερισμό φόρτου εργασίας και επιπλέον τον καλύτερο καταμερισμό της υπολογιστικής ισχύς των server, δηλαδή Pi = P/ n.



Σχήμα 3.6 Round Robin

Τα θετικά του συγκεκριμένου αλγόριθμου, εκτός από την πολύ απλή εφαρμογή του είναι πολύ φτηνός, προβλέψιμος και σε περίπτωση που κάποιος server δεν είναι διαθέσιμος θα στήλη την αίτηση στο αμέσως επόμενο διαθέσιμο. Όμως σε περίπτωση που ένας server είναι πολύ αργός και δεν προλαβαίνει να διεκπεραιώσει τις αιτήσεις θα δημιουργηθεί μεγάλη ουρά (δεν έχει την δυνατότητα να ελέγχει την κίνηση/ φόρτο εργασίας σε κάθε server) οπότε δεν θα έχουμε τα επιθυμητά αποτελέσματα, οπότε θεωρείται κατάλληλος σε clusters που όλοι οι servers έχουν ίδιες δυνατότητες, πράγμα αρκετά δύσκολο.

3.3.1 WEIGHTED ROUND ROBIN (WRR)

Στην περίπτωση που έχουμε ένα cluster με servers οι οποίοι έχουν διαφορετικές δυνατότητες (μνήμη, CPU, αυτή είναι και η πιο συνηθισμένη περίπτωση), υπάρχει μια βελτιωμένη έκδοση του Round Robin, ο Weighted

Round Robin αλγόριθμος όπου κάθε server μέσα στο cluster αντιπροσωπεύεται από ένα 'βάρος' w (weight) το οποίο αντιπροσωπεύει τις δυνατότητες κάθε server για πόσες αιτήσεις μπορεί να χειρίζεται σε σχέση με τους άλλους server, έτσι αν Pi ≥ Pj τότε wi ≥ wj δηλαδή ο server με τις περισσότερες δυνατότητες θα εξυπηρετεί και τις περισσότερες αιτήσεις.

Έστω ότι έχουμε ένα cluster με 5 servers, γνωρίζοντας τις δυνατότητες τους αναθέτουμε τα ακόλουθα weights:

Αριθμός server στο cluster	**Weight, w**
Server_1	5
Server_2	3
Server_3	1
Server_4	7
Server_5	2

Πίνακας 3.1 Weights που αναθέτουμε

Σύμφωνα με τον παραπάνω πίνακα, οι πέντε πρώτες αιτήσεις θα εξυπηρετηθούν από τον Server_1, οι τρείς επόμενες από τον Server_2, η μία επόμενη από τον Server_3, οι εφτά επόμενες από τον Server_5. Τέλος οι servers με μεγαλύτερο w δέχονται πρώτη τις νέες αιτήσεις σε σχέση με τους server που έχουν μικρό w πάντα με κυκλική επανάληψη και όχι με τυχαία σειρά. Υπάρχει βέβαια και η ιδανική περίπτωση όπου όλοι οι server είναι ίση σε δυνατότητες, $P_1 = P_2 = \ldots\ldots = P_n$ όπου θα έχουμε $w_1 = w_2 = \ldots\ldots = w_n = 1/n$

3.3.2 ROUND ROBIN DNS (RR-DNS)

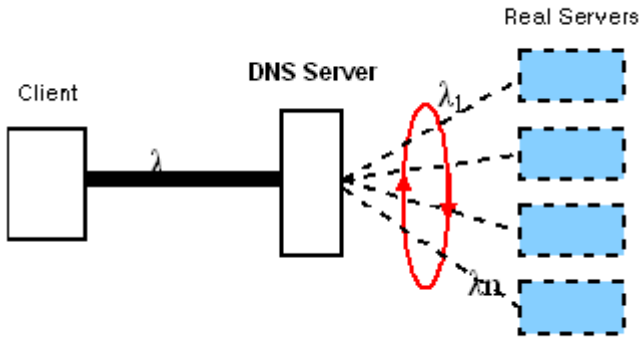
Ο RR-DNS επιτρέπει ένα domain name να συνδεθεί με πολλαπλές IP διευθύνσεις, όπου κάθε μια αντιστοιχεί και σε ξεχωριστό server. Οι αιτήσεις που φτάνουν στο συγκεκριμένο domain name δρομολογούνται στους αντίστοιχους server χρησιμοποιώντας την τεχνική Round Robin.

Πιο συγκεκριμένα, όταν η αίτηση φτάσει πρώτα στον DNS server για να αντιστοιχηθεί το domain name με την IP διεύθυνση, εμφανίζονται όλες οι IP με τις οποίες έχει γίνει αντιστοιχία και με Round Robin αλγόριθμο (κυκλικά) δρο-

μολογούνται οι αιτήσεις κάθε φορά και σε άλλο server που ανήκουν φυσικά στο ίδιο cluster. Στην συνέχεια αιτήσεις του κάθε client αποστέλλονται πάντα στον ίδιο server.

www.papadakis.com	Server1.papadakis.com	192.168.0.1
	Server2.papadakis.com	192.168.0.2
	Server3.papadakis.com	192.168.0.3
	Server4.papadakis.com	192.168.0.4

Πίνακας 3.2 Αντιστοιχία IP διευθύνσεων



Σχήμα 3.7 Round Robin DNS

Ο συγκεκριμένος αλγόριθμος δεν υπολογίζει τον παράγοντα διαθεσιμότητα (περίπτωση που κάποιος server είναι εκτός λειτουργίας) και συνεχίζει να στέλνει αιτήσεις χωρίς ανταπόκριση καθυστερώντας σημαντικά το όλο σύστημα. Δεν χρησιμοποιείται συχνά σε τοπικά clusters, όμως έχει ευρεία χρήση σε περιπτώσεις καταμερισμού φορτίου όπου οι servers βρίσκονται σε διαφορετικές γεωγραφικές περιοχές (χώρες) δίνοντας την αίσθηση στον χρήστη ότι βρίσκεται στην ίδια σελίδα.

3.3.3 Optimized Weighted Round Robin (OWRR)

Ο Optimized Weighted Round Robin αλγόριθμος είναι όμοιος με τον Weighted Round Robin, με την σημαντική διαφορά ότι ο κάθε server μπορεί να

καθορίσει δυναμικά μόνος του διαφορετικό βάρος (weight) ανάλογα με την διαθεσιμότητα του και το φόρτο εργασίας κάθε στιγμή.

