ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΠΟΛΥΜΕΣΩΝ

# Peer-to-Peer Συστήματα

## Συγκριτική Μελέτη BitTorrent-Gnutella

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

**Μπαλιούσκα Πέτρου**

**Α.Μ. 63**

\

**Επίβλέπουσα:Φραγκοπούλου Παρασκευή**

# Περίληψη

Τα συστήματα Peer-to-Peer (P2P) είναι συστήματα δικτύων, στα οποία οι κόμβοι συμμετέχουν ισότιμα στο δίκτυο, εν αντιθέσει με τα κεντρικοποιημένα συστήματα, στα οποία είναι απαραίτητη η παρουσία ενός κεντρικού κόμβου για να επιτευχθεί η επικοινωνία μεταξύ των κόμβων του δικτύου.

Στην παρούσα εργασία παρουσιάζονται τα δυο σημαντικότερα και πιο διαδεδομένα διομότιμα συστήματα, το BitTorrent και το Gnutella, ενώ στην συνέχεια διεξάγουμε μια σύγκριση μεταξύ τους.

Συγκεκριμένα, αρχικά γίνεται μια σύντομη παρουσίαση των peer-to-peer δικτύων στα οποία κατατάσσονται το Bittorrent και το Gnutella. Ακολουθεί η παρουσίαση και η ανάλυση των δύο συστημάτων ξεχωριστά για το καθένα και στη συνέχεια παρατίθενται τα πλεονεκτήματα και τα μειονεκτήματα τους ώστε να υπάρχει ένα μέτρο σύγκρισης για το που υπερτερεί και μειονεκτεί το καθένα.

# Λέξεις Κλειδιά

Διομότιμα συστήματα, ομότιμοι κόμβοι, πρωτόκολλο BitTorrent, πρωτόκολλο Gnutella.

# Abstract

Peer-to-Peer (P2P) networks are systems in which the nodes participate equally in the network, in contrast to centralized systems, in which a server is necessary to achieve communication between the nodes.

The purpose of this work is to present the two most important and widespread peer-to-peer systems, BitTorrent and Gnutella, and then to differentiate them.

In details, we initially made a brief presentation of peer-to-peer networks classified the Bittorrent and Gnutella. Follows the presentation and analysis of two separate systems for each and then the benefits and drawbacks are listed to be a benchmark for each performance.

# KeyWords

Peer-to-Peer (P2P) sytems, peer node, BitTorrent protocol, Gnutella protocol.

# Ευχαριστίες

. . .

# Πίνακας Περιεχομένων

**3 Bittorrent**



**4 Gnutella**

**Σχήματα**

**2 Peer-to-Peer Συστήματα**



**3 Bittorrent**



**4 Gnutella**



**5 Σύγκριση Bittorrent - Gnutella**

**Παράρτημα**

**Πίνακες**

# Κεφάλαιο 1

# Εισαγωγή

## 1.1 Αντικείμενο της εργασίας

Τα διομότιμα συστήματα έχουν πρόσφατα γίνει πολύ δημοφιλή στις κοινωνικές, ακαδημαϊκές και επιχειρησιακές δραστηριότητες. Η βασική ιδέα του peer-to-peer πηγάζει από το γεγονός ότι υπάρχουν πολλοί υπολογιστές σε σπίτια και γραφεία οι οποίοι μένουν για μεγάλα διαστήματα αχρησιμοποίητοι. Ως εκ τούτου, η ιδέα του διομότιμου είναι να αξιοποιηθούν οι αχρησιμοποίητοι κύκλοι των υπολογιστών για την δημοσίευση και ανταλλαγή υλικού ή για την επεξεργασία πληροφοριών. Όντως τα peer-to-peer συστήματα έχουν γίνει συνώνυμα με δημοφιλή συστήματα όπως το Napster, Gnutella, Kazaa, και BitTorrent τα οποία δίνουν τη δυνατότητα στους χρήστες να ανταλλάσουν αρχεία πολυμέσων (π.χ., τραγούδια και ταινίες).

Πέραν της ανταλλαγής αρχείων πολυμέσων, που είναι τρομερά επιτυχής, τα διομότιμα συστήματα είναι σημαντικά και χρήσιμα και σε άλλες εφαρμογές. Ο στόχος λοιπόν της εργασίας είναι να μελετήσουμε σε βάθος τα δυο δημοφιλέστερα πρωτόκολλα ανταλλαγής αρχείων, σύμφωνα με έρευνες [1], που δεν είναι άλλα από το BitTorrent και το Gnutella. Η μελέτη και κατανόηση αυτών των πρωτοκόλλων θα συμβάλει τόσο στην εξοικίωση με τις αντίστοιχες τεχνολογίες ανταλλαγής αρχείων όσο και στην δημιουργία ενός θεωρητικού υπόβαθρου για την ανάπτυξη νέων υπηρεσιών με χρήση αυτών των πρωτοκόλλων.

## 1.2 Δομή της εργασίας

Η πτυχιακή αυτή εργασία είναι χωρισμένη στις παρακάτω θεματικές ενότητες:

**Κεφάλαιο 1** : Εισαγωγή στο θέμα της εργασίας.

**Κεφάλαιο 2**: Εισαγωγή στα P2P συστήματα. Κατηγοριοποίηση P2P συστημάτων, αρχιτεκτονικές, μηχανισμοί αναζήτησης.

**Κεφάλαιο 3**: Περιγραφή πρωτοκόλλου BitTorrent. Ιστορικά στοιχεία, αρχιτεκτονική, περιγραφή δομικών στοιχείων και χαρακτηριστικών.

**Κεφάλαιο 4**: Παρουσίαση πρωτοκόλλου Gnutella. Ιστορικά στοιχεία, αρχιτεκτονική, δομή πακέτων και περιγραφή επιμέρους πεδίων.

**Κεφάλαιο 5**: Συγκριτική μελέτη πρωτοκόλλων BitTorrent και Gnutella.

# Κεφάλαιο 2

# Peer-to-Peer Συστήματα

## 2.1 Εισαγωγή

Τα τελευταία χρόνια με την ραγδαία εξάπλωση του Internet δημιουργήθηκε μια ακόμη ανάγκη. Η ανάγκη για κοινή χρήση υπολογιστικών πόρων (αποθήκευση, υπολογιστική ισχύ, κλπ.). Έτσι συγκροτούνται δίκτυα από πολλούς υπολογιστές με σκοπό την ανταλλαγή αρχείων, συνήθως μουσικά κομμάτια, ή για ανταλλαγή εγγράφων, ή για κατανεμημένο υπολογισμό ή και για παροχή κατανεμημένων υπηρεσιών. Τέτοια δίκτυα είναι γνωστά ως peer-to-peer (P2P) δίκτυα.

Σε ένα peer-to-peer δίκτυο κάθε κόμβος που μετέχει είναι ισότιμος με κάθε άλλο και μπορεί να ενεργήσει είτε σαν πελάτης (client) είτε σαν εξυπηρέτης (server). Κινητήρια δύναμη για ανάπτυξη εφαρμογών peer-to-peer αποτελεί η αποκεντρικοποιημένη και κατανεμημένη δομή τέτοιων συστημάτων που δεν απαιτούν διαχείριση και συντήρηση, οικονομικές αξιώσεις ή άλλους νομικούς περιορισμούς. Οι κόμβοι προσαρμόζονται, αυτοδιοργανώνονται καθώς εισέρχονται ή αποχωρούν από το σύστημα, ικανοποιώντας την ιδιότητα της κλιμάκωσης και της ανοχής στις αποτυχίες [2]. Οι λειτουργίες του είναι κατανεμημένες στους κόμβους που μετέχουν σε ένα τέτοιο σύστημα, όπου εκατομμύρια διαφορετικοί χρήστες μπορούν να είναι παρόντες ταυτόχρονα.

Peer-to-peer συστήματα [3] είναι κατανεμημένα συστήματα που αποτελούνται από διασυνδεδεμένους κόμβους, ικανούς να αυτοδιοργανώνονται σε τοπολογίες δικτύου με σκοπό την κοινή χρήση πόρων όπως περιεχόμενα, κύκλους μηχανής, χώρο αποθήκευσης, και εύρος, ικανά να προσαρμόζονται στις αποτυχίες και στις παροδικές μετακινήσεις κόμβων ενώ διατηρούν προσβάσιμη συνδετικότητα και εκτελούνται χωρίς την απαίτηση για μεσολάβηση ή υποστήριξη ενός καθολικού κεντρικού εξυπηρέτη.

Οι πρώτες κατανεμημένες εφαρμογές (SMTP, FTP) εμφανίζονται στα τέλη της δεκαετίας του ’80, αρχές ’90 και αποτελούν τον πρόδρομο των peer to peer συστημάτων [4].

Στο τέλος της δεκαετίας του ’90 το internet είναι «κοινός τόπος» και η αναπτυσσόμενη τεχνολογία επιτρέπει την ανάδυση εφαρμογών μέσω των οποίων οι χρήστες ανταλλάσσουν αρχεία. Το Napster [5] είναι η πρώτη χαρακτηριζόμενη peer-to-peer εφαρμογή, όπου κεντρικοί διακομιστές διατηρούν ευρετήρια για το που βρίσκονται τα αρχεία που ο χρήστης αναζητά, και τα οποία μπορεί να κατεβάσει απευθείας από την θέση που βρίσκονται αποθηκευμένα. Μετεξέλιξη του Napster αποτελεί η Gnutella [5] όπου οι χρήστες τώρα συνδέονται μεταξύ τους για την εύρεση των επιθυμητών αρχείων.

Πολλές άλλες peer-to-peer εφαρμογές εμφανίζονται την ίδια περίοδο για την ανταλλαγή μουσικών και μη αρχείων. Τόσο η επιστημονική κοινότητα όσο και ο εμπορικός κόσμος δείχνουν έντονο ενδιαφέρον για τα peer-to-peer συστήματα τα οποία όχι μόνο γίνονται αποδεκτά αλλά υιοθετούνται ευρέως διαμοιράζοντας αρχεία ή παρέχοντας κατανεμημένο υπολογισμό [4].

Στην συνέχεια θα παρουσιάσουμε την βασική λειτουργία των peer-to-peer συστημάτων, την διαχείριση ερωτημάτων, και τις αρχιτεκτονικές που προτάσσονται για το χειρισμό των λειτουργιών αναζήτησης, δρομολόγησης, εντοπισμού πληροφορίας, που υποβάλλονται στο peer-to-peer σύστημα.

## 2.2 Αναζήτηση

Τα peer-to-peer συστήματα είναι από την φύση τους κατανεμημένα και η γενική τους χρήση είναι ο διαμοιρασμός αρχείων (file sharing). Το κλειδί για την χρησιμότητά τους, αλλά και μια μεγάλη πρόκληση από σχεδιαστική άποψη είναι η τεχνική που χρησιμοποιείται για την αναζήτηση και ανάκτηση των επιθυμητών δεδομένων. Το πρόβλημα εστιάζει [4]:

- **στον εντοπισμό των δεδομένων** - ο μηχανισμός που χρησιμοποιείται για τον εντοπισμό του επιθυμητού δεδομένου.

- **στην δρομολόγηση της ερώτησης** - η στρατηγική που καθορίζει σε πόσους και ποιους γείτονες θα σταλεί μια ερώτηση που φθάνει σε έναν κόμβο και δεν μπορεί να απαντηθεί από τον ίδιο.

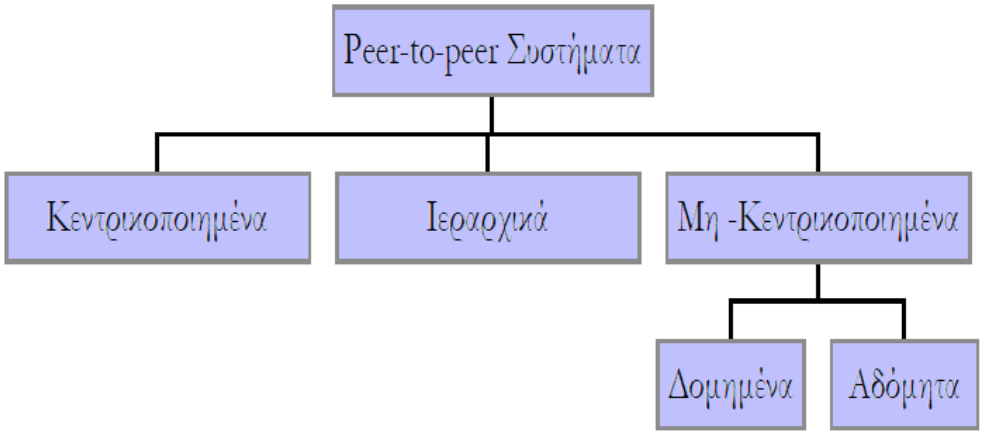
Η αποτελεσματικότητα της τεχνικής αναζήτησης για ένα συγκεκριμένο σύστημα εξαρτάται από τις ανάγκες της εφαρμογής. Παραδοσιακά τα ερωτήματα που υποστηρίζονται βασίζονται σε ένα αναγνωριστικό (ID) ή σε μια λέξη κλειδί ή σε μια κανονική έκφραση.

Η στρατηγική που χρησιμοποιείται για τον εντοπισμό και την ανάκτηση της πληροφορίας είναι κρίσιμος παράγοντας στα peer-to-peer συστήματα αφού επηρεάζει την αποτελεσματικότητα, την ανοχή και την προσαρμοστικότητα σε αποτυχίες, την αυτό-διατήρηση (nodes join and leave) και εξαρτάται από την τοπολογία του overlay δικτύου, την δόμησή του και την αρχιτεκτονική του.

## 2.3 Αρχιτεκτονική

Οι κόμβοι (πρόκειται για προσωπικούς υπολογιστές, σταθμούς εργασίας, κλπ.) που μετέχουν σε ένα peer-to-peer σύστημα σχηματίζουν ένα δίκτυο επικάλυψης (overlay network) πάνω από την υπάρχουσα υποδομή του διαδικτύου. Διασυνδέονται, επικοινωνούν και ανταλλάσσουν πληροφορίες μεταξύ τους σε τοπολογίες ανεξάρτητα από το δίκτυο υποδομής (IP network) διατηρώντας την αυτονομία τους [5]. Η αρχιτεκτονική του δικτύου επηρεάζει τον μηχανισμό δρομολόγησης μηνυμάτων αναζήτησης, την απόδοση, την ικανότητα κλιμάκωσης, την προσαρμοστικότητα - ανοχή σε σφάλματα, κλπ. και στοχεύει στην υποστήριξη λειτουργιών όπως διαμοιρασμό αρχείων (file sharing), κατανεμημένο υπολογισμό (distributed computing), επικοινωνία – συνεργασία μεταξύ των χρηστών (collaboration network).

Υπάρχουν διάφορες αρχιτεκτονικές (Σχήμα 2.1) για τον σχηματισμό του overlay δικτύου [5]:




**Σχήμα 2.1** – Peer-to-peer αρχιτεκτονικές.

### 2.3.1. Κεντρικοποιημένα peer-to-peer συστήματα

Στις κεντρικοποιημένες αρχιτεκτονικές υπάρχει ένας κεντρικός εξυπηρέτης (Directory Server) στον οποίο απευθύνουν οι κόμβοι τα ερωτήματα τους για να πληροφορηθούν που βρίσκονται οι επιθυμητές πληροφορίες (π.χ Napster). Μια τέτοια αρχιτεκτονική αν και είναι αρκετά αποδοτική, δεν έχει την ιδιότητα της κλιμάκωσης ενώ έχει ενιαίο σημείο της αποτυχίας (bottleneck) [5].