3.4 PORT ADDRESS TRANSLATION (PAT)

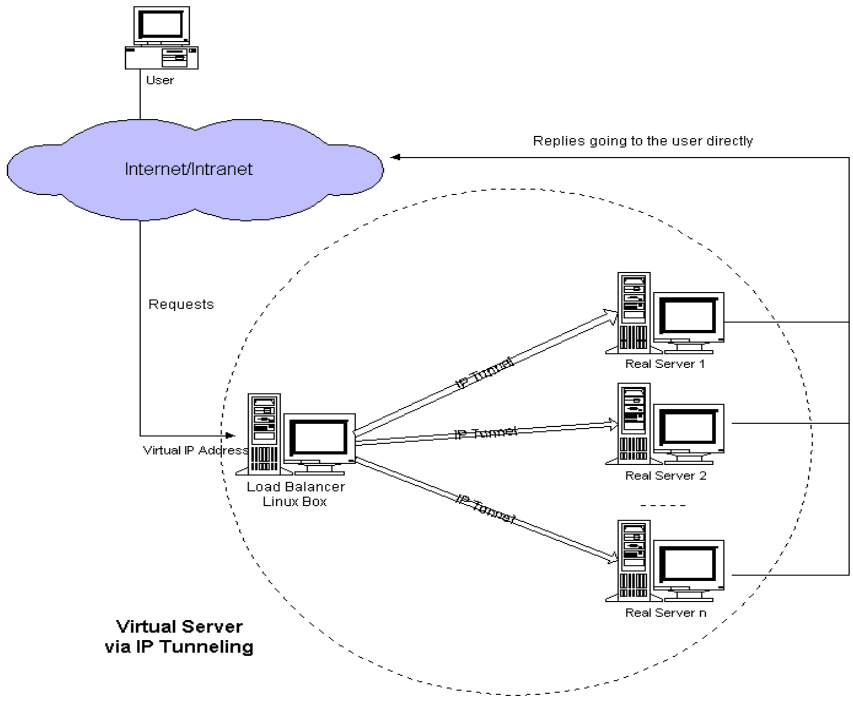
[5]Η συγκεκριμένη τεχνική, μεταφράζει τον αριθμό πόρτας στο TCP/UDP πακέτο σε ένα προκαθορισμένο αριθμό πόρτας (port). Πιο αναλυτικά, ρυθμίζουμε την πόρτα 80 (port 80) του load balancer να είναι ταυτισμένη (bind) με την πόρτα 1000 των server μέσα στο cluster, έτσι κάθε πακέτο / αίτησης που φτάνει στην πόρτα 80 του load balancer να προωθείται στην πόρτα 1000 κάποιου server και όχι στην αντίστοιχη πόρτα 80 όπως γίνεται συνήθως.

Με την χρήση PAT μπορούμε να έχουμε καλλίτερη ασφάλεια στο cluster αφού μπορούμε να αποκλείσουμε να λαμβάνονται πληροφορίες σε όποιες πόρτες εμείς θέλουμε, έτσι μπορούμε να συνδέσουμε για παράδειγμα την πόρτα 80 του load balancer με την πόρτα 4000 των real server και όχι με την αντίστοιχη 80, έτσι οι client δεν θα δούνε καμία διαφορά αφού θα συνεχίσουν να στέλνου τις αιτήσεις του στην πόρτα 80 αλλά του load balancer, οπότε αν κάποιος κακόβουλος προσπαθήσει να εισέλθει από την πόρτα αυτή, θα είναι η πόρτα του load balancer και όχι των sever, αυτό βέβαια δεν σημαίνει ότι είναι αδύνατο να εισέλθει κάποιος μετά τον load balancer, απλά θα το κάνουμε λίγο πιο δύσκολο.

Επίσης ο αλγόριθμος PAT μας επιτρέπει να εκτελούμε την ίδια υπηρεσία σε πολλαπλές πόρτες και ανάλογα με την υπηρεσία που έχουμε και με ποια πόρτα την συνδέουμε μπορούμε να έχουμε διαφορετική κλιμάκωση (scalability) στο cluster μας. Τέλος μας δίνει την δυνατότητα όταν έχουμε ένα site σε πολλούς server να αντιπροσωπεύουμε όλους τους server με την ίδια VIP διεύθυνση.

3.5 IP TUNNELING

IP tunneling είναι μια τεχνική όπου η αρχική VIP διεύθυνση περικλείεται μέσα σε ένα πακέτο και προωθείται ξανά σε μια εσωτερική IP διεύθυνση μέσα στο cluster. Έστω ότι ο χρήστης στέλνει μια αίτηση (πακέτο) στο site www.spapada.com, το πακέτο θα το παραλάβει ο load balancer (ο οποίος έχει μια VIP) και θα εξετάσει τον προορισμό του και την πόρτα προορισμού, έπειτα επιλέγεται ένας server από το cluster και ο load balancer περικλείει (encapsulates) το πακέτο μαζί με το IP datagram και το προωθεί στον επιλεγμένο server. Στο παρακάτω σχήμα[8] φαίνεται ο τρόπος επικοινωνίας μεταξύ server και client:

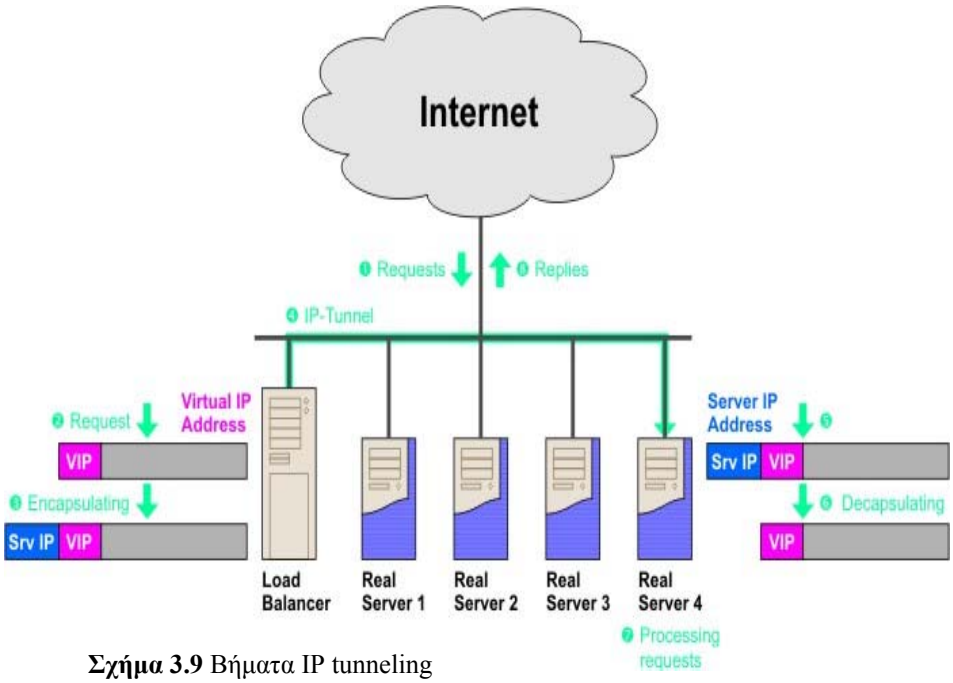


Σχήμα 3.8 IP Tunneling

Όλα τα εισερχόμενα πακέτα τα οποία ανήκουν στην συγκεκριμένη σύνδεση, πακετάρονται και στέλνονται στον ίδιο server. Όταν ο server λάβει τα πακέτα τα ανοίγει (decapsulates) και επεξεργάζεται την αίτηση στέλνοντας την απάντησης απευθείας στον χρήστη σύμφωνα με το πίνακα δρομολόγησης (routing table) και η σύνδεση τερματίζεται.

[8] από http://linuxvirtualserver.org

Απαιτείται όλοι οι server να έχουν tunneling interface ρυθμισμένο με την VIP του load balancer. Αυτό το interface δεν ανταποκρίνεται στις αιτήσεις ARP (ARP request) και η MAC διεύθυνση του load balancer θα είναι στον πίνακα ARP του router που τον συνδέει με το internet. Αυτός είναι και ο κύριος λόγος που ο load balancer encapsulates τα εισερχόμενα πακέτα και έπειτα τα προωθεί στον επιλεγμένο server μέσα στο cluster. Έτσι η απάντηση μπορεί να σταλεί πίσω στον client χωρίς αλλαγές από τον load balancer αφού ο πραγματικός server (real server) μπορεί να συμπληρώσει την VIP ως την διεύθυνση αποστολής (source IP). Στο παρακάτω σχήμα[9] παρουσιάζονται όλα τα βήματα:



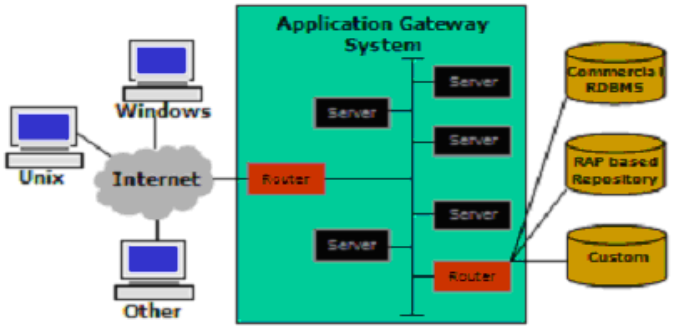
Σχήμα 3.9 Βήματα IP tunneling

Με το να καταγράφουμε τα πακέτα, και να τα στέλνουμε απευθείας πίσω στον χρήστη, μπορούμε να χειριστούμε μεγάλο αριθμό αιτήσεων και μπορεί να χρησιμοποιηθεί σε cluster με μεγάλο αριθμό server για εφαρμογές με μεγάλες απαιτήσεις. Παρόλα αυτά θα πρέπει το λειτουργικό σύστημα των server μέσα στο cluster να υποστηρίζουν IP tunneling interface.

[9] από http://www7.informatik.uni-erlangen.de/~ksjh/research/cluster/

3.6 APPLICATION GATEWAY SYSTEM (AGS)

Το παρακάτω σχήμα[10] παρουσιάζει την επικοινωνία μεταξύ client και server:



Σχήμα 3.10 Application Gateway Systems

Ο AGS αλγόριθμος για να δουλέψει πρέπει να εγκαταστήσουμε και να εκτελέσουμε ένα πρόγραμμα το rate.d σε κάθε server στο cluster. Χρησιμοποιώντας IP multicast η broadcast ο client στέλνει ένα πακέτο που περιέχει μια αίτηση για εξυπηρέτηση στο cluster, το rate.d σε όλους τους server δέχεται και αξιολογεί την αίτηση, και ο αλγόριθμος επιλέγει ένα server για να εξυπηρετήσει την αίτηση χρησιμοποιώντας μια ισότιμη τυχαία σειρά. Η επιλογή γίνεται από κάθε rate.d που είναι εγκατεστημένο χωριστά αλλά συνήθως επιλέγουν τον ίδιο server ο οποίος στέλνει μια unicast απάντηση σον client αναφέροντας ότι θα δεκτή την αίτηση. Μόλις ο client λάβει την απάντηση ξεκινάει την σύνδεση με το αντίστοιχο server μέσα στο cluster με κάποια από τις γνωστές μεθόδους όπως FTP, HTTP.

Πως όμως το rate.d επιλέγει το κατάλληλο server για να απαντήσει την αίτηση? Αναθέτει την εξυπηρέτηση στο συγκεκριμένο server βασισμένο σε κάποια πακέτα που μεταδίδει και αναλύει ο κάθε server πολλές φορές κάθε δευτερόλεπτο υπολογίζοντας από αυτά το φόρτο εργασίας του κάθε server. Η κάθε rate.d εφαρμογή παρακολουθεί αυτές τις μεταδόσεις πακέτων όχι μόνο στον δικό της server αλλά και σε όλους τους server που το "τρέχουν" εκείνοι την στιγμή μέσα στο cluster. Τα πακέτα που στέλνονται και αναλύονται είναι αντίστροφος ανάλογος του φόρτου εργασίας εκείνη την στιγμή και συνεχίζονται

[10] από http://www.cnri.reston.va.us/index.html

να στέλνονται μέχρις ότου η rate.d εφαρμογή να έχει καταγεγραμμένη τι ίδιο φόρτο εργασίας σε κάθε server οπότε και θα επιλέγουν με μεγάλη ακρίβεια σε ποιον server θα στείλουν και αν θα στείλουν κάποια αίτηση για εξυπηρέτηση. Η επιλογή εξαρτάται μόνο από της πληροφορίες από τα πακέτα που μεταδίδονται και αναλύονται, τα χαμένα πακέτα δεν υπολογίζονται, κάνοντας έτσι τον rate.d αλγόριθμο πολύ σταθερό και ακριβή. Επιπλέον σε περίπτωση που περισσότεροι από ένας server απαντήσουν στον client ο αλγόριθμος επιλέγει να εξυπηρετήσει ο πρώτος server που απάντησε στην αίτηση.

Υπάρχει όμως και η περίπτωση κανένας server να μην δεκτή την αίτηση, τότε ο αλγόριθμος μεταδίδει μαζί με το πακέτο και μια λίστα με IDs των server οι οποίοι είχαν απαντήσει πρόσφατα σε κάποια αίτηση την οποία λαμβάνουν όλοι οι rate.d που "τρέχουν" στους server μέσα στο cluster. Όταν το λάβει και αυτός ο server ο οποίος έπρεπε να έχει δεκτή την αίτηση και εξακολουθεί να την απορρίπτει γίνεται ξανά αποστολή παραβλέποντας τον συγκεκριμένο server από την λίστα, η διαδικασία επαναλαμβάνεται μέχρι κάποιος να δεκτή την αίτηση.

3.7 CONNECTION ALGORITHMS

3.7.1 LEAST CONNECTIONS (LC)

Ο load balancer που χρησιμοποιεί τον συγκεκριμένο αλγόριθμο, προωθεί τις αιτήσεις στον server με τον μικρότερο αριθμό συνδέσεων και ανήκει στην κατηγορία των δυναμικών αλγόριθμων αφού καταγράφει συνεχώς τις αυξομειώσεις στις συνδέσεις στον κάθε server χωριστά. Καθώς ο server με τις λιγότερες συνδέσεις θα λάβει και τις περισσότερες αιτήσεις με την πάροδο του χρόνου θα φτάσει να έχει και αυτός τις ίδιες συνδέσεις με τους υπόλοιπους server έχοντας έτσι όλοι την ίδια ουρά αιτήσεων προς εξυπηρέτηση, έτσι θα ισχύει [10]:

$$\frac{\lambda_i}{\lambda_j} = \frac{P_i}{P_j} = \frac{P_i / P}{P_j / P} \qquad \text{for} \qquad \forall i, j \qquad (1)$$

Δηλαδή ο αλγόριθμος θα προωθεί με τέτοιο τρόπο ώστε ο λόγος του ποσοστού αφίξεων να είναι ίσος με το ποσοστό απόδοσης των server, έτσι αν οι δυνατότητες κάποιου server είναι πολύ μεγαλύτερες από κάποιου άλλου θα λάβει και την ανάλογη αύξηση στις αιτήσεις που θα πρέπει να εξυπηρετήσει. Επιπλέον συνεπάγεται ότι ο λόγος ανάμεσα το ποσοστό των αιτήσεων και το ποσοστό απόδοσης είναι σταθερό για όλους τους server ,δηλαδή ισχύει:

$$\frac{\lambda_1}{P_1} = \frac{\lambda_2}{P_2} = \cdots = \frac{\lambda_n}{P_n} = \frac{\lambda}{P}. \qquad r_{LC} = \frac{ns_1P_1}{P - \lambda s_1 P_1}.$$

Από όπου μπορούμε να πάρουμε την σχέση (2) η οποία μας δίνει τον μέσο χρόνο απόκρισης (mean response time) του cluster και στην οποία παρατηρούμε [10] ότι το r_{LC} είναι ανεξάρτητο από ποσοστό κατανομής της απόδοσης το οποίο είναι πολύ σημαντικό σε cluster όπου οι server έχουν μεγάλες διαφορές στην απόδοση τους αφού θα έχουμε πιο σταθερούς χρόνους απόκρισης.