### 2.3.2. Ιεραρχικά

Οι κόμβοι οργανώνονται σε ιεραρχική δομή όπως γίνεται με τους DNS στο διαδίκτυο. Στα ιεραρχικά peer-to-peer συστήματα εισάγεται η έννοια των “super-peers”. Η δομή τους μπορεί να είναι κεντρικοποιημένη ή μη.

### 2.3.3. Μη Κεντρικοποιημένα peer-to-peer συστήματα

Μια άλλη κατηγορία αρχιτεκτονικών είναι οι μη-κεντρικοποιημένες όπου οι κόμβοι συγκροτούν το overlay δίκτυο είτε δομημένα ακολουθώντας κανόνες για τον σχηματισμό του δικτύου, είτε αδόμητα όπου δεν υπάρχει ούτε κεντρικό directory ούτε ακριβείς οδηγίες για τον σχηματισμό τοπολογίας του δικτύου και την τοποθέτηση των περιεχομένων.

#### 2.3.3.1 Δομημένα peer-to-peer συστήματα

Στα δομημένα peer-to-peer συστήματα οι κόμβοι οργανώνονται σε δομημένο γράφο για το σχηματισμό του overlay δικτύου. Στα δεδομένα αντιστοιχίζεται ένα κλειδί και η τοποθέτηση τους στους κόμβους γίνεται με προκαθορισμένο τρόπο έτσι ώστε να διευκολύνεται η αναζήτησή τους και να επιτυγχάνεται η κλιμάκωση. Η τοποθέτηση των αρχείων [5] στα χαλαρά δομημένα συστήματα βασίζεται στην εκτίμηση (on hints) για το που μπορεί να βρεθεί η αναζητούμενη πληροφορία. Στα αυστηρά δομημένα συστήματα τόσο η δόμηση του overlay δικτύου όσο και η τοποθέτηση των αρχείων είναι σαφώς καθορισμένη.

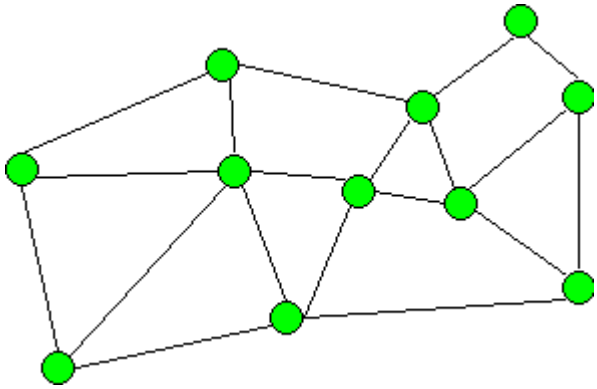
Ο εντοπισμός ενός αντικειμένου-δεδομένου από μια εφαρμογή στα δομημένα συστήματα γίνεται σε μικρό αριθμό βημάτων (network hops), υπό την απαίτηση βέβαια να διατηρείται ένας μικρός πίνακας δρομολόγησης σε κάθε κόμβο.

Παραδείγματα τέτοιων συστημάτων αποτελούν τα [5]:

- Content Addressable Network (CAN),
- Chord,
- Tapestry,
- Pastry,
- Kademlia,
- Viceroy.

#### 2.3.3.2 Αδόμητα peer-to-peer συστήματα

Στα συστήματα αυτά δεν υπάρχει καμιά δομή στο overlay δίκτυο, όπως απεικονίζεται στο Σχήμα 2.2.. Τα περιεχόμενα τοποθετούνται σε κόμβους στο δίκτυο χωρίς γνώση της τοπολογίας ή άλλης συσχέτισης με αυτό. Τα μη δομημένα συστήματα είναι κατάλληλα σε περιπτώσεις όπου μεγάλο πλήθος κόμβων μετέχει παροδικά στο δίκτυο χωρίς όμως αποδοτικούς μηχανισμούς αναζήτησης, κλιμάκωσης, διαθεσιμότητας. Υποστηρίζουν καλύτερα πολύπλοκα ερωτήματα σε σχέση με τα δομημένα [4].



**Σχήμα 2.2** – Παράδειγμα αδόμητου συστήματος. Οι πράσινοι κύκλοι συμβολίζουν τους κόμβους

Παραδείγματα αδόμητων peer-to-peer δικτύων είναι:

- Napster,
- Gnutella,
- BitTorrent,
- FastTrack,
- KaZaA.

## 2.4 Διαχείριση Ερωτημάτων [6]

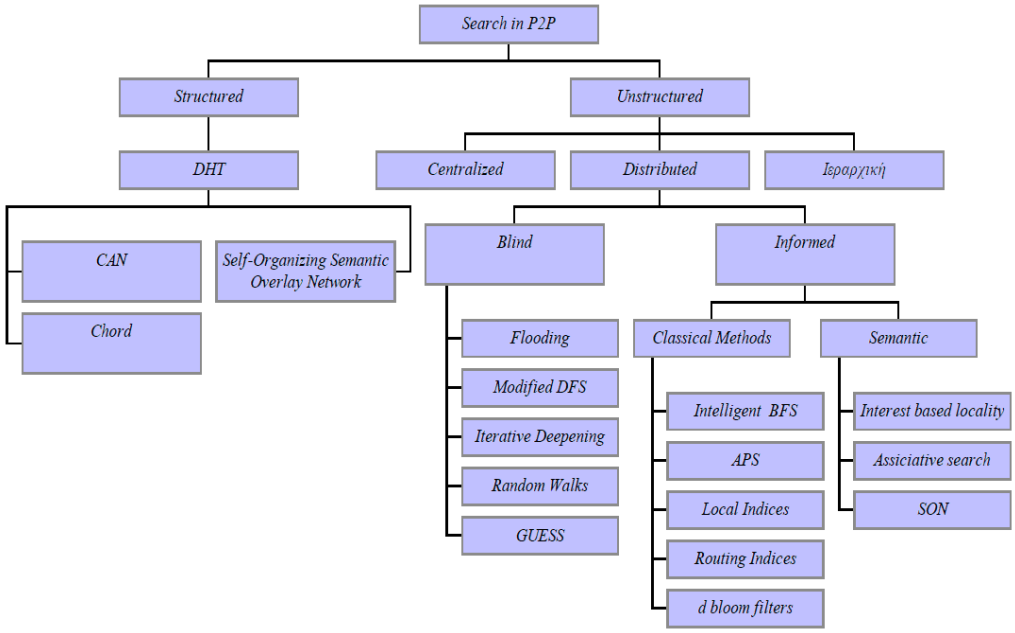
Ένα κύριο σημείο μελέτης στα peer-to-peer συστήματα είναι η διαχείριση ερωτημάτων που υποβάλλονται στο σύστημα και αποσκοπούν:

- στον εντοπισμό της επιθυμητής πληροφορίας (αρχείου, ή τμήμα του),
- την δρομολόγηση άλλων ερωτημάτων ή την προώθηση ενημερώσεων,
- τον εντοπισμό κόμβων για σύνδεση (join).

Η αναζήτηση γίνεται είτε με βάση κάποιο κλειδί (key) που αντιστοιχεί στο προς αναζήτηση δεδομένο, είτε με βάση λέξεις κλειδιά (keyword) για μεμονωμένα δεδομένα ή για περιοχή δεδομένων (range, multi-attribute queries).

Κλασσικές μέθοδοι αναζήτησης όπως κεντρικοποιημένη, ιεραρχική και πλημμύρα εφαρμόζονται και εδώ και εμφανίζονται τόσο στα δομημένα όσο και στα αδόμητα peer-to-peer συστήματα, ενώ προτείνονται επεκτάσεις ή βελτιστοποιήσεις αυτών. Οι μέθοδοι αναζήτησης διακρίνονται επίσης σε τυφλές (blind) όπου δεν υπάρχει καθόλου πληροφορία για την αναζήτηση και σε πληροφορημένες (informed)

όπου υπάρχει είτε κεντρική είτε κατανεμημένη υπηρεσία πληροφόρησης. Στο Σχήμα 2.3 απεικονίζονται οι στρατηγικές αναζήτησης στα peer -to-peer δίκτυα.




**Σχήμα 2.3** – Στρατηγικές αναζήτησης στα peer -to-peer δίκτυα.

### 2.4.1 Κεντρικοποιημένη αναζήτηση

Μια κεντρική υπηρεσία καταλόγου διατηρεί πληροφορίες τόσο για τα δεδομένα όσο και για τους κόμβους που τα διαθέτουν (είναι αποθηκευμένα σε αυτούς - Napster). Με άλλα λόγια σε μια κεντρική βάση δεδομένων καταχωρούνται μια τιμή κλειδί για το στοιχείο (π.χ. τίτλος τραγουδιού) και η θέση (ο κόμβος) που το στοιχείο βρίσκεται αποθηκευμένο. Αφού εντοπιστεί η επιθυμητή πληροφορία, οι εμπλεκόμενοι κόμβοι επικοινωνούν μεταξύ τους απευθείας για την ανάκτηση της πληροφορίας. Αν η πληροφορία είναι διαθέσιμη σε περισσότερες πηγές τότε επιλέγεται η “καταλληλότερη” όσον αναφορά το κόστος, την ταχύτητα, την διαθεσιμότητα σύμφωνα με τις ανάγκες του χρήστη.

Η αναζήτηση με τον τρόπο αυτό είναι ταχύτατη – επιτυγχάνεται σε ένα βήμα – όμως δεν παύει η κεντρικοποιημένη δομή της, να αποτελεί περιορισμό στην κλιμάκωση και κεντρικό σημείο αποτυχίας για τα συστήματα αυτά.

### 2.4.2 Ιεραρχική αναζήτηση

Βασίζεται στο παραδοσιακό σχήμα που χρησιμοποιείται στο διαδίκτυο για τον εντοπισμό μιας IP διεύθυνσης με την χρήση Domain Name System (DNS). Στα peer-to-peer συστήματα ειδικοί κόμβοι, οι superpeers, απαρτίζουν μια ιεραρχική δομή (FastTrack's/KaZaA). Η αναζήτηση αρχίζει από την κορυφή της ιεραρχίας και διασχίζει ένα μονοπάτι μέχρι τον κόμβο που περιέχει το στοιχείο (επιθυμητή πληροφορία). Όμως δεν υπάρχει καμιά εγγύηση ότι το στοιχείο θα βρεθεί.

Μια αναζήτηση μπορεί να ολοκληρωθεί σε O(logN) βήματα. Όμως μια αποτυχία ή μια αποχώρηση ενός κόμβου μπορεί να δημιουργήσει σοβαρά προβλήματα ιδιαίτερα αν βρίσκεται ψηλά στην ιεραρχία.

### 2.4.3 Τυφλή αναζήτηση

Η αναζήτηση γίνεται με τη μέθοδο της πλημμύρας (flooding). Όταν δηλαδή ζητείται το δεδομένο X τότε ο κόμβος κοιτάει πρώτα την τοπική του βάση. Αν το βρει επιστρέφει το δεδομένο, διαφορετικά προωθεί την ερώτηση στους γείτονές του (Gnutella). Αυτή η μέθοδος αναζήτησης δεν εγγυάται ότι το δεδομένο θα βρεθεί και σπαταλά πόρους τους συστήματος (εύρος, υπολογιστική ισχύς κύκλους μηχανής, κλπ.). Επίσης, πρέπει να υπάρχουν μηχανισμοί για την αποφυγή κύκλων, και το τερματισμό της αναζήτησης μετά από κάποιο αριθμό hops. Οι λύσεις που προτείνονται για το πρόωρο σταμάτημα της αναζήτησης είναι η χρήση της παραμέτρου TTL (Time to Live) και Expanding Ring.

Η παράμετρος TTL τίθεται σε μια τιμή αρχικά, συνήθως 7, και μειώνεται κατά ένα καθώς περνά το ερώτημα από κόμβο σε κόμβο. Η διάδοση των μηνυμάτων σταματά όταν αυτή η τιμή μηδενιστεί.

Εναλλακτικά, αντί για προκαθορισμένη τιμή της παραμέτρου TTL, επιλέγεται η διαδοχική αύξησή της εφόσον δεν βρεθεί η ζητούμενη πληροφορία. Η μέθοδος αυτή είναι γνωστή ως Expanding Ring. Αρχικά, η αναζήτηση ξεκινά με ένα μικρό TTL. Αν δεν βρεθεί αποτέλεσμα, αρχίζει νέα αναζήτηση με αυξημένο TTL. Η διαδικασία επαναλαμβάνεται έως ότου βρεθεί το αποτέλεσμα.

Η τυφλή αναζήτηση με χρήση TTL για μια τυπική τιμή του και με C συνδέσεις κατά μέσο όρο ανά κόμβο (4 ή 5 συνδέσεις συνήθως) παράγει μηνύματα που διακινούνται στο δίκτυο:

$$2 \cdot \sum_{i=0}^{TTL} C \cdot (C-1)^i \qquad \text{(σχ.1.1)}$$

Για να μειωθεί αυτή η πληθώρα μηνυμάτων προτείνονται παραλλαγές του μηχανισμού της πλημμύρας που επιχειρούν αποδοτικότερη διαχείριση ερωτημάτων σε συστήματα που ο αριθμός των κόμβων είναι μεγάλος:

- **Modified – BFS** - η ερώτηση προωθείται από τον κόμβο σε τυχαίο υποσύνολο των γειτόνων του.
- **Iterative Deepening** - η αναζήτηση γίνεται σε καθορισμένο βάθος (TTL, hops). Αν το αντικείμενο δεν βρεθεί τότε το ερώτημα επαναλαμβάνεται με νέες τιμές για την παράμετρο βάθους.
- **Random Walks** - για την μείωση της πληθώρας μηνυμάτων στο δίκτυο προτείνεται επίσης η χρήση ενός ή περισσοτέρων περιπατητών (walker). Ένας περιπατητής επιλέγει τυχαία έναν γείτονα σε κάθε βήμα της αναζήτησης. Για καλύτερα αποτελέσματα αναζήτησης μπορούν να χρησιμοποιηθούν περισσότεροι από ένας περιπατητές (για παράδειγμα k). Για να σταματήσουν την αναζήτηση οι περιπατητές χρησιμοποιούνται τεχνικές όπως TTL και checking. Σύμφωνα με την πρώτη, ο περιπατητής σταματά μετά από ορισμένο αριθμό βημάτων ενώ σύμφωνα με την δεύτερη περιοδικά γίνεται έλεγχος αν δόθηκε απάντηση.

### 2.4.4 Πληροφορημένη αναζήτηση

Στα peer-to-peer συστήματα προτάθηκαν διάφορες λύσεις για βελτιστοποίηση του τρόπου αναζήτησης. Οι κόμβοι που μετέχουν στο peer-to-peer σύστημα σχηματίζουν ένα είδος δομής για το overlay δίκτυο ή ένα επίπεδο πάνω από το overlay δίκτυο. Η δομή αυτή χρησιμοποιείται για την δρομολόγηση των ερωτημάτων ώστε να αποφευχθούν τα μειονεκτήματα της πλημμύρας.

#### 2.4.4.1 Δομημένα συμμετρικά σχήματα αναζήτησης

Τα συστήματα αυτά είναι δομημένα, δηλαδή η θέση των κόμβων και των δεδομένων πάνω στο σύστημα είναι συγκεκριμένη και ο προσδιορισμός της γίνεται με βάση ένα κατανεμημένο πίνακα κατακερματισμού (Distributed Hash Table -DHT) [6]. Παραδείγματα τέτοιων συστημάτων αποτελούν το CAN, Chord, Kademlia, Pastry και Viceroy. Τόσο οι κόμβοι όσο και τα δεδομένα πρέπει να έχουν ένα μοναδικό αναγνωριστικό. Τα κάθε δεδομένο (ή αντικείμενο) αντιστοιχίζεται και αποθηκεύεται σε έναν κόμβο με βάση μια συνάρτηση κατακερματισμού, επιτυγχάνοντας εν μέρει και εξισορρόπηση φορτίου (load balance).

Κατά την διαδικασία της αναζήτησης, η θέση του επιθυμητού δεδομένου εντοπίζεται με μια συνάρτηση κατακερματισμού και το ερώτημα προωθείται στους κόμβους που είναι πιο κοντά με βάση κάποια συνάρτηση απόστασης (distance function) και λαμβάνοντας υπόψη τον πίνακα δρομολόγησης (routing indices) που κάθε κόμβος διατηρεί τοπικά.

Το σύστημα Chord διατηρεί μια skiplist (ή finger tables) δομή και η αναζήτηση ολοκληρώνεται σε O(logN) βήματα αν υπάρχουν N συνδεδεμένοι κόμβοι. Στο CAN η αναζήτηση γίνεται με προώθηση του ερωτήματος στον προορισμό (γνωστό εκ των προτέρων) με χρήση greedy αλγορίθμους. Τα συστήματα Kaldemia, Pastry και Tapestry διατηρούν δεντρική δομή (tree) και η αναζήτηση απαιτεί $O(\log_{2^b} N)$ όπου b παράμετρος του αλγορίθμου που συνήθως έχει τιμή 4 και N όπως και παραπάνω οι συνδεδεμένοι κόμβοι. Το Viceroy peer-to-peer σύστημα έχει δομή πεταλούδας (butterfly) και η αναζήτηση διαρκεί το πολύ O(logN) βήματα.

Το Semantic Overlay Network αποτελεί μια άλλη προσέγγιση βασισμένη στους DHTs και επεκτείνοντας την ιδέα του CAN προτείνει την τοποθέτηση των δεδομένων στο χώρο όχι με βάση το ID τους αλλά με βάση το περιεχόμενό τους. Έτσι επιτυγχάνεται ένα είδος εννοιολογικής ομαδοποίησης των δεδομένων σε ένα overlay δίκτυο. Για κάθε έγγραφο (document) δημιουργείται ένα διάνυσμα (με την βοήθεια του Latent Semantic Index - LSI) και με βάση αυτό τοποθετείται στον CAN χώρο είτε το ίδιο είτε ο δείκτης του.

Όταν υποβάλλεται ένα ερώτημα σε κάποιο κόμβο, τότε παράγεται ένα εννοιολογικό διάνυσμα ερωτήματος και δρομολογείται στο overlay δίκτυο με την μέθοδο της πλημμύρας σε μια ακτίνα r. Όλοι οι κόμβοι που λαμβάνουν το ερώτημα

ελέγχουν τοπικά για το έγγραφο που ταιριάζει καλύτερα και απαντούν στον αιτούντα κόμβο.

Το μέσο μήκος του μονοπατιού δρομολόγησης είναι όπως και στο CAN $\left( d / 4 \cdot n^{\frac{1}{d}} \right)$ όπου d το πλήθος των διαστάσεων και n το πλήθος των κόμβων. Όμως, όσο το πλήθος των διαστάσεων αυξάνει τα αποτελέσματα δεν είναι τα επιθυμητά. Για το λόγο αυτό, προτάσσονται δυο βελτιώσεις: Rolling Index και Content – Direct Search. Η χρήση του Rolling Index επιλύει το πρόβλημα της ασυμφωνίας μεταξύ του εννοιολογικού και του CAN χώρου ως προς τις διαστάσεις με την δημιουργία rotate vectors που βασίζεται στην εκτίμηση για αποτελεσματική διαμέριση του CAN [6]. Η Content – Direct Search χρησιμοποιείται για καθοδηγούμενη αναζήτηση έτσι ώστε να μειωθεί ο αριθμός των επισκεπτόμενων κόμβων.

#### 2.4.4.2 Routing Indices (RIs)

Πρόκειται για έναν μηχανισμό κατανεμημένου ευρετηρίου που σε κάθε κόμβο αποθηκεύει ένα μικρό τμήμα του. Σκοπός των RIs είναι να δείξουν την κατεύθυνση που βρίσκεται το αντικείμενο και όχι την πραγματική του θέση. Όταν υποβάλλεται ένα ερώτημα σε έναν κόμβο, εκείνος κοιτάζει την λίστα των γειτόνων του και το προωθεί σε αυτόν με την μεγαλύτερη “goodness” τιμή. Η έννοια “goodness” μπορεί να θεωρηθεί ως ο αριθμός των σχετικών με το ζητούμενο αντικείμενο αρχείων. Τρία εναλλακτικά σχήματα παρουσιάζονται:

- **compound RI** - για κάθε πιθανό μονοπάτι συναθροίζεται η “goodness” για το αντικείμενο και προκύπτουν οι εγγραφές του ευρετηρίου.
- **hop-count RI** - διατηρούνται αρχεία συνάθροισης για ένα ορισμένο αριθμό από hops.
- **exponential RI** - τα αποτελέσματα του RI πίνακα προκύπτουν ως αποτέλεσμα μιας φόρμουλας κόστους πάνω στο hop-count RI.

### 2.4.4.3 Εννοιολογική προσέγγιση

Η βασική ιδέα αυτής της οργάνωσης είναι απλή: αν ένας κόμβος διαθέτει περιεχόμενο (π.χ. έγγραφο, μουσικό κομμάτι, κλπ.) που ενδιαφέρει και κάποιον άλλον κόμβο, τότε είναι πολύ πιθανό να έχει επίσης και άλλα περιεχόμενα που ενδιαφέρουν επίσης τον άλλον. Μπορεί λοιπόν να δημιουργηθεί ένα δίκτυο μέσω ενός πρωτοκόλλου αυτοδιοργάνωσης.

Μερικές τέτοιες προσεγγίσεις είναι:

- Interest – Based locality,
- Associative Search,
- Semantic Overlay Network (SONs).

**Interest – Based locality**

Σύμφωνα με την προσέγγιση αυτή οι κόμβοι που έχουν κοινά ενδιαφέροντα δημιουργούν απευθείας συνδέσεις μεταξύ τους (shortcuts), τις οποίες και χρησιμοποιούν για τον εντοπισμό δεδομένων. Τα shortcuts παρέχουν μια χαλαρή δομή πάνω από το υπάρχον overlay δίκτυο (Gnutella). Αν μια αναζήτηση μέσω αυτών (shortcuts) αποτύχει τότε χρησιμοποιείται το overlay δίκτυο. Για παράδειγμα έστω ότι ένας κόμβος X αναζητά τα αρχεία file1, file2, file3 και σε κάποιους κόμβους εντοπίζει κάποια από αυτά, ενώ στον κόμβο Y εντοπίζει και τα τρία αρχεία. Σκοπός είναι το σύστημα να εντοπίζει τέτοιους κόμβους και να κατεβάζει τα αρχεία που επιθυμεί απευθείας από αυτούς.

Το ερώτημα είναι πως θα ανακαλυφθούν οι κόμβοι αυτοί για να μπουν στην shortcuts λίστα του κόμβου και πως θα επιλεγούν μεταξύ άλλων οι καταλληλότεροι. Η ανακάλυψη των shortcuts όταν ένας κόμβος εισέρχεται στο σύστημα για πρώτη φορά γίνεται μέσω ερωτημάτων από τον κόμβο προς τους γείτονές με την μέθοδο της πλημμύρας. Από αυτά που βρίσκει επιλέγει κάποια τυχαία και τα προσθέτει στην λίστα του. Η λίστα αυτή αποθηκεύεται από τον κάθε κόμβο που δεσμεύει μνήμη για το σκοπό αυτό. Καθώς οι κόμβοι έρχονται και φεύγουν στο σύστημα και τα ενδιαφέροντα μεταβάλλονται, τα περιεχόμενα της λίστας ανανεώνονται και ενημερώνονται.

Η επιλογή των shortcuts που θα προστεθούν στη λίστα γίνεται από τον κόμβο που την διατηρεί με συνδυασμό κριτηρίων όπως η πιθανότητα «προμήθειας»

περιεχομένου, latency, μήκος μονοπατιού, διαθεσιμότητα εύρους, ποσότητα περιεχομένων, φόρτο shortcuts.

**Associative Search**

Όπως και στην προηγούμενη περίπτωση, έτσι και εδώ, οι κόμβοι οργανώνονται ή ομαδοποιούνται ένα επίπεδο πάνω από το υφιστάμενο peer-to-peer σύστημα (Gnutella). Η ομαδοποίηση των κόμβων σε μια χαλαρή τοπολογία γίνεται με βάση τα όμοια εννοιολογικά δεδομένα με στόχο να απαντά αποδοτικά σε ερωτήματα για αυτά καθώς και να υποστηρίζει partial match ερωτήματα ή ακόμη και την εύρεση σπανίων δεδομένων. Η αναζήτηση είναι καθοδηγούμενη (guided search) καθώς το ερώτημα προωθείται στους συναφείς κόμβους που σχηματίζουν κάποιο σύνολο. Το σύνολο, που ονομάζεται guide rule, σχηματίζεται από κόμβους που ικανοποιούν κάποια συνθήκη. Μια κατηγορία guide rule είναι το possession rule όπου η συνθήκη που πρέπει να ικανοποιείται είναι η παρουσία ενός δεδομένου στο τοπικό ευρετήριο. Οι κόμβοι μπορούν να μετέχουν σε διαφορετικά – περισσότερα του ενός guide rule. Ουσιαστικά η αναζήτηση εκμεταλλεύεται τις συνδέσεις μεταξύ των κόμβων και προωθεί τα ερωτήματα σε συναφείς “κοινότητες” κόμβων. Χρησιμοποιούνται δυο αλγόριθμοι για την καθοδηγούμενη αναζήτηση [6]:

- **RAPIER (Random Possession Rule)** - ο αλγόριθμος βασίζεται σε μια απλή επαναληπτική στρατηγική: Επιλέγει ένα τυχαίο στοιχείο από το τοπικό ευρετήριο και εφαρμόζει “blind” αναζήτηση στους κόμβους που ανήκουν στο “possession rule” σε προκαθορισμένο βάθος. Αν δεν βρεθεί το δεδομένο η αναζήτηση γίνεται στο υφιστάμενο δίκτυο (π.χ Gnutella) όπως αυτό την υποστηρίζει (τυφλή αναζήτηση).
- **GAS (Greedy Guide Rule)** - ο αλγόριθμος GAS είναι μια βελτιστοποιημένη έκδοση του προηγουμένου αλγορίθμου (Rapier). Ενώ πριν η επιλογή του στοιχείου από την λίστα γινόταν τυχαία, τώρα λαμβάνεται υπόψη η συνεισφορά του καθενός (πιθανότητα) στην επιτυχή αναζήτηση. Έτσι ο κανόνας καθοδήγησης που επιλέγεται είναι αυτός που έχει την μεγαλύτερη πιθανότητα για επιτυχή αναζήτηση.

**Semantic Overlay Network**

Τα Semantic Overlay Networks στηρίζονται στη λογική ότι θα είναι αποδοτικότερο η δρομολόγηση των ερωτημάτων να γίνεται μόνο προς τους κόμβους που είναι πιο πιθανό να έχουν απαντήσεις. Για να επιτευχθεί αυτό οι κόμβοι με εννοιολογικά όμοια περιεχόμενα σχηματίζουν cluster. Η κατηγοριοποίηση γίνεται ιεραρχικά και σχηματίζουν ένα overlay δίκτυο σε ανώτερο επίπεδο, με τους κόμβους να καταλαμβάνουν θέσεις στην ιεραρχία ανάλογα με το πλήθος των δεδομένων (files) που κατέχουν αλλά και την εννοιολογική τους σημασία. Ο σχηματισμός αυτός δεν αποκλείει τα λάθη, δηλαδή τις εσφαλμένες κατηγοριοποιήσεις όταν η πληροφορία δεν είναι αρκετή.

Όταν ένα ερώτημα εισάγεται σε ένα τέτοιο σύστημα, αρχικά θα πρέπει και το ίδιο να κατηγοριοποιηθεί και στη συνέχεια να προωθηθεί στο αντίστοιχο συστατικό (τμήμα της ιεραρχίας) για να απαντηθεί. Αν το αποτέλεσμα δεν είναι αρκετό (η απάντηση δεν βρέθηκε, ή δεν βρέθηκαν όλα τα δεδομένα) τότε το ερώτημα προωθείτε σε ανώτερα επίπεδα.

### 2.4.5 Σύγκριση μεθόδων αναζήτησης

Γενικά ο εντοπισμός δεδομένων στα peer-to-peer συστήματα απαιτεί μεγάλο εύρος αν η αναζήτηση γίνεται μέσω κεντρικού server (Napster), υψηλό overhead αν γίνει αναζήτηση flooding (Gnutella) και υψηλό κόστος ενημέρωσης αν διατηρούνται πολλά αντίγραφα των δεδομένων για πιο σύντομη αναζήτηση – ανάκτηση.

Στα συστήματα που βασίζονται σε πίνακες κατακερματισμού η αναζήτηση απαιτεί O(logN) βήματα κατά μέσο όρο με την προϋπόθεση ότι υπάρχει εξισορρόπηση φορτίου και ομοιόμορφη κατανομή ερωτήσεων. Στα ιεραρχικά συστήματα η δρομολόγηση ερωτήσεων απαιτεί περίπου O(logN) βήματα αν το δέντρο της ιεραρχίας είναι ισορροπημένο (balanced).

Μετά την παρουσίαση των αλγορίθμων αναζήτησης το συμπέρασμα είναι ότι κλειδί για μία μέθοδο αναζήτησης είναι ο μικρός αριθμός κόμβων που προσπελαύνονται όσο πιο γρήγορα είναι δυνατό και με το λιγότερο δυνατό overhead. Σε αυτό συμβάλλουν και οι προτασσόμενες βελτιστοποιήσεις που περιλαμβάνουν προσαρμοστικές μεθόδους για τον τερματισμό των αλγορίθμων, την απάλειψη κύκλων στην δρομολόγηση μηνυμάτων, και στη μικρή αύξηση του αριθμού των

κόμβων που επισκέπτεται ο αλγόριθμος σε κάθε βήμα. Στον Πίνακα 1.1 απεικονίζεται συνοπτικά η σύγκριση μεταξύ δομημένων και αδόμητων p2p δικτύων.

| | Δομημένα (Structured) | | Μη Δομημένα (Unstructured) | | | |
|---|---|---|---|---|---|---|
| | **CAN** | **Chord** | **Napster** | **Gnutella** | **FreeNet** | **FastTrack /KaZaA** |
| **Architecture** | d-dimension ID coordination space | Ring | Central | General Graph, Flat & Ad-Hoc Network | General Graph | General graph |
| **Parameter** | $N$- number of nodes<br>$d$ number of dimensions | $N$- number of nodes | None | None | None | None |
| **Search Protocol** | (Key, Value) maps point in space | Matching Key with node ID | Centralized | Query Flooding | Key, Descriptive text search | Super Peers |
| **Time Complexity** | $O(dn^{1/d})$ | O(logN) | O(1) | Unbounded – No guarantee to locate data TTL limits, if data located | Hops - to -Live limits | - |
| **Space Complexity** | 2d | logN | Constant | Constant | Constant | Centralized |
| **Peers join & leaves** | 2d | $(\log N)^2$ | Constant | Constant | Constant | Centralized |
| **Functionality** | DHT based | | Centralized | Flat topology | Loosely DHT | peers connected to Super-Peers |

**Πίνακας 1.1** – Σύγκριση μεταξύ δομημένων και αδόμητων p2p δικτύων.

## 2.5 Ασφάλεια, Ανωνυμία, Έλεγχος Πρόσβασης

Σε αυτή την παράγραφο περιγράφουμε θέματα σχετικά με την ασφάλεια, ανωνυμία, εμπιστοσύνη και έλεγχο πρόσβασης στα peer-to-peer συστήματα.