Σε ένα ομογενές [10] cluster όπου όλοι οι server έχουν την ίδια απόδοση ο least connection αλγόριθμος θα στείλει τις ίδιες αιτήσεις στους server οπότε θα ήταν πιο οικονομικό να χρησιμοποιήσουμε τον Round-Robin αφού θα έχουμε τα ίδια αποτελέσματα. Αντίθετα σε ένα ανομοιογενές cluster ο συγκεκριμένος αλγόριθμος παρέχει ένα απλό μηχανισμό που στέλνει τις περισσότερες αιτήσεις στο server με την μεγαλύτερη υπολογιστική ισχύει. Παρόλο που δεν μειώνει σημαντικά τον μέσο χρόνο απόκρισης του cluster ανταποκρίνεται πολύ καλά όταν οι server δεν έχουν μεγάλες αποκλίσεις στις δυνατότητες τους, για να μειώσουμε τον μέσο χρόνο απόκρισης του cluster θα πρέπει να ικανοποιείται η παρακάτω εξίσωση [10] μεταξύ των του server i και του server j :

(3)
$$\frac{\lambda_i}{\lambda_j} = \frac{\frac{P_i}{P} - (prob.\_farm\_idle)\left(\frac{\sqrt{P_i}}{\sum_{k=1}^{n}\sqrt{P_k}}\right)}{\frac{P_j}{P} - (prob.\_farm\_idle)\left(\frac{\sqrt{P_j}}{\sum_{k=1}^{n}\sqrt{P_k}}\right)},$$

Αν ικανοποιούνται η εξισώσεις 2 και 3 τότε θα έχουμε τον βέλτιστο χρόνο απόκρισης για τον load balancer που χρησιμοποιεί τον Least Connection αλγόριθμο και θα είναι ίσος με [10] :

$$r_{\min} = \frac{1}{\lambda}\left[\frac{\left(\sum_{k=1}^{n}\sqrt{P_k}\right)^2}{P-\lambda s_1 P_1} - n\right] \le \frac{1}{\lambda}\left[\frac{n\sum_{k=1}^{n}P_k}{P-\lambda s_1 P_1} - n\right]$$

$$= \frac{n s_1 P_1}{P-\lambda s_1 P_1} = r_{LC}$$

Ας δούμε ένα παράδειγμα [10], υποθέστε ότι έχουμε ένα cluster με τρεις servers των οποίο οι απόδοση τους φαίνεται στον πίνακα 3.3 . Σταθεροποιούμε την απόδοση του server 1 στο 10 και υποθέτουμε ότι κάθε αίτηση χρειάζεται μια μονάδα (π.χ ένα second) για να επεξεργαστή από τον server 1. Το ποσοστό άφιξης αιτήσεων στο cluster είναι ένα ανά μονάδα χρόνου (second). Έπειτα μεταβάλουμε την απόδοση των server 2 και 3 έτσι ώστε η συνολική απόδοση P (server 1+server 2+server 3) να είναι 30.

Server 1 Performance Rating	Server 2 Performance Rating	Server 3 Performance Rating	Συνολική Performance Rating (P)	$\mathbf{r}_{LC}$	$\mathbf{r}_{min}$
10	10	10	30	1.50	1.50
10	9	11	30	1.50	1.49
10	8	12	30	1.50	1.47
10	7	13	30	1.50	1.43
10	6	14	30	1.50	1.37
10	5	15	30	1.50	1.30

Πίνακας 3.3 Παράδειγμα Least Connection

Όπως ήταν αναμενόμενο ο μέσος χρόνος απόκρισης $\mathbf{r}_{LC}$ δεν αλλάζει ακόμα και στις περιπτώσεις όπου οι server δεν έχουν την ίδια απόδοση.

3.7.2 WEIGHTED LEAST CONNECTIONS (WLC)

Όπως αναφέραμε, ο least connection αλγόριθμος δεν είναι αποτελεσματικός σε ένα cluster όπου οι server έχουν διαφορετικές υπολογιστικές δυνατότητες, έτσι υπάρχει η βελτιωμένη έκδοση του αλγόριθμου ο weighted least connection αλγόριθμος. Όπως στον weighted Round Robin και εδώ κάθε server μέσα στο cluster αντιπροσωπεύεται από ένα "βάρος" με την διαφορά ότι ο WRR αλγόριθμος καθορίζει το "βάρος" βασισμένος ποσοστό αιτήσεων που φτάνουν στον κάθε server, ενώ ο WLC βασίζεται στον αριθμό των ενεργών συνδέσεων του κάθε server.

Ο αριθμός των συνδέσεων και το ποσοστό των αιτήσεων εδώ σχετίζονται μεταξύ τους, το βέλτιστο "βάρος" (W) για το βέλτιστο ποσοστό συνδέσεων δίνεται από την σχέση [10]:

$$w_i(WLC) = \left.\frac{\lambda_i S_i}{1-\lambda_i S_i}\right|_{\lambda_i = optimal\_request\_rate\_to\_server\_i}$$

όπου Si είναι ο χρόνος εξυπηρέτησης του server i, επίσης αυτά τα "βάρη" μπορούν να παρουσιαστούν και με την παρακάτω σχέση [10] σαν ποσοστό σε σχέση με τις συνολικές συνδέσεις στο cluster

$$w_i(WLC) \leftarrow \frac{w_i(WLC)}{\sum_{k=1}^{n} w_k(WLC)}$$

Έστω ότι έχουμε τρεις server των οποίων η απόδοση φαίνεται στον παρακάτω πίνακα [10], σύμφωνα με τα παραπάνω υπολογίζουμε τα "βάρη" σε ποσοστό επί τις εκατό και τα παραθέτουμε δίπλα από κάθε server.

Server 1 Performance Rating	W1 %	Server 2 Performance Rating	W2 %	Server 3 Performance Rating	W3 %	r_{min}
10	33	10	33	10	33	1.50
10	33	9	28	11	39	1.49
10	34	8	23	12	43	1.47
10	34	7	17	13	49	1.43
10	35	6	11	14	54	1.37
10	36	5	2	15	62	1.30

Πίνακας 3.4 Παράδειγμα Weighted Least Connection

Στον server 1 όπου έχουμε σταθερή απόδοση έχουμε μικρές αλλαγές στο w1, αντίθετα στον server 2 παρατηρούμε ότι καθώς η απόδοση του server μειώνεται σημαντικά και το ποσοστό στα "βάρη" που του δίνει ο load balancer μειώνονται και αυτά, επόμενος και οι αιτήσεις που θα λάβει θα είναι λιγότερες. Στον server 3 συμβαίνει το ακριβώς το αντίθετο.

3.7.3 LOCALITY-BASED LEAST-CONNECTION SCHEDULING

Ο αλγόριθμος χρησιμοποιείται συνήθως σε cache clusters και δρομολογεί αιτήσεις στους servers με τις λιγότερες ενεργές συνδέσεις που σχετίζονται με την IP διεύθυνση προορισμού. Σε περίπτωση που κάποιος server υπερφορτωθεί, υπάρχουν περισσότερες ενεργές συνδέσεις, βρίσκει τον server με λιγότερες συνδέσεις και δρομολογεί εκεί της επόμενες αιτήσεις.

3.7.4 LOCALITY-BASED LEAST-CONNECTION WITH REPLICATION SCHEDULING

Η διαφορά με τον locality based least connection είναι στο ότι ο load balancer καταγράφει συνεχώς ποίοι server μπορούν να εξυπηρετήσουν κάποιον client και σε περίπτωση που ο server που εξυπηρετεί υπερβεί τις δυνατότητες του επιλέγεται ένας νέος ο οποίος έχει τις λιγότερες ζωντανές συνδέσεις εκείνη την στιγμή. Αν η λίστα με τους server που μπορούν να εξυπηρετήσουν δεν ανανεωθεί για αρκετό διάστημα, ο server με το μεγαλύτερο φόρτο εργασίας αφαιρείται από την λίστα προκειμένου να αποφύγουμε μεγάλο βαθμό επανάληψης (high degree of replication).