Η ασφάλεια [7] είναι ιδιαίτερη απαίτηση στα peer-to-peer συστήματα, καθώς υπάρχει ανάγκη για διαθεσιμότητα, μυστικότητα, εμπιστοσύνη, ακεραιότητα, αυθεντικότητα. Εξ αιτίας της αυτονομίας και της ανοικτής δομής τους τα συστήματα αυτά είναι ιδιαίτερα ευάλωτα σε επιθέσεις.

Το ενδιαφέρον επικεντρώνεται στην ασφαλή αποθήκευση δεδομένων και στην ασφαλή δρομολόγηση [7]. Η ασφαλή αποθήκευση με την χρήση παραδοσιακών και νέων πρωτοκόλλων και αλγορίθμων κρυπτογραφίας (ψηφιακές υπογραφές, multi-key

encryption, firewalls) με σκοπό να χτισθεί εμπιστοσύνη μεταξύ των κόμβων και των κοινόχρηστων αντικειμένων. Η ασφαλή δρομολόγηση επιτυγχάνεται με τεχνικές ελέγχου αυθεντικότητας και ακεραιότητας, την ασφαλή αντιστοίχηση ID, κλπ.

Το χαρακτηριστικό της ανωνυμίας επιτρέπει στους χρήστες να χρησιμοποιούν το peer-to-peer σύστημα διατηρώντας όμως την ανωνυμία τους, αποφεύγοντας έτσι τυχόν νομικού περιορισμούς (πρόβλημα που αντιμετώπισε το Napster). Η ανωνυμία [7] αναφέρεται στο συγγραφέα (ή εκδότη), στην ταυτότητα του κόμβου και του αντικειμένου και στις λεπτομέρειες του ερωτήματος.

Για την επίτευξη της ανωνυμίας χρησιμοποιούνται τεχνικές [7] όπως multicasting, Covert paths, ανώνυμες συνδέσεις κλπ.

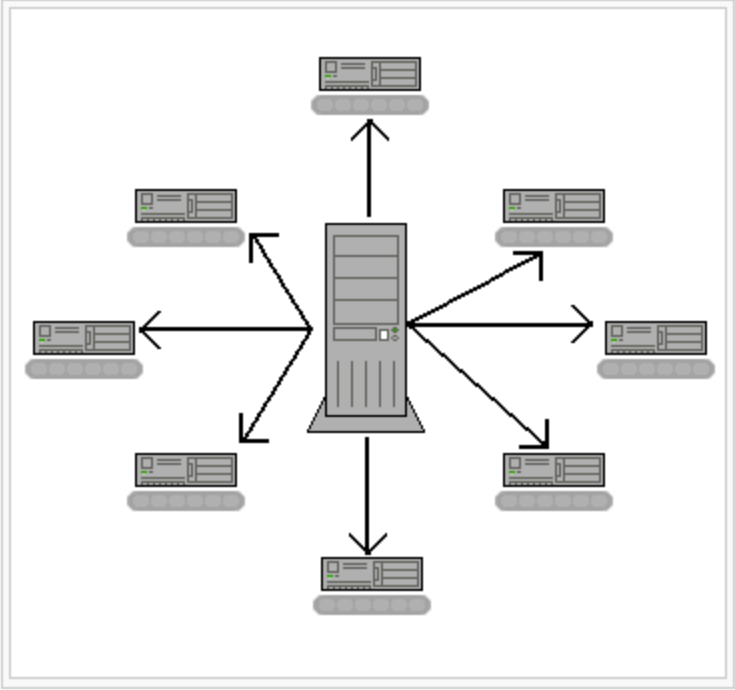
Ο έλεγχος πρόσβασης, η πιστοποίηση, η διαχείριση ταυτοτήτων είναι θέματα που δεν έχει δοθεί ιδιαίτερη σημασία [3]. Αφού το περιβάλλον είναι κατανεμημένο είναι δυνατό οι ίδιες φυσικές οντότητες να εμφανίζονται με διαφορετικές ταυτότητες. Τελικά ο έλεγχος πρόσβασης και η πιστοποίηση προκύπτει από την κατανεμημένη δομή όπου η ευθύνη και ο έλεγχος περνά στους κόμβους.

# Κεφάλαιο 3

# BitTorrent

## 3.1 Η λύση του προβλήματος μεταφοράς αρχείων

Έστω ότι κάποιος διαθέτει ένα αρχείο, το οποίο επιθυμούν να αποκτήσουν πολλοί. Καθιστώντας το διαθέσιμο μέσω HTTP (Σχήμα 3.1) όλο το κόστος της φόρτωσης (uploading) το επωμίζεται το φιλοξενούν (hosting) μηχάνημα, δηλαδή όσο πιο δημοφιλές γίνεται το αρχείο, τόσο αυξάνουν και οι απαιτήσεις των πελατών σε χωρητικότητα διαύλου. Συνεπώς, αν το αρχείο γίνει υπερβολικά δημοφιλές και ο αριθμός των πελατών που το μεταφορτώνουν μεγάλος, ο εξυπηρετητής θα καταρρεύσει και τελικά το αρχείο δεν θα είναι πλέον διαθέσιμο σε κανέναν. Η λύση αυτού του φαύλου κύκλου είναι το πρωτόκολλο BitTorrent.

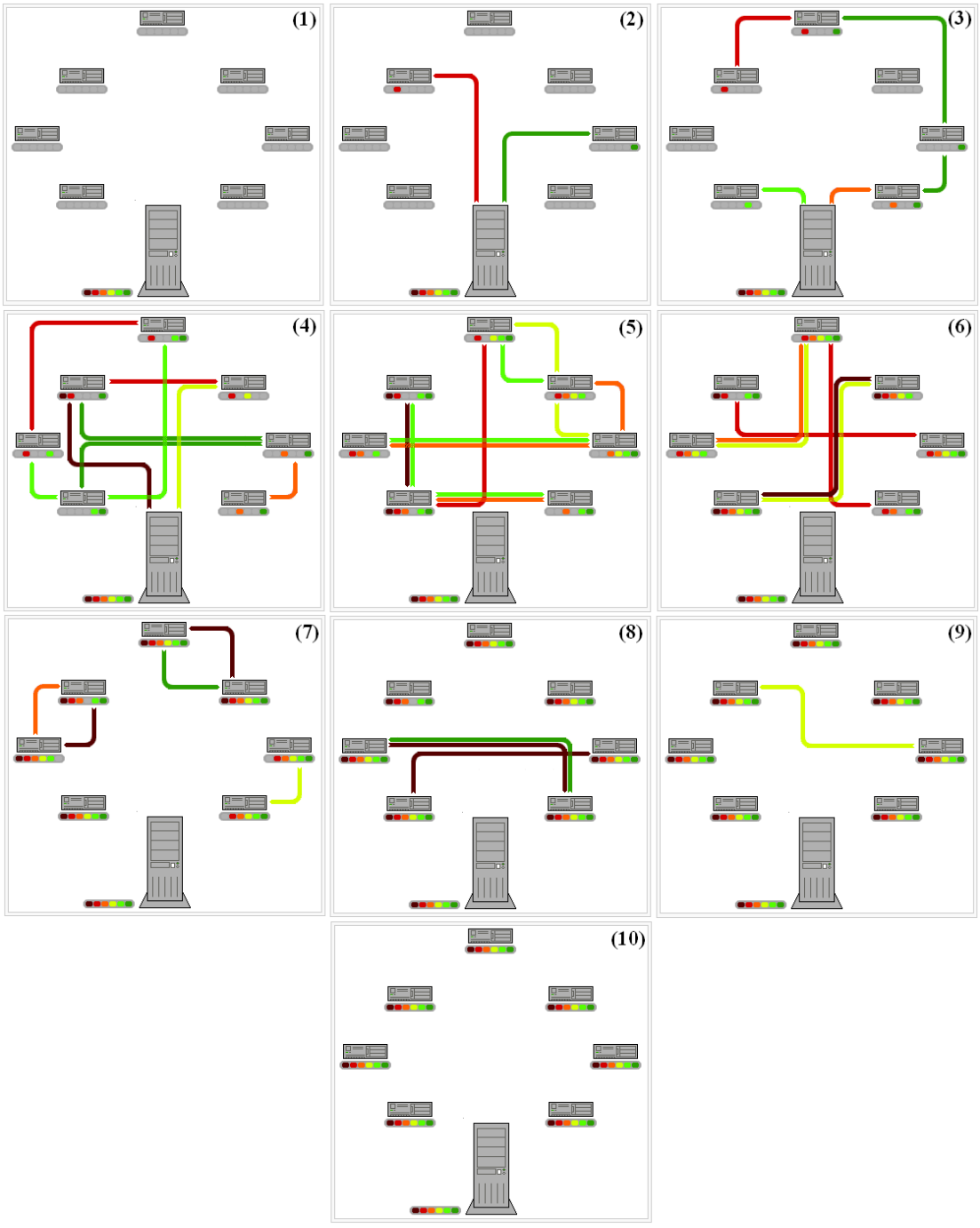


**Σχήμα 3.1** – Κλασσικός τρόπος μεταφοράς αρχείων.

Το BitTorrent είναι ένα P2P πρωτόκολλο διαμοιρασμού αρχείων (file sharing protocol) που χρησιμοποιείται για τη διανομή μεγάλης ποσότητας δεδομένων. Είναι ένα από τα πιό κοινά πρωτόκολλα για τη μεταφορά μεγάλων αρχείων και εκτιμάται ότι αντιπροσώπευε περίπου το 27-55% της συνολικής κίνησης του Internet (ανάλογα με τη γεωγραφική θέση) τον Φεβρουάριο του 2009 [8]. Δημιουργήθηκε το 2002 από τον Bram Cohen. Η πρώτη του εμφάνιση έγινε στο CodeCon [9] ενώ από τότε έχει γίνει ιδιαίτερα διάσημο και για νόμιμο αλλά και για παράνομο downloading.

Η βασική αρχή λειτουργίας του απεικονίζεται συνοπτικά στο Σχήμα 3.2 [8]. Όταν ένας χρήστης κατεβάζει δεδομένα, ταυτόχρονα τα διαθέτει και σε άλλους χρήστες. Όλοι οι χρήστες που κατεβάζουν τα ίδια δεδομένα σχηματίζουν ένα δίκτυο και γνωρίζουν ο ένας την ύπαρξη του άλλου μεσώ ενός ανιχνευτή (tracker) στον οποίο συνδέονται περιοδικά και οποίος τους αποστέλλει τη λίστα με τους άλλους χρήστες. Στο δίκτυο που σχηματίζεται κάποιοι χρήστες έχουν διαθέσιμο το σύνολο των δεδομένων που διανέμεται και έχουν το ρόλο μόνο του διανομέα (seed). Ο εξυπηρετητής ανίχνευσης (tracker) είναι ο κεντρικός κόμβος ενός συστήματος διανομής που βασίζεται στο πρωτόκολλο BitTorrent. Ένας χρήστης, που επιθυμεί να κατεβάσει κάποιο διαθέσιμο αρχείο δεδομένων από το σύστημα, συνδέεται σε αυτόν και λαμβάνει τη λίστα των χρηστών που κατεβάζουν το ίδιο αρχείο και έτσι ο νέος χρήστης εισάγεται στο δίκτυο διανομής. Με τον τρόπο αυτό μειώνεται ο φόρτος του κεντρικού εξυπηρετητή αλλά και οι απαιτήσεις σε εύρος ζώνης του δικτύου. Ένας χρήστης μπορεί να λάβει πολλά αρχεία ταυτόχρονα χωρίς οποιαδήποτε ιδιαίτερη απώλεια του ποσοστού μεταφοράς (transfer rate). Συνεπώς, όσο μεγαλύτερος ο αριθμός χρηστών τόσο ευκολότερα και γρηγορότερα ένα αρχείο μεταφέρεται μεταξύ των χρηστών της υπηρεσίας.

Το BitTorrent αναδύει μια νέα πολιτική διανομής αρχείων κατά την οποία οι χρήστες του καλούνται να “προσφέρουν” σε άλλους χρήστες τα αρχεία που “κατεβάζουν”. Με τον τρόπο αυτό επιτυγχάνεται η μέγιστη χρησιμοποίηση του εύρους ζώνης όλων των χρηστών που παίρνουν μέρος στη διανομή των αρχείων. Να σημειωθεί ότι το BitTorrent λειτουργεί μόνο ως πρωτόκολλο μεταφοράς αρχείων και δεν προσφέρει τη δυνατότητα αναζήτησης αρχείων [10]. Παρακάτω περιγράφονται λεπτομερώς ο τρόπος λειτουργίας, το πρωτόκολλο καθώς και τα πλεονεκτήματα της τεχνολογίας αυτής.




**Σχήμα 3.2** –Μεταφορά αρχείων με το πρωτόκολλο BitTorrent [8].

## 3.2 Αρχιτεκτονική του BitTorrent

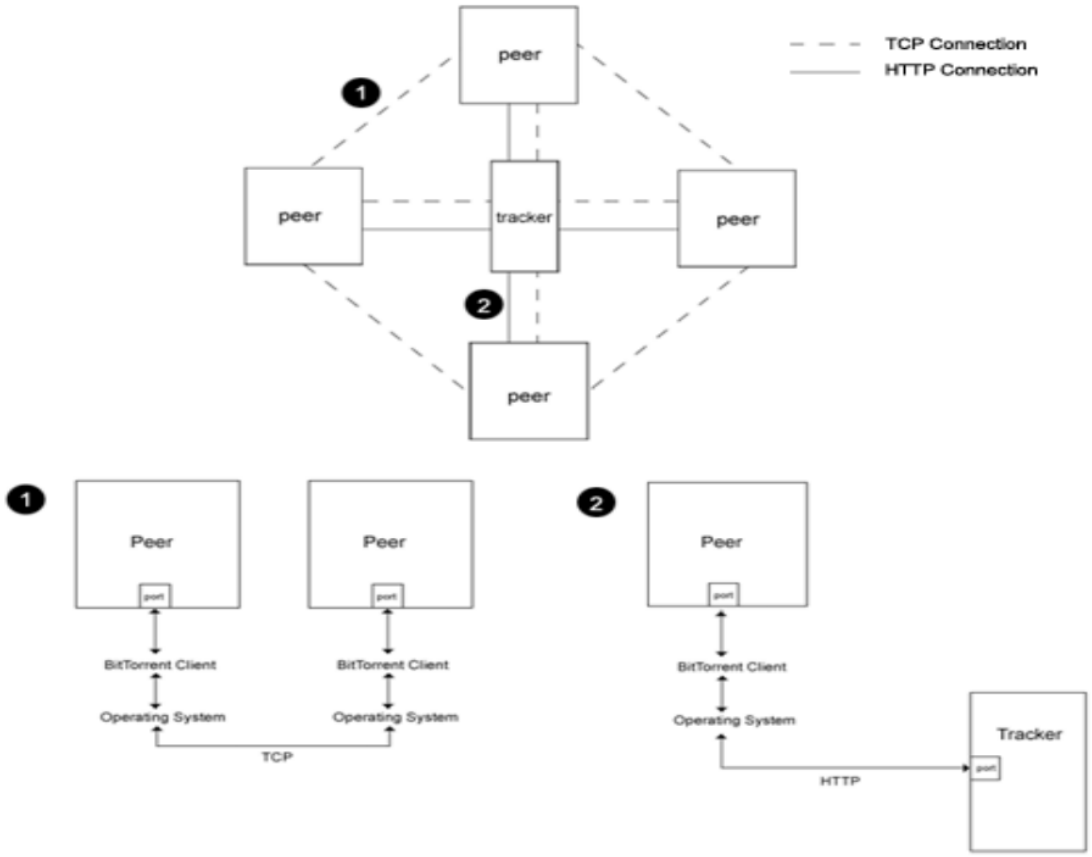
Το πρωτόκολλο Bittorrent μπορεί να αναλυθεί σε πέντε βασικά δομικά στοιχεία [10]:

- **Metainfo File** – ένα αρχείο που περιέχει όλες τις απαραίτητες λεπτομέρειες για να λειτουργήσει το πρωτόκολλο.
- **Tracker** – ένας εξυπηρετητής που βοηθάει στη διαχείρηση του πρωτοκόλλου BitTorrent.
- **Peers** – οι χρήστες που ανταλλάσουν δεδομένα μέσω του πρωτοκόλλου BitTorrent.
- **Data** – τα αρχεία που μεταφέρονται μέσω του πρωτοκόλλου.
- **Client** – το πρόγραμμα που εγκαθιστάται στον υπολογιστή ενός χρήστη και υλοποιεί το πρωτόκολλο.

Οι χρήστες (peers) χρησιμοποιούν το πρωτόκολλο TCP (Transport Control Protocol) για να επικοινωνήσουν και να ανταλλάξουν δεδομένα. Αυτό το πρωτόκολλο είναι προτιμότερο από άλλα όπως το UDP (User Datagram Protocol) διότι το TCP εγγυάται την αξιόπιστη και την in-order παράδοση των δεδομένων από τον αποστολέα στο δέκτη. Το UDP δεν μπορεί να δώσει τέτοιες εγγυήσεις, και τα δεδομένα μπορούν μπερδευτούν, ή ακόμα και να χαθούν. Ο εξυπηρετητής επιτρέπει στους χρήστες να διεξάγουν ερωτήματα για το ποιος χρήστης έχει ποιο αρχείο και τους επιτρέπει να ξεκινήσουν την επικοινωνία. Οι χρήστες επικοινωνούν με τον εξυπηρετητή μέσω απλού κειμένου (plain text) μέσω HTTP (Hypertext Transfer Protocol). Το διάγραμμα που Σχήματος 3.3 απεικονίζει πως οι χρήστες αλληλεπιδρούν μεταξύ τους και επίσης πως επικοινωνούν με τον κεντρικό εξυπηρετητή.

### 3.2.1 Metainfo File

Όταν κάποιος επιθυμεί να δημοσιεύσει δεδομένα χρησιμοποιώντας το πρωτόκολλο BitTorrent, πρέπει να δημιουργήσει ένα αρχείο metainfo. Αυτό το αρχείο είναι συγκεκριμένο για τα δεδομένα που δημοσιεύονται, και περιέχει όλες τις πληροφορίες για το torrent όπως, τα δεδομένα που θα περιλαμβάνονται, και τη IP διεύθυνση του tracker που θα συνδεθεί. Ένας tracker είναι ένας κεντρικός υπολογιστής που «διαχειρίζεται» ένα torrent, και θα μελετηθεί σε επόμενη




**Σχήμα 3.3** – Αρχιτεκτονική πρωτοκόλλου BitTorrent [10].

παράγραφο. Στο αρχείο δίνεται μια επέκταση ".torrent", και τα δεδομένα εξάγονται από το αρχείο από έναν BitTorrent client. Αυτό είναι ένα πρόγραμμα που τρέχει στον υπολογιστή του χρήστη, και υλοποιεί το πρωτόκολλο BitTorrent. Κάθε αρχείο torrent πρέπει να περιέχει τις ακόλουθες πληροφορίες, (ή "κλειδιά") [10]:

- **info** – ένα λεξικό που περιγράφει το-α αρχείο-α του torrent. Είτε για ένα αρχείο, είτε τη δομή καταλόγου για περισσότερα αρχεία. Οι κρυπτογραφήσεις για κάθε κομμάτι (piece) δεδομένων, σε format SHA 1 αποθηκεύονται εδώ. Τα κλειδιά του είναι τα εξής:
    - **name** – το όνομα του αρχείου ή σε περίπτωση πολλαπλών αρχείων, το όνομα του directory που περιέχει όλα τα αρχεία (αλφαριθμητικό χαρακτήρων).
    - **piece length** – αριθμός bytes κάθε κομματιού (ακέραιος). Ο αριθμός αυτός είναι δύναμη του 2 και δεν πρέπει να είναι ούτε πολύ μεγάλος, καθώς η διακίνηση μεγάλων κομματιών δεν είναι αποδοτική, ούτε και πολύ μικρός, καθώς αυτό έχει ως επακόλουθο μεγάλο .torrent αρχείο.

- **pieces** – αλφαριθμητικό που περιέχει τη συνένωση όλων των SHA1 τιμών, μία για κάθε κομμάτι (αλφαριθμητικό byte).

  Σε περίπτωση ενός μόνο αρχείου έχουμε τα εξής:

  - **length** – το μήκος του αρχείου σε bytes (ακέραιος).
  - **md5sum** – αχρησιμοποίητο από το BitTorrent.

  Σε περίπτωση πολλαπλών αρχείων έχουμε:

  - **files** – μια λίστα από λεξικά, ένα για κάθε αρχείο. Κάθε λεξικό περιέχει τα παρακάτω κλειδιά:
    - **length**
    - **md5sum**
    - **path** – που έχει σαν τιμή μια λίστα με ένα ή περισσότερα αλφαριθμητικά που μαζί συγκροτούν τη διαδρομή του κάθε αρχείου.

- **announce** – ανακοινώνεται η URL διεύθυνση του tracker ως συμβολοσειρά (string).

Οι ακόλουθες πληροφορίες είναι προαιρετικές που μπορούν επίσης να χρησιμοποιηθούν:

- **announce-list** – χρησιμοποιείται για καταγραφή των εφεδρικών (backup) trackers.

- **ημερομηνία δημιουργίας** – ο χρόνος δημιουργίας του torrent με τον τρόπο γραφής της ώρας στο UNIX (ακέραιο πλήθος δευτερολέπτων από την 1/1/1970 00:00:00 UTC).

- **comment** – τυχόν σχόλια από το δημιουργο του torrent.

- **created by** – όνομα και έκδοση του προγράμματος που χρησιμοποιήθηκε για τη δημιουργία του αρχείου metainfo.

Τα κλειδιά αυτά δομούνται στο αρχείο metainfo με τον ακόλουθο τρόπο:

```
{'info': {'piece length': 131072, 'length': 38190848L, 'name':
'Cory_Doctorow_Microsoft_Research_DRM_talk.mp3', 'pieces': '\xcb\xfaz\r\x9b\xe1\x9a\xe1\
x83\x91~\xed@\.....', } 'announce': 'http://tracker.var.cc:6969/announce', 'creation date':
1089749086L }
```

Δεν αποστέλλονται όμως ως έχουν, δηλαδή ως απλό κείμενο. Κωδικοποιούνται πριν την αποστολή. Η κωδικοποίηση γίνεται χρησιμοποιώντας μια συγκεκριμένη μέθοδο του BitTorrent γνωστή ως “bencoding” [10].

#### 3.2.1.1 Bencoding

Το bencoding είναι ένας τρόπος ώστε να προσδιοριστούν τα δεδομένα, και χρησιμοποιείται από το BitTorrent για την αποστολή στοιχείων μεταξύ του BitTorrent client και ενός tracker. Υποστηρίζει [11]:

- **Αλφαριθμητικά** – τα αλφαριθμητικά παριστάνονται με ένα δεκαδικό πρόθεμα που φανερώνει το μήκος τους, ακολουθούμενο από άνω και κάτω τελεία και το αλφαριθμητικό, δηλαδή:

  <string length in base ten ASCII> : <string data>

- **Ακέραιοι** – για τους ακεραίους η κωδικοποίηση περιλαμβάνει το πρόθεμα i ακολουθούμενο από τον ακέραιο με βάση το δέκα και τέλος τον χαρακτήρα e, δηλαδή:

  i<base ten ASCII>e

- **Λίστες** – η αναπαράσταση των λιστών γίνεται με το χαρακτήρα l στην αρχή, τα στοιχεία της λίστας κωδικοποιημένα με κωδικοποίηση bencoding, και το χαρακτήρα e στο τέλος, δηλαδή:

  l<bencoded values>e

- **Λεξικά** – τα λεξικά αναπαρίστανται με d ακολουθούμενο από τα ζευγάρια κλειδιών και αντίστοιχων τιμών τους, ακολουθούμενα από το χαρακτήρα e, δηλαδή:

  d<bencoded string><bencoded element>e

Τα κλειδιά πρέπει να είναι κωδικοποιημένα αλφαριθμητικά ενώ οι τιμές μπορούν να είναι ακέραιοι, αλφαριθμητικά, λίστες ή ακόμα και λεξικά, επίσης κωδικοποιημένα. Ακολουθεί παράδειγμα bencoding:

```
4:spam // represents the string "spam"
i3e // represents the integer "3"
l4:spam4:eggse // represents the list of two strings: ["spam","eggs"]
d4:spaml1:a1:bee // represents the dictionary {"spam" => ["a" , "b"] }
```

Τα αρχεία .torrent βρίσκονται συνήθως μέσω ιστοσελίδων (sites) και μηχανών αναζήτησης ή μηχανών αναζήτησης των trackers [10], δεδομένου ότι το πρωτόκολλο BitTorrent δεν διαθέτει λειτουργία αναζήτησης.

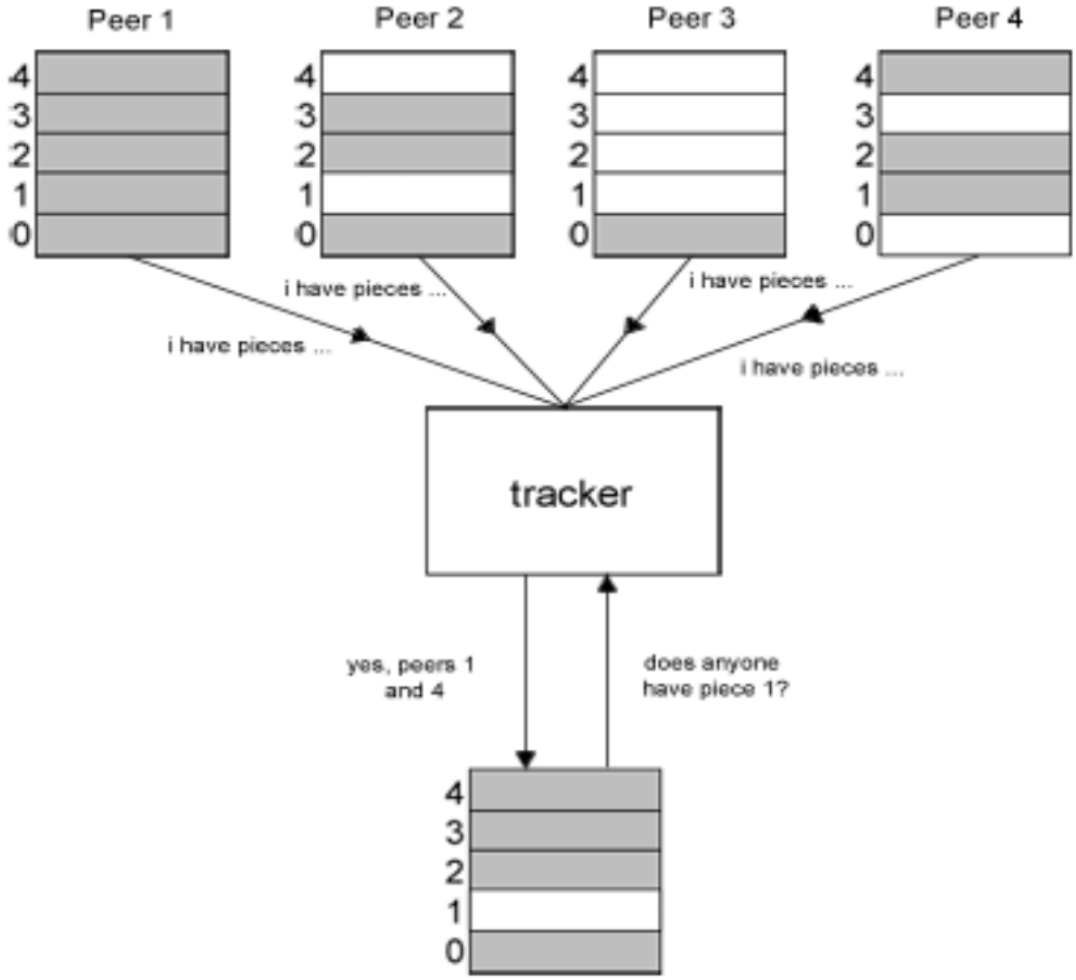
#### 3.2.1.2 Διανομή αρχείου Metainfo

Επειδή όλες οι πληροφορίες που απαιτούνται για το torrent περιλαμβάνονται σε ένα μόνο αρχείο, αυτό το αρχείο μπορεί εύκολα να διανεμηθεί μέσω άλλων πρωτοκόλλων, και καθώς το αρχείο αναπαράγεται, ο αριθμός των ομότιμων μπορεί να αυξηθεί πολύ γρήγορα. Η πιο δημοφιλής μέθοδος διανομής είναι χρησιμοποιώντας ιστοχώρους (sites) ανοιχτούς στον καθένα (οπότε μειώνεται η αξιοπιστία του περιεχομένου τους) είτε κλειστούς (κοινότητες), δηλαδή οι εγγραφές γίνονται με προσκλήσεις, ή σε πολύ συγκεκριμένες ημέρες και για περιορισμένο αριθμό θέσεων [12]. Στη δεύτερη περίπτωση, συνήθως το περιεχόμενο είναι πιο αξιοπρόσεκτο και έμπιστο, ενώ οι ταχύτητες που επιτυγχάνονται, είναι και αυτές καλύτερες. Για παράδειγμα θα αναφέρω μόνο έναν ιστόχωρο, τον linuxtracker.org, όπου διαθέτει πληθώρα λογισμικού ανοιχτού κώδικα (open-source software), διανομές Linux κτλ. Με τον τρόπο αυτό, ένας ομότιμος που διαθέτει ολόκληρο το αρχείο (seeder) θα φορτώσει το αρχείο, και έπειτα οι άλλοι μπορούν να κατεβάσουν ένα αντίγραφο αυτού μέσω του πρωτοκόλλου HTTP και να συμμετέχουν στο torrent.

### 3.2.2 Tracker

Ένας tracker χρησιμοποιείται για να διαχειριστεί τους χρήστες που συμμετέχουν σε ένα torrent (γνωστοί ως ομότιμοι χρήστες - peers). Αποθηκεύει στατιστικά για το torrent, αλλά ο κύριος ρόλος του είναι να επιτρέπει στους ομοτίμους «να βρούν ο ένας τον άλλον» και να αρχίσουν να επικοινωνούν, π.χ. να βρεί τους ομότιμους που έχουν τα δεδομένα που ζητούνται. Οι ομότιμοι δεν γνωρίζουν τίποτα ο ένας για τον άλλον μέχρι την απάντηση από τον tracker. Όταν ένας ομότιμος έρχεται σε επαφή με τον tracker, αυτός αναφέρει ποια κομμάτια ενός αρχείου κατέχουν. Με αυτό το τρόπο, όταν κάποιος άλλος ομότιμος θέσει ένα «ερώτημα» στον tracker, εκείνος του παρέχει μια τυχαία λίστα ομοτίμων που συμμετέχουν στο torrent και έχουν το-α απαιτούμενο-α κομμάτι-α, όπως απεικονίζεται στο Σχήμα 3.4.. Ο ρόλος του tracker τελειώνει μόλις οι ομότιμοι

εντοπίσουν ο ένας τον άλλον [13]. Από εκεί και πέρα, η επικοινωνία γίνεται άμεσα μεταξύ των ομοτίμων, και ο tracker δεν εμπλέκεται πλέον.