3.7.5 MAXIMUM CONNECTIONS (MC)

Εδώ ο load balancer που χρησιμοποιεί τον συγκεκριμένο αλγόριθμο βάζει όριο στον κάθε server μέσα στο cluster, ένα μέγιστο αριθμό των συνδέσεων που μπορεί να έχει, έτσι και ο αριθμός των αιτήσεων που μπορεί να εξυπηρετήσει δεν μπορεί να υπερβεί κάποιο όριο αποτρέποντας την περίπτωση ο server να προσπαθεί να ανταποκριθεί σε φόρτο εργασίας μεγαλύτερο από αυτό που μπορεί πραγματικά να αντιμετωπίσει ώστε να έχει την βέλτιστη απόδοση.

[10] Έστω ‘ότι Li είναι ο ανώτερος αριθμός συνδέσεων του server i. Μπορούμε μα μοντελοποιήσουμε τον load balancer ως ένα σύστημα ουράς με ελάχιστο χώρο αναμονής και να υπολογίσουμε τον ελάχιστο μέσο χρόνο απόκρισης για να εξυπηρετήσει την κάθε αίτηση στον server i ri(Li). Ο παρακάτω πίνακας δείχνει την διαφορά χωρίς και με όριο L=5 όπου φαίνεται σημαντικά η διαφορά.

Utilization (%)	Response time without a connection limit	Response time with a connection limit L=5
10	1.11	1.11
20	1.25	1.24
30	1.42	1.41
40	1.66	1.61
50	2.00	1.83
60	2.50	2.07
70	3.33	2.32
80	5.00	2.56
90	10.00	2.79
99	10,000	2.99

Πίνακας 3.5 Παράδειγμα Maximum Connections

3.8 DESTINATION HASHING SCHEDULING

Ο destination hashing scheduling αλγόριθμος δρομολογεί τις αιτήσεις μέσα στο cluster συμβουλευόμενος ένα στατικό hash πίνακα (hash table) σύμφωνα με την διεύθυνση προορισμού (destination IP). Είναι σχεδιασμένος για proxy-cache clusters.

3.9 SOURCE HASHING SCHEDULING

Ομοίως ο source hashing scheduling αλγόριθμος δρομολογεί τις αιτήσεις μέσα στο cluster συμβουλευόμενος ένα στατικό hash πίνακα (hash table) σύμφωνα με την διεύθυνση αποστολέα (source IP). Είναι σχεδιασμένος για proxy-cache clusters.

3.10 SHORTEST EXPECTED DELAY SCHEDULING

Ο shortest expected delay scheduling αλγόριθμος δρομολογεί τις αιτήσεις στον server ο οποίος θα επεξεργαστεί τις αιτήσεις με την λιγότερη αναμενόμενη καθυστέρηση η οποία υπολογίζεται από την σχέση (Ci + 1) / Ui, όπου i ο αριθμός του server, Ci ο αριθμός των συνδέσεων του i server και Ui είναι ο σταθερός ρυθμός εξυπηρέτησης (weight) του i server.

3.111 NEVER QUEUE SCHEDULING

Εδώ ο αλγόριθμος δεν έχει σαν κριτήριο επιλογής την ταχύτητα των server υιοθετώντας έτσι ένα διπλό μοντέλο, ο load balancer θα αρχίσει να στέλνει τις αιτήσεις στον server ο οποίος είναι αδρανής αντί στον πιο γρήγορο του cluster ή αν δεν υπάρχει κάποιος server σε αδράνεια θα στέλνει τις αιτήσεις στον server ο οποίος έχει τον μικρότερο χρόνο απόκρισης μέσα στο cluster.

3.12 SERVER RESPONSE TIME (SRT)

Εδώ ο αλγόριθμος προωθεί τις αιτήσεις στον server ο οποίος έχει τον πιο γρήγορο χρόνο απόκρισης (response time), τεχνική η οποία φαίνεται να είναι ιδανική αφού θεωρητικά το να εξυπηρετεί ο ποιος γρήγορος θα έχουμε και πιο αποδοτική εξυπηρέτηση. Τι θα γίνει όμως αν ο load balancer συνεχίζει να στέλνει αιτήσεις στον ίδιο server αφού αυτόν θεωρεί τον πιο γρήγορο στο να απαντά ; το αποτέλεσμα θα αναστραφεί αφού πλέον ο συγκεκριμένος server δεν θα είναι σε θέση να εξυπηρετεί το ίδιο γρήγορο λόγω του μεγάλου αριθμού αιτήσεων.

[10] Υποθέστε ότι ο ποίο γρήγορος server είναι u*100 % απασχολημένος, όταν η βασική χρήση (base utilization) αυξηθεί 10% από u σε 1.1u τότε ο χρόνος απόκρισης θα αυξηθεί περίπου 10u / (1-1.1u)%, για παράδειγμα αν η βασική χρήσης είναι στο 20% μια αύξηση 10% στην χρήση του server θα σήμαινε μια αύξηση 2.6% στον χρόνο απόκρισης του server. Στον παρακάτω πίνακα φαίνονται αρκετά ακόμα παραδείγματα και τα οποία αποδεικνύουν ότι η σχέση μεταξύ του χρόνου απόκρισης του server και της βασικής χρήσης δεν είναι γραμμική.

Base Server Utilization (%)	Response Time Increase
10	1.1
20	2.6
30	4.5
40	7.1
50	11.1
60	17.6
70	30.4
80	66.7
90	900

Πίνακας 3.6 Server Response Time SRT

Προκειμένου να μην φτάσουμε ποτέ σε αυτήν την περίπτωση, ο αλγόριθμος σκέφτεται έξυπνα διενεργώντας κάθε φορά [10] “what if” ανάλυση στον server που πρόκειται να στείλει την αίτηση, ξέροντας τις δυνατότητες του cluster έστω ότι αρχικά θα έχουμε P1≥ P2≥…… ≥ Pn έτσι ο load balancer θα στέλνει όλες τις αιτήσεις στον server_1 μέχρι την στιγμή που ο χρόνος απόκρι-

σης του server_1 γίνει μικρότερος από αυτόν του server_2, έπειτα θα συνεχίσει να στέλνει τις αιτήσεις στον server_2 μέχρι την στιγμή που ο χρόνος απόκρισης του server_2 γίνει μικρότερος από αυτόν του server_3 μέχρι τον τελευταίο server, επιτυγχάνοντας έτσι κανένας server να μην περάσει τις δυνατότητες του και να μην φτάσει ποτέ σε υπερφόρτωση, επιπλέον κάποια στιγμή σε ιδανικές συνθήκες θα καταφέρουμε όλοι οι server να έχουν τον ίδιο χρόνο απόκρισης δηλαδή :

$r_i = r_j$ και $\lambda i - \lambda j = \frac{1}{s_i} - \frac{1}{s_j}$ για $\forall$ i,,j

3.13 LOWEST CPU UTILIZATION ALGORITHM

Εδώ ο αλγόριθμος προωθεί τις αιτήσεις στον server ο οποίος εμφανίζει την χαμηλότερη χρήση της Κεντρικής Μονάδας Επεξεργασίας (CPU). Ο συγκεκριμένος αλγόριθμος είναι κατάλληλος για αιτήσεις με δυναμικό περιεχόμενο.

3.14 SOURCE IP ADDRESS ALGORITHM

Ο αλγόριθμος προωθεί τις αιτήσεις σύμφωνα με την IP διεύθυνση προορισμού του client, είναι πολύ απλός αλγόριθμος και δουλεύει πολύ καλά σε clusters τα οποία ανήκουν σε μικρά δίκτυα, όπως κάποιο τμήμα κάποιας εταιρείας.

3.15 RESPONSE TIME ALGORITHM

Ο αλγόριθμος χρησιμοποιεί μια λίστα με τους χρόνους απόκρισης κάθε server μέσα στο cluster για να καθορίσει ποιος έχει το λιγότερο φόρτο εργασίας και να επιλεγεί πρώτος στο να εξυπηρετήσει αιτήσεις. Η λίστα με τους χρόνους

ανανεώνεται συνέχεια (δίνεται η δυνατότητα να επιλεγεί από τον διαχειριστή του cluster το πόσο συχνά) , έτσι η σειρά των server αλλάζει συνεχώς.