**Σχήμα 3.4** – Tracker [10].

Ο tracker είναι μια υπηρεσία HTTP/HTTPS και τυπικά εφαρμόζεται στην θύρα 6969 [10]. Η διεύθυνση του tracker που διαχειρίζεται ένα torrent καθορίζεται στο αρχείο metainfo, ενώ ένας μόνο tracker μπορεί να διαχειριστεί πολλάπλά torrents. Πολλαπλοί trackers μπορούν επίσης να καθοριστούν, ως αντίγραφα ασφαλείας (backups), τα οποία διαχειρίζονται από το BitTorrent client που τρέχει στον υπολογιστή του χρήστη. Οι BitTorrent clients επικοινωνούν με τον tracker χρησιμοποιώντας αιτήματα HTTP GET, που είναι μια standard CGI μέθοδος. Αυτό αποτελείται από την προσθήκη ενός "?" στο URL, και χωρίζοντας τις παραμέτρους με ένα "&".

#### 3.2.2.1 THP: Tracker HTTP Protocol

Το BitTorrent μπορεί να περιγραφεί με όρους δυο υπο-πρωτοκόλλων: ένα που περιγράφει τις αλληλεπιδράσεις ανάμεσα στον tracker και όλους τους clients, και ένα που περιγράφει όλες τις αλληλεπιδράσεις μεταξύ των clients. Ο tracker εφαρμόζει το ένα υπο-πρωτόκολλο, το THP (Tracker HTTP Protocol).

Το πρωτόκολλο του tracker εφαρμόζεται πάνω από το πρωτόκολλο HTTP/HTTPS. Αυτό σημαίνει ότι η μηχανή που τρέχει τον tracker τρέχει έναν εξυπηρετητή HTPP ή HTTP, και έχει τη συμπεριφορά που περιγράφεται στην συνέχεια [10]:

**1.** Ο client στέλνει ένα αίτημα GET στο URL του tracker, με ορισμένες CGI μεταβλητές και τιμές που προστίθενται στο URL . Αυτό γίνεται με τον τυποποιημένο τρόπο, δηλαδή εάν το βασικό URL είναι :
“http://some.url.com/announce”, τότε το πλήρες URL θα ήταν της μορφής:
“http://some.url.com/announce?var1=value1&var2=value2&var3=value3”.

**2.** Ο tracker αποκρίνεται με ένα “text/plain” document, που περιέχει ένα bencoded λεξικό. Αυτό το λεξικό έχει όλες τις απαιτούμενες πληροφορίες για τον ομότιμο.

**3.** Ο ομότιμος τότε στέλνει εκ νέου αιτήματα, είτε σε τακτά χρονικά διαστήματα, είτε όταν εμφανίζεται ένα γεγονός, και ο tracker αποκρίνεται.

Οι CGI μεταβλητές και τιμές που προστίθενται στο βασικό URL από τον client κατά την αποστολή ενός GET αιτήματος είναι [10]:

- **info_hash** – 20-byte SHA1 hash του κλειδιού info από το metainfo file.
- **peer_id** – 20-byte αλφαριθμητικό που χρησιμοποιείται σαν μια μοναδική ID για τον client.
- **port** – η πόρτα στη οποία ακούει ο client .
- **uploaded** – η συνολική ποσότητα δεδομένων που «ανάβηκε» από την στιγμή που ο client έστειλε το γεγονός 'started' στο tracker σε ASCII με βάση το 10.
- **downloaded** – η συνολική ποσότητα δεδομένων που «κατέβηκε» από την στιγμή που ο client έστειλε το γεγονός 'started' στο tracker σε ASCII με βάση το 10.
- **left** – ο αριθμός των bytes που ο client υπολοίπεται να «κατεβάσει» σε ASCII με βάση το 10.

- **compact** – υποδεικνύει ότι ο client δέχεται τις συμπιεσμένες απαντήσεις. Η λίστα από τους ομότιμους μπορεί να αντικατασταθεί από 6 bytes ανά ομότιμο. Τα πρώτα 4 bytes είναι ο host, και τα τελευταία 2 bytes είναι η πόρτα.
- **event** – αν υπάρχει, πρέπει να είναι ένα από τα ακόλουθα: started, stopped, completed.
- **IP** – (προαιρετικό) η IP διεύθυνση της μηχανής του client, σε διάστηκτο (dotted) format.
- **numwant** – (προαιρετικό) ο αριθμός των ομοτίμων που ο client επιθυμεί να λάβει από τον tracker.
- **key** – (προαιρετικό) επιτρέπει ένας client να αναγνωρίσει μόνος του αν η IP διεύθυνση των ομοτίμων αλλάζει.
- **trackerid** – (προαιρετικό) αν προηγούμενο announce περιείχε το trackerid, θα έπρεπε να καθοριστεί εδώ.

Ο tracker στη συνέχεια αποκρίνεται με ένα "text/plain" document με τα ακόλουθα κλειδιά [10]:

- **failure message** – αν υπάρχει, δε θα συμπεριλαμβάνονται άλλα κλειδιά. Η τιμή του είναι ένα μήνυμα σφάλματος, ικανό να διαβαστεί από άνθρωπο, που εξηγεί γιατί το αίτημα απέτυχε.
- **warning message** – παρόμοιο με το failure message, αλλά η απόκριση συνεχίζει να υφίσταται επεξεργασία.
- **interval** – ο αριθμός των δευτερολέπτων που ένας client θα πρέπει να περιμένει μεταξύ των αιτημάτων που στέλνει στον tracker.
- **tracker id** – ένα αλφαριθμητικό που ο client θα πρέπει να στείλει πίσω με την επόμενή του ανακοίνωση.
- **complete** – ο αριθμός των ομοτίμων που έχουν το αρχείο ολόκληρο.
- **incomplete** – ο αριθμός ομοτίμων που δεν είναι seeders (leechers).
- **peers** – μια λίστα από λεξικά που περιλαμβάνουν:
    - **peer id** – η ταυτότητα του κόμβου.
    - **IP** – η IP διεύθυνση (Ipv6 ή Ipv4) ή το DNS όνομα.
    - **port** – ο αριθμός της θύρας του κόμβου.

Αξίζει να σημειωθεί ότι το μέγεθος της λίστας είναι εξ’ ορισμού 50 [11]. Αν υπάρχουν λιγότεροι κόμβοι, τότε η λίστα είναι πιο μικρή. Αν υπάρχουν περισσότεροι, τότε ο tracker διαλέγει τυχαία τους κόμβους που περιλαμβάνει στην απάντηση.

#### 3.2.2.2 Scraping

Ο tracker υποστηρίζει μία υπηρεσία που ονομάζεται “scrape”. Ο σκοπός του είναι να παρέχει πληροφορίες για τα torrents που φιλοξενεί[5].

Το scraping είναι η διαδικασία υποβολής ερωτήματος για τη κατάσταση ενός δεδομένου torrent (ή όλων των torrents) που ο tracker διαχειρίζεται. Το αποτέλεσμα είναι γνωστό ως "scrape page". Για να ληφθεί η "scrape page", πρέπει να ξεκινήσεις με το announce του URL, να βρείς την τελευταία '/' και αν η λέξη που ακολουθεί το '/' είναι το 'announce' να αντικατασταθεί με το 'scrape'.
Ακολουθεί παράδειγμα:

| Announce URL | | Scrape URL |
|---|---|---|
| http://example.com/annnounce | → | http://example.com/scrape |
| http://example.com/a/annnounce | → | http://example.com/a/scrape |
| http://example.com/announce.php | → | http://example.com/scrape.php |

Ο tracker τότε αποκρίνεται με ένα “text/plain” document με τα ακόλουθα bencoded κλειδιά [10]:

- **files** – ένα λεξικό που περιλαμβάνει ένα ζεύγος κλειδιών για κάθε torrent. Κάθε κλειδί είναι φτιαγμένο από μια 20-byte δυαδική hash τιμή. Η τιμή του κλειδιού εκείνου είναι ένα εμφωλευμένο λεξικό με τα ακόλουθα κλειδιά:
- **complete** – ο αριθμός των ομότιμων που έχουν ολόκληρο το αρχείο (seeds).
- **downloaded** – πόσες φορές ολόκληρο το αρχείο έχει «κατέβει».
- **incomplete** – ο αριθμός των ενεργών χρηστών που «κατεβάζουν» (leechers).
- **name** – (προαιρετικό) το όνομα του torrent.

### 3.2.3 Ομότιμοι

Οι ομότιμοι είναι οι χρήστες που συμμετέχουν σε ένα torrent, και διαθέτουν τμήμα του αρχείου, ή το πλήρες αρχείο (γνωστό ως seed). Τα κομμάτια ζητούνται από τους ομότιμους , αλλά δεν είναι εγγυημένο ότι θα σταλθούν, αφού εξαρτόνται από τη κατάσταση του ομότιμου. Το BitTorrent χρησιμοποιεί τις θύρες του πρωτοκόλλου TCP (Transmission Control Protocol) 6881-6889 για την αποστολή μηνυμάτων και δεδομένων μεταξύ των ομοτίμων, και σε αντίθεση με τα άλλα πρωτόκολλα, δεν χρησιμοποιεί το πρωτόκολλο UDP (User Datagram Protocol).

#### 3.2.3.1 Επιλογή κομματιού

Οι ομότιμοι χρήστες συνεχώς βάζουν στην σειρά τα κομμάτια που απαιτούν για «κατέβασμα». Για το λόγο αυτό ο tracker σταθερά απαντάει σε κάθε ομότιμο με μια λίστα άλλων ομότιμων που έχουν τα κομμάτια που ζητούνται. Το ποιό κομμάτι απαιτείται εξαρτάται από το BitTorrent client. Υπάρχουν τρια επίπεδα επιλογής τμημάτων, που αλλάζουν ανάλογα με το επίπεδο περάτωσης που ο ομότιμος βρίσκεται [10].

#### 3.2.3.2 Τυχαία επιλογή κομματιού

Όταν το κατέβασμα ξεκινά, ο ομότιμος δεν έχει τίποτα να ανεβάσει, με αποτέλεσμα να επιλέγεται τυχαία ένα κομμάτι ώστε να ξεκινήσει το κατέβασμα. Επιλέγονται συνεπώς τυχαία κομμάτια μέχρι να ολοκληρωθεί και να ελεγθεί το πρώτο κομμάτι. Από την στιγμή που συμβεί αυτό, η στρατηγική αλλάζει και γίνεται αυτή του σπανιότερου πρώτου.

#### 3.2.3.3 Σπανιότερο πρώτο

Κατά την επιλογή του επόμενου προς κατέβασμα κομματιού, προτιμώνται τα σπανιότερα κομμάτια των ομότιμων με τους οποίους υπάρχει σύνδεση, τεχνική που αναφέρεται ως «σπανιότερο πρώτο». Με την τεχνική αυτή βεβαιώνουμε ότι όλοι οι κόμβοι έχουν κομμάτια επιθυμητά από άλλους κόμβους και επιπλέον ότι τα κομμάτια που είναι πιο κοινά αφήνονται για αργότερα. Έτσι η πιθανότητα ένας κόμβος που αυτήν την περίοδο έχει επιθυμητά κομμάτια αργότερα να μην έχει τίποτα ενδιαφέρον σχεδόν εξαλείφεται.

Η θεωρία πληροφορίας υπαγορεύει ότι κανένας κόμβος δεν μπορεί να ολοκληρώσει έως ότου έχουν φορτωθεί όλα τα κομμάτια του αρχείου από το διανομέα [11]. Σε εφαρμογές με ένα μόνο διανομέα, του οποίου η ικανότητα για κατέβασμα είναι αρκετά μικρότερη από αυτή πολλών κόμβων, η απόδοση είναι πολύ καλύτερη εάν διαφορετικοί κόμβοι μεταφορτώσουν διαφορετικά κομμάτια από το διανομέα. Η τεχνική του σπανιότερου πρώτου είναι αποδοτική μόνο όταν φορτώνονται νέα κομμάτια από το διανομέα, δεδομένου ότι έτσι κάθε κόμβος θα είναι σε θέση να δει ποιοι κόμβοι έχουν τα κομμάτια που ο διανομέας έχει φορτώσει ήδη.

Σε κάποιες περιπτώσεις ο αρχικός διανομέας απομακρύνεται από το σύστημα για λόγους δαπανών, αφήνοντας μόνο τους κόμβους που μεταφορτώνουν. Αυτό ενέχει τον κίνδυνο ένα συγκεκριμένο κομμάτι να μην είναι διαθέσιμου από κανένα τρέχοντα κόμβο. Η τεχνική που εξετάζουμε χειρίζεται καλά αυτή την κατάσταση, με την αντιγραφή των σπανιότερων κομματιών όσο το δυνατόν γρηγορότερα ώστε να μειωθεί ο κίνδυνος οριστικής απώλειάς τους.

#### 3.2.3.4 Τρόπος Endgame

Μερικές φορές ένα κομμάτι ζητείται από έναν κόμβο με πολύ αργά ποσοστά μεταφοράς. Αυτό δεν είναι πρόβλημα στη μέση του κατεβάσματος, αλλά θα μπορούσε ενδεχομένως να καθυστερήσει το κατέβασμα στο τέλος του. Για να αποτραπεί αυτό το πρόβλημα, μόλις ζητηθούν τα κομμάτια που θέλει ένας ομότιμος, αυτός στέλνει αιτήματα για όλα τα κομμάτια σε όλους τους ομότιμους. Στέλνει μηνύματα ακύρωσης για τα κομμάτια που φθάνουν ώστε να μην σπαταλά εύρος ζώνης. Στην πράξη δεν σπαταλάται εύρος ζώνης με αυτόν τον τρόπο, καθώς η χρονική περίοδος που διαρκεί η τεχνική αυτή είναι πολύ μικρή, και το τέλος ενός αρχείου μεταφορτώνεται πάντα γρήγορα.

#### 3.2.3.5 Διανομή μεταξύ ομότιμων

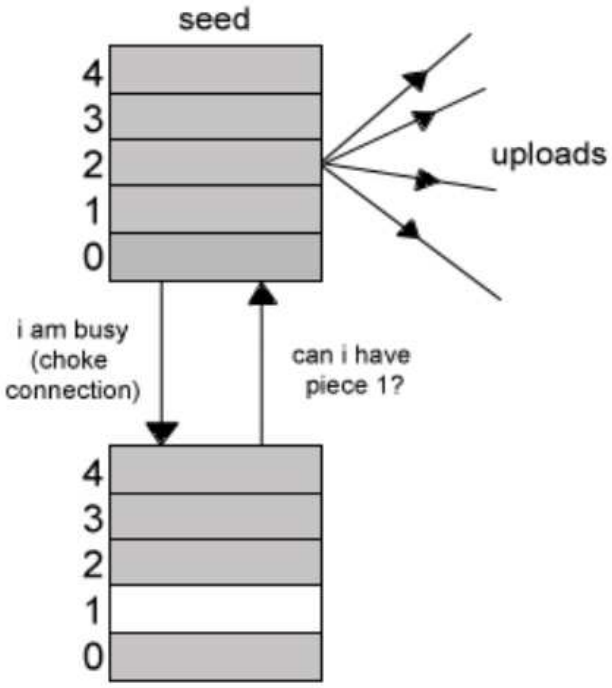
Ο ρόλος του tracker ολοκληρώνεται όταν οι ομότιμοι «βρούν ο ένας τον άλλο». Από την στιγμή αυτή, η επικοινωνία γίνεται απευθείας μεταξύ των ομότιμων, και ο tracker δεν ανακατεύεται. Το σύνολο των ομοτίμων με τους οποίους ένας BitTorrent client βρίσκεται σε επικοινωνία αποκαλείται σμήνος (swarm).