Εδώ χρησιμοποιούνται δυο παράμετροι:

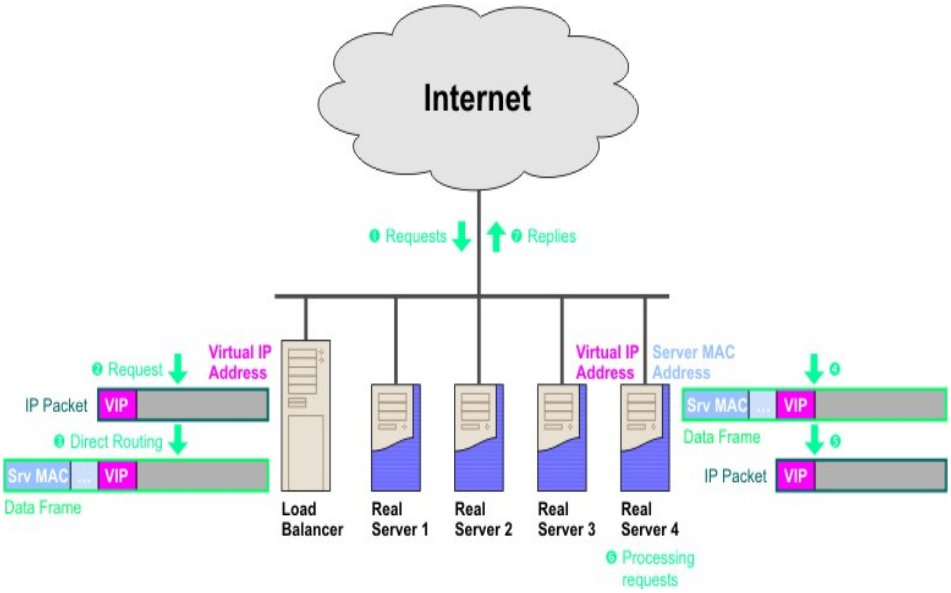
- Response refresh rate όπου ο διαχειριστής καθορίζει την διάρκεια (σε milli-seconds) κατά την οποία θα υπολογίζεται ο χρόνος απόκρισης κάθε server και στο τέλος της οποίας συμπληρώνεται η λίστα με τους χρόνους όλων των server.
- Max Clients που καθορίζει τον μέγιστο αριθμό client που μπορεί να εξυπηρετεί κάθε server, μόλις κάποιος server φτάσει αυτό το όριο ο αλγόριθμος δεν δρομολογεί άλλες αιτήσεις μέχρις ότου κάποιος client αποσυνδεθεί και μειωθεί ο αριθμός των ενεργών συνδέσεων.

3.16 DIRECT ROUTING (OR DIRECT PATH ROUTING)

Εδώ η VIP διεύθυνση ανατίθεται σε ένα interface στον load balancer και στις alias συσκευές του real server. Καθώς η MAC διεύθυνση του interface του load balancer είναι αυτή που βρίσκεται στο ARP πίνακα για την VIP στον router τα πακέτα από τις αιτήσεις φτάνουν στον load balancer πρώτα. Ο load balancer δεν χρειάζεται να ξαναγράψει η να encapsulate το πακέτο, μεταδίδει την απάντηση στον real server καθορισμένο από ένα scheduling αλγόριθμο βάζοντας την MAC διεύθυνση του συγκεκριμένου real server ως διεύθυνση αποστολής. Μόλις το πακέτο φτάσει στον real server, το δέχεται αφού το alias μεταφέρει την VIP διεύθυνση.

Αυτή η τεχνική καταμερισμού φορτίου έχει πολύ καλή απόδοση αφού δεν υπάρχει επανεγγραφή των πακέτων που φτάνουν παρά μόνο μετάφραση δυναμικά της IP διεύθυνσης στην MAC διεύθυνση και παρουσιάζεται στο παρακάτω σχήμα[11]

[11] από http://www7.informatik.uni-erlangen.de/~ksjh/research/cluster/



Σχήμα 3.11 Direct Routing

3.17 LAST VISITED ROUTING

Εδώ ο αλγόριθμος στέλνει της αιτήσεις κάθε client πάντα στον ίδιο server μέσα στο cluster ο οποίος εξυπηρέτησε την πρώτη αίτηση του συγκεκριμένου client. Ο τρόπος επιλογής του server την πρώτη φορά μπορεί να γίνει με οποιοδήποτε άλλο αλγόριθμο από αυτούς που περιγράφουμε και στην συνέχεια δεν χρησιμοποιείται ξανά.

3.18 LEAST LOADED ROUTING

Κάνει καταμερισμό φορτίου της CPU ομοιόμορφα ανάμεσα σε όλους τους κόμβους μέσα σε ένα cluster. Ο παράγοντας που παίρνει υπόψη ο server (συνήθως χρησιμοποιείται proxy server) για να αποφασίσει που θα προωθήσει την νέα αίτηση είναι ο φόρτος εργασίας που έχει ο server εκείνη την στιγμή και μετριέται με το πόσο γρήγορα μπορεί να εξυπηρετεί τις αιτήσεις (processing time) και όχι με το πόσες αιτήσεις μπορεί εξυπηρετεί ταυτόχρονα.

Ο συγκεκριμένος αλγόριθμος διαφέρει από τους υπόλοιπους γιατί δεν μπορεί να δουλέψει μόνος του, χρειάζεται ένα μηχανισμό επικοινωνίας μεταξύ

του proxy server και των server μέσα στο cluster. Για να έχει ο proxy server το ακριβές φόρτο εργασίας κάθε στιγμή χρησιμοποιεί δυο μεθόδους (ο διαχειριστής επιλέγει ανάλογα με τις ανάγκες του δικτύου), περιοδικά καταγράφει το φόρτο κάθε server η όταν κάποιος server στήλη μια αίτηση για εξυπηρέτηση τότε μόνο ο proxy server καταγράφει το φόρτο όλων των server και επιλέγει που να στήλη την αίτηση.

3.19 PORT-BOUND SERVERS

Όταν ρυθμίζεις κάποιον virtual server πρέπει να προσδιορίσεις τις TCP και UDP πόρτες οι οποίες θα καθοδηγούνται από τον συγκεκριμένο server. Ωστόσο, αν ρυθμίσω NAT στο cluster θα μπορώ να τους ρυθμίσω να λειτουργούν ως port-bound servers, δηλαδή να φτιάξω μια ομάδα servers μέσα στο cluster (έτσι θα είναι σαν να έχω cluster μέσα στο cluster) τους οποίος θα αντιπροσωπεύει μια VIP και θα είναι όλοι υπεύθυνοι για την ίδια εργασία (service) όπως Hypertext Transfer Protocol (HTTP).

3.20 CLIENT-ASSIGNED LOAD BALANCING

Client-assigned load balancing σου επιτρέπει να περιορίσεις την πρόσβαση σε κάποιον virtual server η cluster καθορίζοντας την λίστες με IP που τους επιτρέπεται η πρόσβαση και λίστες με IP που δεν τους επιτρέπεται. Με αυτό τον τρόπο μπορούμε να καθορίσουμε μια ομάδα IP όπως εσωτερικές διευθύνσεις από κάποιο υπό-δίκτυο (subnet) το οποίο να μπορεί να έχει πρόσβαση στο cluster και κάποια άλλη διαφορετική ομάδα η οποία να μην έχει.

3.21 STICKY CONNECTIONS

Εδώ ο αλγόριθμος καταγράφει ποιος server εξυπηρέτησε τον κάθε client (ουσιαστικά καταγράφει μια λίστα από τις IP διευθύνσεις των client) και σε περίπτωση που έρθει αίτηση για εξυπηρέτηση από τον ίδιο client τον δρομολογεί

στον ίδιο server μέσα στο cluster. Πως όμως επιτυγχάνεται αυτό; Ο load balancer δημιουργεί sticky αντικείμενα (sticky objects) για να παρακολουθεί τις αιτήσεις από τους client. Αυτά τα αντικείμενα παραμένουν στην βάση δεδομένων του load balancer μετά την τελευταία σύνδεση κάποιου client για χρονικό διάστημα το οποίο καθορίζει ο διαχειριστής και ονομάζεται sticky timer.

Σε περίπτωση που ο αλγόριθμος εκτελείται σε virtual server οι νέες συνδέσεις από κάποιον client στέλνονται στον server που εξυπηρέτησε τον προηγούμενο client αρκεί να υπάρχει μια διαθέσιμη σύνδεση, ο χρόνος ανάμεσα από το τέλος της προηγούμενης σύνδεσης και της αρχής της νέα σύνδεσης να είναι μέσα στο stick timer. Επιπλέον ο sticky connection αλγόριθμος επιτρέπει την ομαδοποίηση (coupled) υπηρεσιών που εκτελούνται σε διαφορετικούς virtual server. Αυτό μας επιτρέπει συνδέσεις που προορίζονται για την ίδια υπηρεσία να εκτελούνται από τον ίδιο server. Για παράδειγμα το HTTP χρησιμοποιεί την πόρτα 80 και το HTTPS την πόρτα 443, αν και οι δυο υπηρεσίες ομαδοποιηθούν μπορούν οι αιτήσεις από τον ίδιο client (ίδια IP) η και κάποιο υπό δίκτυο να εξυπηρετούνται από τον ίδια server.

3.22 DELAYED REMOVAL OF TCP CONNECTION CONTEXT

Because of IP packet ordering anomalies, IOS SLB might “see” the termination of a TCP connection (a finish [FIN] or reset [RST]) followed by other packets for the connection. This problem usually occurs when there are multiple paths that the TCP connection packets can follow. To correctly redirect the packets that arrive after the connection is terminated, IOS SLB retains the TCP connection information, or context, for a specified length of time. The length of time the context is retained after the connections terminated is controlled by a configurable delay timer.

3.23 ONE-IP

[11] Ο One-IP αλγόριθμος είναι από τις πρώτες υλοποιήσεις σε στρώμα 2 (L2) τεχνολογία clustering και αναπτύχθηκε στα εργαστήρια της Bell και υποστηρίζει δυο διαφορετικούς μηχανισμούς. Ο πρώτος, όταν ο load balancer λάβει μια αίτηση συμβουλεύεται ένα πίνακα με IP διευθύνσεις ο οποίος του δείχνει ποιον server να επιλέξει. Ο δεύτερος, ο load balancer μεταδίδει πακέτα (περιέχουν και την IP διεύθυνση του client που στέλνει την αίτηση) σε όλο το cluster και κάθε server μέσο κάποιου φίλτρου επιλέγει πιο client (IP) θέλει να εξυπηρετήσει.

3.24 RANDOM

Ο αλγόριθμος random είναι ιδιαίτερα απλός, καθώς για κάθε αίτηση που πρέπει να εξυπηρετηθεί από τον server του cluster επιλέγεται τυχαία ένας server. Όλοι οι servers έχουν την ίδια πιθανότητα να επιλεγούν, έτσι για ένα cluster με 10 real server, κάθε ένας επιλέγεται με πιθανότητα 1/10.

3.25 EQUILOAD

[26]Είναι μια νέα τεχνική που παρουσιάζει το τμήμα πληροφορικής του College of William and Mary και στην οποία το μέγεθος κάθε εισερχόμενης αίτηση είναι αυτό που καθορίζει ποιος server θα την εξυπηρετήσει.

Αν υποθέσουμε ότι έχουμε N servers μέσα στο cluster, ο EquiLoad χρειάζεται να υποδιαιρέσει όλες τις πιθανές αιτήσεις σε N διαστήματα (intervals) , [s0 Ψ 0; S1), [s1; S2), . . . , [sN-1; SN ≡ ∞), έτσι ο server i θα είναι υπεύθυνος να εξυπηρετεί αιτήσεις μα μέγεθος μεταξύ Si-1 και Si. Όταν ο load balancer λάβει μια αίτηση ελέγχει το μέγεθος S και αν Si-1 ≤ S ≥ Si βάζει την αίτηση στην ουρά, αν όχι στέλνει την αίτηση σε κάποιον διαθέσιμο server.

ΠΙΝΑΚΑΣ 3.7 ΜΕ ΠΡΟΙΟΝΤΑ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ LOAD BALANCING

Cisco Systems	www.cisco.com	Local Director
Alteon Web Systems	www.alteonwebsystems.com	
ArrowPoint Communica-tions	www.arrowpoint.com	
Cabletron Systems	www.ctron.com	
F5 networks	www.f5.com	
Intel	www.intel.com/network	
Zeus Technology	www.zeus.co.uk	1. e-network Dispacher 2. Web accelerator
IBM		
2X	www.2x.com	
Nortel	www.nortel.com	
Vatronix Secure systems	www.vantronix.com	
Sysmaster	www.sysmaster.com	
Astaro internet security	www.astaro.com	
Astro	www.astrocorp.com	

ΕΠΙΛΟΓΟΣ

Η χρήση Load Balancer για την δημιουργία clusters και τον καταμερισμό του φόρτου εργασίας ανάμεσα στους servers, αποτελεί πλέον την πιο διαδεδομένη μέθοδο παγκοσμίως και για αυτό χρησιμοποιείται ευρέος από μικρές έως πολύ μεγάλες επιχειρήσεις, οργανισμούς και εκπαιδευτικά ιδρύματα. Προσφέρουν μεγάλα πλεονεκτήματα όπως Κλιμάκωση (scalability) του δικτύου όπου σε περιπτώσεις αύξησης της υπολογιστική ισχύ του δικτύου μας δεν χρειάζεται να έχουμε μεγάλης ισχύος servers, αλλά μικρότερης ισχύος και κάθε φορά που θέλουμε περισσότερο απλά προσθέτουμε ακόμα server η ακόμα και κάποιο άλλο cluster μέσα στο είδη υπάρχον (scaling out), λύση πολύ πιο οικονομική. Επιπλέον μπορούμε να αυξήσουμε την ισχύει του κατά μέρους server απλά προσθέτοντας μνήμη, μεγαλύτερο δίσκο η κάποιον πιο γρήγορο επεξεργαστή (scale up). Διαθεσιμότητα όπου σε περίπτωση που κάποιος server δεν ανταποκριθεί θα βγει εκτός cluster αναλαμβάνοντας οι υπόλοιποι να απαντήσουν στις αιτήσεις μέχρι να λυθεί το πρόβλημα και επανέλθει στο cluster. Ευχρηστία, εδώ ο διαχειριστής μπορεί να αφαιρει η να προσθέτει servers για αναβάθμιση, τεχνικό έλεγχο κ.τ.λ. όποια στιγμή θεωρεί ότι υπάρχει ανάγκη χωρία να βγάζει εκτός λειτουργίας το site, και τέλος ασφάλεια, αφού η μόνη IP που ουσιαστικά βλέπει ο χρήστης είναι αυτή του Load Balancer και όχι των server του cluster, μειώνοντας τις περιπτώσεις εισβολής στο σύστημα μας.