Προκειμένου να ξέρουμε ποιος κόμβος έχει τι, το πρωτόκολλο BitTorrent κόβει τα αρχεία σε κομμάτια σταθερού μεγέθους, συνήθως 250 MB. Κάθε ομότιμος ενημερώνει τους υπόλοιπους ποια κομμάτια έχει. Για τον έλεγχο της ακεραιότητας των κομματιών αυτών χρησιμοποιούνται οι συνόψεις SHA1 καθενός από αυτά, που συμπεριλαμβάνονται στο αρχείο .torrent. Κάθε ομότιμος αναφέρει ένα κομμάτι ως αποκτημένο μόνο αφού έχει βρει τη σύνοψη SHA1 του κομματιού που έλαβε και έχει πιστοποιήσει ότι ισούται με αυτή που περιέχεται στο .torrent αρχείο.

Οι ομότιμοι θα συνεχίσουν να κατεβάζουν δεδομένα από όλους τους διαθέσιμους ομότιμους. Μπορούν επίσης να εμποδίσουν άλλους ομότιμους από το να κατεβάσουν, εάν είναι απαραίτητο. Αυτό είναι γνωστό ως choking [10].

### 3.2.3.6 Choking

Όταν ένας ομότιμος λαμβάνει ένα αίτημα για ένα κομμάτι από έναν άλλο ομότιμο, μπορεί να επιλέξει να αρνηθεί να το μεταφέρει. Εάν αυτό συμβεί, ο ομότιμος λέγεται ότι είναι πνιγμένος (choked). Αυτό γίνεται για διάφορους λόγους, αλλά ο πιο κοινός είναι ότι αρχικά ο ομότιμος θα υποστηρίζει μόνο τον αρχικό αριθμό ταυτόχρονων ανεβασμάτων (max_uploads). Όλα τα επιπλέον αιτήματα προς τον ομότιμο θα χαρακτηρισθούν ως choked. Συνήθως η αρχική τιμή για την μεταβλητή max_uploads είναι 4 [10].




**Σχήμα 3.5** – Choking από έναν ομότιμο [10].

Ο ομότιμος θα παραμείνει choked μέχρι ένα μήνυμα unchoked να σταλλεί.
Ο έλεγχος συμφόρησης TCP συμπεριφέρεται αναξιόπιστα κατά τη διάρκεια πολλών ταυτόχρονων συνδέσεων [14]. Επίσης, το choking επιτρέπει σε κάθε ομότιμο να χρησιμοποιεί τον αλγόριθμο tit-for-tat για να εξασφαλισθεί ότι θα υπάρχει ένα αξιόλογο download rate. Κατά τον αλγόριθμο tit-for-tat [14], οι χρήστες τείνουν να προτιμούν και να συνεισφέρουν σε αυτούς που τους είχαν “βοηθήσει”.

Υπάρχουν διάφορα κριτήρια που ένας καλός choking αλγόριθμος πρέπει να ικανοποιεί [14]:

- πρέπει να θέτει όρια στον αριθμό των ταυτόχρονων uploads, για καλύτερη απόδοση του TCP.
- πρέπει να αποφεύγει τη γρήγορη εναλλαγή choking-unchoking, γνωστό ως ‘fibrillation’.
- πρέπει να αποκρίνεται στους ομότιμους που τους επιτρέπεται να κατεβάζουν.
- τέλος, πρέπει να δοκιμάζει τις αχρησιμοποίητες συνδέσεις κατά διαστήματα, για να ανακαλύψει αν είναι καλύτερες από αυτές που ήδη χρησιμοποιεί, διαδικασία γνωστή ως “optimistic unchoking”.

#### 3.2.3.7 Optimistic unchoking

Το “optimistic unchoking” είναι ένας μηχανισμός του Bittorrent για την ανακάλυψη “καλύτερων ομοτίμων”, δηλαδή αυτών που θα προσφέρουν περισσότερο upload. Οι ομότιμοι κάνουν upload μόνο σε ένα υποσύνολο των ομοτίμων που συνδέονται, οι οποίοι αποκαλούνται προτιμημένοι ομότιμοι (preferred peers) [14]. Ένας ομότιμος επιλέγει έναν άλλο ομότιμο όχι απαραίτητα μεταξύ των preferred peers και κάνει upload σε αυτόν, με την ελπίδα ότι ο ομότιμος αυτός θα ανταποκριθεί. Εάν αυτός προφέρει καλύτερο upload rate από οποιονδήποτε από τους preferred peers, τότε θα γίνει προτιμημένος ομότιμος και θα μετατοπίσει τον πιό αργό προτιμημένο ομότιμο. Αυτό εξασφαλίζει ότι ένας ομότιμος προχωρεί πάντα προς την καλύτερη χρησιμοποίηση του εύρους ζώνης. Ο ομότιμος που καθόριζεται σε αυτό εναλλάσεται κάθε 30 δευτερόλεπτα. Αυτός είναι αρκετός χρόνος για τα upload/download rates να φτάσουν στη μέγιστη δυνατότητά τους. Δηλαδή, το πρωτόκολλο δοκιμάζει τυχαία ένα νέο γείτονα κάθε 30 sec. Με τον τρόπο αυτό, δίνεται μια ευκαιρία στο νέο γείτονα για να συμμετάσχει.

#### 3.2.3.8 Anti-snubbing

Υπάρχει περίπτωση κάποια στιγμή ένας ομότιμος στο BitTorrent να γίνει choked από όλους τους ομότιμους, από τους οποίους έκανε download στο παρελθόν. Σε τέτοιες περιπτώσεις θα συνεχίσει συνήθως να έχει χαμηλό download rate μέχρις ότου, μέσω του optimistic unchoking, βρεί καλύτερους ομότιμους. Για να μετριαστεί αυτό το πρόβλημα, όταν περάσει ένα λεπτό χωρίς να πάρει κανένα κομμάτι από έναν συγκεκριμένο ομότιμο, το BitTorrent υποθέτει ότι είναι “snubbed” από εκείνο τον ομότιμο και δεν κάνει upload σε αυτόν εκτός αν επιλεχθεί από το optimistic unchoking [15].

#### 3.2.3.9 Επικοινωνία μεταξύ ομότιμων

Οι ομότιμοι που ανταλλάσσουν δεδομένα βρίσκονται σε συνεχή επικοινωνία. Οι συνδέσεις είναι συμμετρικές, και επομένως τα μηνύματα μπορούν να σταλλούν και στις δύο κατευθύνσεις. Αυτά τα μηνύματα αποτελούνται από μια χειραψία (handshake), που ακολουθούνται από μια ατέρμονη ροή length-prefixed μηνυμάτων.

#### 3.2.3.10 PWR: Peer Wire Protocol

Το peer wire πρωτόκολλο είναι το έταιρο πρωτόκολλο που περιγράφει τις αλληλεπιδράσεις μεταξύ των clients. Τρέχει πάνω από το πρωτόκολλο TCP. Το μήνυμα που περνάει είναι συμμετρικό, δηλαδή τα μηνύματα που στέλνονται και στις δύο κατευθύνσεις είναι τα ίδια. Όταν ένας client θέλει να αρχίσει μια σύνδεση, ετοιμάζει τη σύνδεση TCP και στέλνει ένα μήνυμα χειραψίας (handshake message) στον άλλο ομότιμο. Εάν το μήνυμα είναι αποδεκτό, η λαμβάνουσα πλευρά στέλνει πίσω ένα μήνυμα χειραψίας. Εάν ο αρχικός ομότιμος δεχτεί, το μήνυμα μπορεί να αρχίσει, και συνεχίζεται αόριστα. Όλοι οι ακέραιοι αριθμοί κωδικοποιούνται ως ψηφιολέξη τεσσάρων byte big-endian, εκτός από το πρώτο πρόθεμα στη χειραψία [10].

- ✓ **Handshake message**

Το μύνημα χειραψίας αποτελείται από πέντε μέρη [10]:

- Ένα μόνο byte, που περιέχει τη δεκαδική τιμή 19. Αυτό είναι το μήκος του αλφαριθμητικού χαρακτήρων που ακολουθεί αυτό το byte.

- Το αλφαριθμητικό χαρακτήρων “BitTorrent protocol”, που περιγράφει το πρωτόκολλο. Τα νεώτερα πρωτόκολλα πρέπει να ακολουθούν αυτήν την σύμβαση για να διευκολύνουν τον εύκολο προσδιορισμό των πρωτοκόλλων.
- Οκτώ δεσμευμένα bytes για την περαιτέρω επέκταση του πρωτοκόλλου. Όλα τα bytes είναι μηδέν στις τρέχουσες υλοποιήσεις.
- Μια 20-byte SHA1 hash της τιμής απεικόνισης του κλειδιού info στο αρχείου torrent. Αυτή είναι η ίδια hash που στέλνεται στον tracker στη μεταβλητή info_hash.
- Το 20-byte αλφαριθμητικό χαρακτήρων που αναπαριστά την peer id. Αυτή είναι η ίδια τιμή που στέλνεται στον tracker.

Εάν ένας ομότιμος είναι ο πρώτος παραλήπτης σε μια χειραψία, και το info_hash δεν ταιριάζει με κανένα torrent που εξυπηρετεί, η σύνδεση πρέπει να κλείσει. Εάν ο ομότιμος που ξεκίνησε τη σύνδεση λάβει μια χειραψία όπου το peer id δεν ταιριάζει με την id που λαμβάνεται από τον tracker, η σύνδεση πρέπει να κλείσει. Κάθε ομότιμος πρέπει να διατηρεί την κατάσταση κάθε σύνδεσης. Η κατάσταση αποτελείται από δύο τιμές, interested και choking. Ένας ομότιμος μπορεί είναι είτε interested είτε not interested για έναν άλλο ομότιμο, και είτε choke είτε not choke για τον άλλο ομότιμο. Το choking σημαίνει ότι κανένα αίτημα δεν θα απαντηθεί, και interested σημαίνει ότι ο ομότιμος ενδιαφέρεται για να κατεβάσει κομμάτια του αρχείου από τον άλλο ομότιμο.

Αυτό σημαίνει ότι κάθε ομότιμος χρειάζεται τέσσερις λογικές (Boolean) τιμές για να γνωρίζει τη κατάσταση της σύνδεσης.

- am_interested
- am_choking
- peer_interested
- peer_choking

Όλες οι συνδέσεις αρχίζουν ως not interested και choking και για τους δύο ομότιμους. Οι χρήστες πρέπει να διατηρούν τη τιμή am_interested συνεχώς ενημερωμένη, και να αναφέρουν τυχόν αλλαγές στον άλλο ομότιμο. Τα μηνύματα που στέλνονται μετά τη χειραψία είναι δομημένα ως εξής:

[μήκος μηνύματος σε ακέραιο αριθμό] [ένα byte που περιγράφει τον τύπο μηνύματος] [ωφέλιμο φορτίο]

- ✓ **Handshaking**

Η χειραψία πραγματοποιείται ως εξής [10]:

- Η χειραψία αρχίζει με το δεκαδικό χαρακτήρα 19 ακολουθούμενο από το αλφαριθμητικό 'BitTorrent Protocol'.
- Μια 20-byte SHA1 hash του bencoded κλειδιού info από το αρχείο metainfo στέλνεται στη συνέχεια. Αν αυτή δεν ταιριάζει μεταξύ των ομοτίμων, η σύνδεση κλείνει.
- Μια 20-byte peer id στέλνεται, η οποία στη συνέχεια χρησιμοποιείται σε αιτήματα του tracker και περιλαμβάνεται στα αιτήματα του ομότιμου. Εάν η id του ομότιμου δεν ταιριάζει με αυτή που αναμενόταν, η σύνδεση κλείνει.

#### 3.2.3.11 Message stream

Αυτή η σταθερή ροή των μηνυμάτων επιτρέπει σε όλους τους ομότιμους στο σμήνος (swarm) να στέλνουν δεδομένα, και να ελέγχουν τις αλληλεπιδράσεις με άλλους ομοτίμους. Ένας ομότιμος θα «ενδιαφερθεί» για δεδομένα εάν υπάρχει ένας ομότιμος που έχει τα απαραίτητα κομμάτια. Εάν ο ομότιμος που έχει αυτά τα δεδομένα δεν είναι choked, τότε αυτά θα μεταφερθούν. Μετά από τη χειραψία, εξ ορισμού, οι συνδέσεις ξεκινάνε σαν choked, και όχι ιδιαίτερου ενδιαφέροντος.

### 3.2.4 Data

Το πρωτόκολλο BitTorrent είναι πολύ ευπροσάρμοστο, και μπορεί να χρησιμοποιηθεί για να μεταφέρει ένα αρχείο, αποτελούμενο από πολλαπλά αρχεία οποιουδήποτε τύπου, που περιλαμβάνονται σε οποιοδήποτε αριθμό καταλόγων. Τα μεγέθη αρχείων μπορούν να ποικίλουν σημαντικά, από μερικά kilobytes εως εκατοντάδες gigabytes [10].

#### 3.2.4.1 Μέγεθος κομματιού

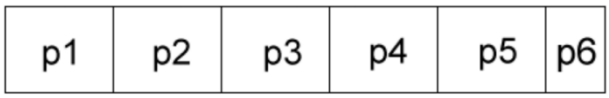
Τα δεδομένα χωρίζονται σε μικρότερα κομμάτια που στέλνονται μεταξύ των ομότιμων χρησιμοποιώντας το πρωτόκολλο BitTorrent. Αυτά τα κομμάτια έχουν συγκεκριμένο μέγεθος [10], γεγονός που επιτρέπει στον tracker να κρατά ετικέτες για

το ποιος έχει κάθε κομμάτι κάποιου δεδομένου. Αυτό επίσης κατατμίζει το αρχείο σε επαληθεύσιμα κομμάτια, έτσι ώστε κάθε κομμάτι να αποκτήσει έναν hash κωδικό, που μπορεί να ελεγθεί από τον ομότιμο που το κατεβάζει για την ακεραιότητα του. Αυτές οι hashes αποθηκεύονται σαν τμήμα του αρχείου metainfo που συζητήσαμε σε προηγούμενη παράγραφο.

Το μέγεθος των κομματιών παραμένει σταθερό σε όλα τα αρχεία στο torrent εκτός από το τελικό κομμάτι. Το μέγεθος του κομματιού εξαρτάται από το ποσό των δεδομένων. Τα μεγέθη κομματιού που είναι πάρα πολύ μεγάλα θα προκαλέσουν την ανεπάρκεια κατά το downloading (μεγαλύτερος κίνδυνος φθοράς δεδομένων στα μεγαλύτερα κομμάτια, λόγω λιγότερων ελέγχων ακεραιότητας), ενώ εάν τα μεγέθη κομματιού είναι πάρα πολύ μικρά, θα υπάρχουν περισσότεροι hash έλεγχοι .

Δεδομένου ότι ο αριθμός κομματιών αυξάνεται, περισσότεροι κωδικοί hash πρέπει να αποθηκευτούν στο αρχείο metainfo. Επομένως, εμπειρικά, τα κομμάτια πρέπει να επιλεχτούν έτσι ώστε το αρχείο metainfo δεν θα είναι μεγαλύτερο από 50 - 75kb. Ο κύριος λόγος για αυτό είναι να περιοριστεί το ποσό φιλοξενίας της αποθήκευσης και του εύρους ζώνης που απαιτούνται με την εύρεση (indexing) των κεντρικών υπολογιστών.