Καταγράψαμε και αναλύσαμε πολλούς αλγόριθμους καταμερισμού φορτίου και κατά την διάρκεια της έρευνας βρήκαμε ακόμα περισσότερους, κάθε ένας με ιδιαίτερα χαρακτηριστικά και επιδόσεις, θετικά και αρνητικά, τα οποία όμως καταλήγουμε στο συμπέρασμα ότι ανάλογα το μέγεθος του δικτύου που εφαρμόζονται και τον τύπο των υπηρεσιών που εξυπηρετούν, αυτές οι ιδιότητες μπορούν να αντιστραφούν. Έτσι θα πρέπει πριν ξεκινήσει η υλοποίηση του δικτύου να αναρωτηθούμε, τι απόδοση θέλουμε να έχει το cluster μας για την συγκεκριμένη υπηρεσία με τον διαθέσιμο εξοπλισμό; Τις πιο πολλές φορές καταλήγουμε (ιδιαίτερα σε πολύ μεγάλα δίκτυα, και δίκτυα ηλεκτρονικού εμπορίου

όπου το να διακοπή η λειτουργία του δικτύου είναι όπως λένε και οι ξένοι ‘out of the question’) στην χρήση περισσότερων από ένα αλγόριθμο, ο κάθε ένα να μας εξυπηρετεί για διαφορετικό σκοπό.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Γ. Παπακωνσταντίνου, Θ. Θεοχάρης, Π. Τσανάκας
"Συστήματα Παράλληλης Επεξεργασίας", Συμμετρία, 1994.

[2] James F. Kurose, Keith W. Rosss
"Δικτύωση Υπολογιστών, Προσέγγιση από Πάνω προς τα Κάτω με Έμφαση στο Διαδίκτυο", Μ. Γκούρδας, 2003.
Σελίδες:1-64

[3] Andrew S. Tanenbaum
"Δίκτυα Υπολογιστών", Κλειδάριθμος, 2003.
Κεφάλαια 1,7.1

[4] Tony Bourke
"Server Load Balancing", O' Reilly, 2001.

[5] Chandra Kopparapu
"Load Balancing Servers, Firewalls, and Caches", John Wiley & Sons, 2002.

[6] Evan Marcus, Hal Stern
"Blueprints for High Availability, Second Edition",.
John Wiley & Sons, 2003

[7] Joel L. Wolf, and Philips S. YU
"On Balancing the Load in a Clustered Web Farm"
IBM T. J. Watson Research Center

[8] Robert Shimonski
"Windows Server 2003, Clustering & Load Balancing.
McGraww-Hill, 2003

[9] Β. Μάγκλαρης, Τ. Χιώτης, Θ. Καρούνος, Φ. Σταματελόπουλος
"Διαχείριση Δικτύων Υπολογιστών", ΕΜΠ, 1994

[10] Yiping Ding
"Performance Impact of Load Balancers on Server Farms"
BMC Software

[11] T. Schroeder, S. Goddard, and B. Ramamurthy
"Scalable Web Server Clustering Technologies"
IEEE Network, pp. 38-45, May/June 2000

[12] V. Cardellini, M. Colajanni, and P. S. Yu
"Dynamic Load Balancing on Web-Server Systems,"
IEEE Internet Computing, vol. 3, no. 3, May/June 1999

[13] O. P. Damani, P. E. Chung, Y. Huang, C. Kintala, and Y. M. Wang
"ONE-IP: Techniques for Hosting a Service on a Cluster of Machines,"
Computer Networks and ISDN Systems, vol. 29, 1997.

[14] D.M.Dias, W. Kish, R. Mukherjee, and R. Tewari
"A Scalable and Highly Available Web Server,"
In *Proceedings of the IEEE Computer Society International Conference '96*, February 1996

[15] M. Colajanni, P. Yu, and D. Dias
"Scheduling Algorithms for Distributed Web Servers,"
In *Proceedings of IEEE International Conference on Distributed Computing Systems*, May 1997.

[16] B. A. Shirazi A. R. Hirson, and K. M. Kavi,
"Scheduling and Load Balancing in Parallel and Distributed Systems,"
IEEE CS Press, 1995

[17] D. L. Eager, E. D. Lazowska, and J. Zahorjan
"Adaptive Load Sharing in Homogenous Distributed Systems,"
IEEE Transactions on Software Engineering, vol. 12, no. 5, 1986.

[18] Cisco Systems
"Scaling the Internet Web Servers"
Http://www.cisco.com/warp/public/751/lodir/scale_wp.htm. November 1997

[19] Aweya J. Ouellette M. Montuno DY. Doroy B. Felske K.
An adaptive load balancing scheme for web servers
International Journal of Network Management, Jan-Feb 2002, v.12

[20] Valeria Cardellini, Emiliano Casalicchio, Michele Colajanni, Philip S. Yu
The State of the Art in Locally Distributed Web-server Systems
IBM Research Report, RC22209 (W0110-048) October 16, 2001

[21] Performance Evaluation of Static and Dynamic Load Balancing Schemes for a Parallel Computational Fluid Dynamics Software (CFD) Application (FLUENT) Distributed across Clusters of Heterogeneous Symmetric Multiprocessor Systems
IBM Red books, © Copyright IBM Corp. 2004.

[22] Xiao Qin, Hong Jiang, Yifeng Zhu, David R. Swanson
Boosting Performance for I/O intensive workload by preemptive job migrations in a cluster system
Department of Comp. Science, University of Nebraska-Lincoln

[23] Arun Iyengar, Erich Nahum, Anees Shaikh, Renu Tewari
Enhancing web performance
IBM T.J. Watson Research Center

[24] Qi Zhang Alma Riska Wei Sun Evgenia Smirni Gianfranco Ciardo
Workload-Aware Load Balancing for Clustered Web Servers
Department of Computer Science College of William and Mary

[25] Sys Master
L7 Load balancing
www.sysmaster.com

[26] Gianfranco Ciardo Alma Riska Evgenia Smirn
EquiLoad: a load balancing policy for clustered web servers
Department of Computer Science College of William and Mary

[27] Bhaskar Gunda
High Availability And Disaster Recovery Strategies White Paper
Open Systems Technologies Grand Rapids, Michigan

[28] Y.B. Lee and P.C. Wong
Designing a Server Array System for Multimedia World-Wide-Web Services
Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong

[29] VALERIA CARDELLINI University of Rome Tor Vergata
MICHELE COLAJANNI University of Modena
PHILIP S. YU IBM T.J. Watson Research Center
DYNAMIC LOAD BALANCING ON WEB-SERVER SYSTEMS

[30] ARUN IYENGAR, JIM CHALLENGER, DANIEL DIAS, AND PAUL DANTZIG
High-Performance Web Site Design Techniques
IBM T.J. Watson Research Center

[31] GEOFFREY C. FOX AND WOJTEK FURMANSKI
PETAOPS AND EXAOPS: SUPERCOMPUTING ON THE WEB
SYRACUSE UNIVERSITY

ΙΣΤΟΣΕΛΙΔΕΣ

1. Cisco Systems Inc.
http://www.cisco.com

2. Dell Systems
http://www.dell.com

3. Dell Power Solutions Magazine
http://www.dell.com/powersolutions

4. Microsoft
http://www.microsoft.com

4. IBM
http://www.ibm.com.

5. Communication Solutions (Διμηνιαίο περιοδικό για Business Data Networking)
http://www.comsol.gr.

6. Ιστοσελίδα με λίστα από πρωτόκολλα
http://www.protocols.com

7. On line βιβλιοθήκη
http://www.wikipedia.org
http://el.wikipedia.org

8. Linux Virtual Server project
http://linuxvirtualserver.org

9. www.websitegear.com

10. Corporation for National Research Initiatives (CNRI)
http://www.cnri.reston.va.us/index.html

11. Friedrich Alexander University
http://www7.informatik.uni-erlangen.de/~ksjh/research/cluster/

12. Load Sharing Technologies
http://ntrg.cs.tcd.ie/undergrad/4ba2.01/group8/index.html

13. Ιστοσελίδα Michael Hasestain
http://www.hasenstein.com/linux-ip-nat/diplom/node1.html

14. Citrix
www.netscaler.com

15. Sys Master
www.sysmaster.com