Τα πιό κοινά μεγέθη των κομματιών είναι 256kb, 512kb και 1Mb. Ο αριθμός κομματιών είναι επομένως: συνολικά μήκος/μέγεθος κομματιού. Τα κομμάτια μπορούν να επικαλύψουν τα όρια αρχείων. Για παράδειγμα, ένα αρχείο 1.4 Mb, μπορεί να χωριστεί σε κομμάτια των 5*256kb, και σε ένα τελικό κομμάτι των 120 kb, όπως απεικονίζεται στο Σχήμα 3.6.




**Σχήμα 3.6** – Κομμάτια ενός αρχείου [10].

### 3.2.5 BitTorrent Clients

Ένας BitTorrent client είναι το πρόγραμμα που εφαρμόζει το πρωτόκολλο BitTorrent. Χρησιμοποιείται για να συνδεθούμε με τους άλλους χρήστες και να διαχειριστούμε τα torrents. Το πρόγραμμα εγκαθιστάται στο λειτουργικό σύστημα και είναι αρμόδιο για τον έλεγχο της ανάγνωσης/γραφής των αρχείων, άνοιγμα των sockets κ.λπ.

Ένα αρχείο metainfo ανοίγεται από το πρόγραμμα για την έναρξη συμμετοχής σε ένα torrent. Μόλις διαβαστεί το αρχείο, εξάγονται τα απαραίτητα δεδομένα, και μια socket ανοίγεται για την επικοινωνία με τον tracker. Ο BitTorrent client χρησιμοποιεί μια πόρτα για το σκοπό αυτό. Το άνοιγμα κάποιου άλλου BitTorrent client θα χρησιμοποιήσει κάποια άλλη πόρτα. Το πρόγραμμα αυτό μπορεί να χειριστεί πολλά ενεργά torrents ταυτόχρονα.

Υπάρχει μία ποικιλία από BitTorrent client και μπορούν να κυμανθούν από πρόγράμματα με βασικές εφαρμογές και με λίγα χαρακτηριστικά γνωρίσματα ως πολύ προηγμένα και εξατομικεύσιμα. Παραδείγματος χάριν, μερικά προηγμένα χαρακτηριστικά γνωρίσματα είναι τα metainfo file wizards και ενσωματωμένοι trackers. Αυτά τα πρόσθετα χαρακτηριστικά γνωρισμάτα έχουν σαν αποτέλεσμα τα προγράμματα αυτά να έχουν διαφορετική συμπεριφορά το καθένα, και να μπορούν να χρησιμοποιήσουν πολλές διαφορετικές πόρτες, ανάλογα με τον αριθμό των διαδικασιών που τρέχουν. Δεδομένου ότι όλες οι εφαρμογές υιοθετούν το ίδιο πρωτόκολλο, δεν υπάρχει κανένα ζήτημα ασυμβιβατότητας. Υπάρχουν γύρω στους 50 BitTorrent clients. Μερικά από τα προγράμματα αυτά είναι [16]:

- ✓ BitTorrent
- ✓ Azureus
- ✓ uTorrent
- ✓ Bitcomet
- ✓ FlashGet
- ✓ Folx
- ✓ OneSwarm

Κατά κανόνα τα προγράμματα αυτά εφαρμόζουν το πρωτόκολλο TCP, για τη κίνηση των δεδομένων ανάμεσα στους ομότιμους, ωστόσο υπάρχουν κάποιες εξαιρέσεις , όπως το uTorrent που μπορεί να χρησιμοποιήσει και το UDP πρωτόκολλο.

Στο Παράρτημα Α’ παρουσιάζεται μια σύντομη περιγραφή του προγράμματος BitTorrent.

# Κεφάλαιο 4

# Gnutella

## 4.1 Ιστορία

Μετά την πρώτη εμφάνιση του Napster η AOL [17] ανέθεσε σε δύο προγραμματιστές της να φτιάξουν κάτι παρόμοιο. Οι Justin Frankel και Tom Pepper που δούλευαν στη Nullsoft [18] (η εταιρεία που δημιούργησε το Winamp [19] και εξαγοράστηκε από την AOL), έγραψαν ένα καθαρά peer to peer πρωτόκολλο το οποίο και ανακοίνωσαν πως θα διέθεταν δωρεάν στο Διαδίκτυο. Μόλις εκδόθηκε η πρώτη έκδοση του προγράμματος στις 14 Μαρτίου του 2000, η AOL φοβισμένη από τις μηνύσεις κατά του Napster αποφάσισε να το αποσύρει. Ήταν όμως ήδη αργά καθώς ικανοί προγραμματιστές που είχαν αποκτήσει το αρχείο κατάφεραν να το δημιουργήσουν από την αρχή και να το προσφέρουν στον κόσμο, αυτή τη φορά με ανοιχτό τον κώδικα. Το όνομά του: Gnutella.

Η ονομασία προήλθε από ένωση του GNU [20] και της Nutella καθώς οι δημιουργοί της Nullsoft θα το προσέφεραν με την GNU άδεια (κάτι το οποίο δεν πρόλαβε να γίνει) και επίσης έτρωγαν πολύ Nutella όσο το έγραφαν [21]. Αυτό που πετυχαίνει διαφορετικό από το Napster και το OpenNap [22] είναι ότι δε χρησιμοποιεί κεντρικούς servers αλλά δρα αποκλειστικά peer to peer. Με τον τρόπο αυτό, κάθε υπολογιστής παίζει το διπλό ρόλο του να αναζητεί το δίκτυο αλλά και του να βοηθάει άλλους να συνδεθούν σε αυτό και να το εξερευνήσουν.

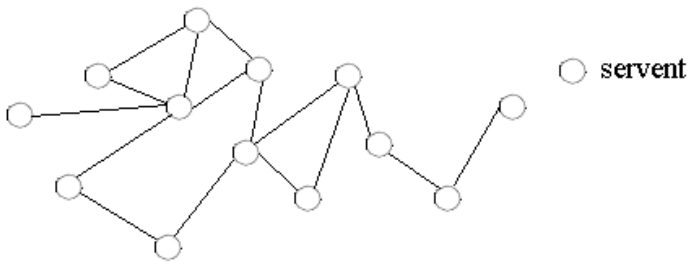
Γι' αυτό το λόγο το δίκτυο Gnutella δε μπορεί να κλείσει έτσι απλά καθώς δεν υπάρχει κεντρική διαχείριση και ακόμη και αν αποσυνδεθούν κάποιοι χρήστες, πάντα θα υπάρχουν κάποιοι άλλοι να πάρουν τη θέση τους. Με τη χρήση του γνωστού LimeWire [23], του Morpheus [24] όπως και του Mldonkey [25], το δίκτυο αυτό έγινε το εναλλακτικό του OpenNap. Παρόλα αυτά, εξ αιτίας της κατασκευής του ήταν ιδιαίτερα αργό. Το Gnutella αυτή τη στιγμή έχει τους περισσότερους clients και είναι ένα από τα μεγαλύτερα δίκτυα στο Internet.

## 4.2 Αρχιτεκτονική

Το Gnutella αποτελείται από ένα σύνολο δυναμικά συνδεδεμένων κόμβων, το οποίο δημιουργεί ένα αδόμητο overlay δίκτυο. Μέσω αυτού πραγματοποιείται η κυκλοφορία στο δίκτυο, που αποτελείται από ερωτήματα, απαντήσεις στα ερωτήματα, και επίσης άλλα μηνύματα ελέγχου για να διευκολύνει την ανίχνευση άλλων κόμβων. Η δυναμική συμπεριφορά του δικτύου έγκειται στο γεγονός ότι οι κόμβοι μπορούν να εισέρχονται και να εξέρχονται στο δίκτυο ανά πάσα στιγμή.

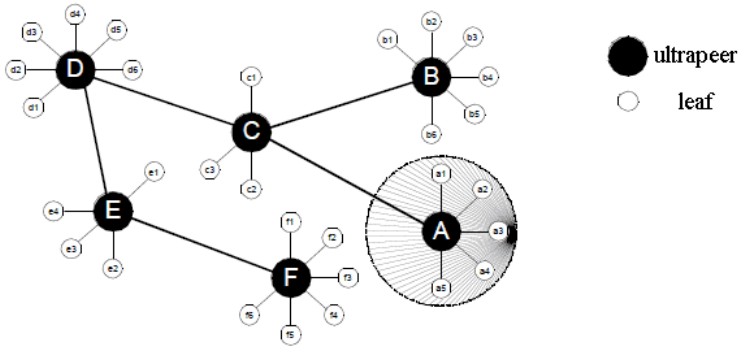
Αρχικά, το δίκτυο Gnutella (έκδοση v0.4) ήταν ένα πλήρως αποκεντρικοποιημένο και αδόμητο P2P δίκτυο. Η τοπολογία του απεικονίζεται στο Σχήμα 4.1. Δεν υπάρχει κάποια κεντρική οντότητα και όλοι οι κόμβοι είναι απολύτως ίσοι μεταξύ τους. Στο δίκτυο αυτό οι ομότιμοι μπορούν να ενεργούν ταυτόχρονα ως servers, παρέχοντας στοιχεία και υπηρεσίες στους υπόλοιπους ομότιμους, και ως clients χρησιμοποιώντας αυτά τα στοιχεία και τις υπηρεσίες. Για το λόγο αυτό οι ομότιμοι στην τοπολογία αυτή ονομάστηκαν servents.

Οι χρήστες αλληλεπιδρούν με τους κόμβους παρεχοντάς τους μία λίστα με τους πόρους που διαθέτουν και επιθυμούν να ανταλλάξουν στο δίκτυο. Μπορούν να εισάγουν αναζητήσεις για άλλους πόρους και να λάβουν ενδεχομένως κάποια αποτελέσματα, και στη συνέχεια να επιλέξουν και να αποθηκεύσουν κάποιους πόρους μεταξύ των αποτελεσμάτων. Το δίκτυο Gnutella χρησιμοποιείται μόνο για να εντοπίσει τους κόμβους που θα μοιραστούν τα δεδομένα [26]. Οι ανταλλαγές δεδομένων γίνονται άμεσα μεταξύ των κόμβων με τη χρήση του standard HTTP πρωτοκόλλου.




**Σχήμα 4.1** – Τοπολογία δικτύου Gnutella v0.4.

Στην συνέχεια το Gnutella με την έκδοση v0.6 απέκτησε μια υβριδική μορφή με την παρουσία των ultrapeers. Η νέα τοπολογία του απεικονίζεται στο Σχήμα 4.2. . Λεπτομέρειες και συγκριτικά στοιχεία θα παρουσιαστούν σε ακόλουθη παράγραφο.




**Σχήμα 4.2** – Τοπολογία δικτύου Gnutella v0.6.

### 4.2.1 Ομότιμοι

Gnutella client είναι ένα πρόγραμμα που υλοποιεί το πρωτόκολλο Gnutella και που χρησιμοποιεί όποιος θέλει να συμμετέχει στο ομόνυμο δίκτυο. Οι εφαρμογές αυτές ονομάζονται gnodes (ή εναλλακτικά servents). Παρέχουν διεπαφές μέσω των οποίων οι χρήστες μπορούν να θέτουν ερωτήματα και να βλέπουν τα αποτελέσματα τους, να δέχονται ερωτήματα από άλλους servents, να ελέγχουν την δυνατότητα απάντησης, και αν υπάρχει, να στέλνουν τα αποτελέσματα. Οι κόμβοι είναι υπεύθυνοι για την διαχείριση της κίνησης που υπάρχει.

Υπάρχουν πολλοί clients που υποστηρίζουν το βασικό πρωτόκολλο Gnutella ή μετέπειτα εκδόσεις του. Δεν υπάρχει όμως τυποποιημένο λογισμικό Gnutella client. Οι clients μπορούν να επικοινωνούν μεταξύ τους. Όμως οι σχεδιαστές ανάπτυξης των προγραμμάτων είναι ελεύθεροι να εφαρμόσουν τη λειτουργικότητα και τυχόν επεκτάσεις στο πρόγραμμα όπως το επιθυμούν. Τόσο οι προδιαγραφές του Gnutella όσο και οι περισσότεροι clients είναι open source (GNU General Public License) και έχουν αναπτυχθεί για κάθε είδους πλατφόρμας (Macintosh, Windows, Linux – Unix).

Ένας νέος κόμβος για να συνδεθεί στο δίκτυο Gnutella πρέπει πρώτα να συνδεθεί με έναν τουλάχιστο κόμβο ο οποίος είναι διαθέσιμος. Αυτό επιτυγχάνεται μέσω του Gnutella client. Όποτε αρχίζει να εκτελείται ένας Gnutella client, δεν γνωρίζει ποιοι άλλοι κόμβοι είναι εκείνη τη στιγμή διαθέσιμοι στο δίκτυο. Το πρώτο βήμα συνεπώς είναι να βρεί τουλάχιστον ένα ενεργό κόμβο. Αυτή η διαδικασία καλείται Bootstrapping [27]. Μέχρι να ολοκληρωθεί αυτή η διαδικασία, ο ομότιμος δεν μπορεί να συμμετέχει σε δραστηριότητες ανταλλαγής αρχείων. Το Gnutella στην αρχική του μορφή (έκδοση v0.4) δεν προέβλεπε κεντρικό server για να βοηθήσει στο σκοπό αυτό. Εάν ένας client εκτελούσε μία δοκιμαστική διαδικασία, π.χ. στέλνοντας

μηνύματα σε όλους τους υπολογιστές, άσχετα αν είναι στο δίκτυο Gnutella ή όχι, τότε αυτό θα οδηγούσε σε μια τεράστια σπατάλη των πόρων.

Το Bootstrap μπορεί να γίνει χρησιμοποιώντας διάφορες μεθόδους. Ωστόσο, υπάρχουν δύο βασικοί αλγόριθμοι που έχουν καθιερωθεί για το το σκοπό αυτό [27]. Στο πρώτο αλγόριθμο, ένας client αποθηκεύει και διατηρεί μία λίστα στο σκληρό του δίσκο από IPs με όλους τους γνωστούς κόμβους από τη τελευταία φορά που συνδέθηκε στο Gnutella. Στην επόμενη σύνδεση στέλνει το κατάλληλο μήνυμα σε αυτές τις IPs. Δεδομένου ότι τα εργαλεία για την ανταλλαγή αρχείων, συχνά εκτελούνται στο background και οι σύγχρονοι υπολογιστές σπάνια είναι εκτός λειτουργίας, υπάρχει μεγάλη πιθανότητα να βρεθεί ένας έγκυρος κόμβος. Επιπλέον, εάν τυχαία ο υπολογιστής συνδεθεί μέσω μιας IP που έχει αλλαξεί, γεγονός συνηθισμένο στις dial-up συνδέσεις, αλλά ο νέος υπολογιστής τρέχει επίσης το πρόγραμμα Gnutella, τότε το Bootstrapping θα λειτουργήσει κανονικά, εφόσον όλοι οι Gnutella clients συμπεριφέρονται το ίδιο.

Ο δεύτερος αλγόριθμος είναι πιό αξιόπιστος και λειτουργεί με“web caches”, το σύστημα GwebCache [47]. Είναι συνηθισμένες ιστοσελίδες που τροφοδοτούνται από εξειδικευμένα PHP, ASP ή Perl scripts. Μόνη τους λειτουργία είναι να επιστρέφουν κάποιες τυχαίες έγκυρες IPs των clients που καταχωρηθήσαν πρόσφατα. Οι ιστοσελίδες ενημερώνονται καθημερινά επομένως υπάρχει επικοινωνία μεταξύ των σελίδων, καθώς επίσης παρέχουν και URLs από άλλες σελίδες. Ένας «καλός» Gnutella client θα πρέπει να αποθηκεύει όλες τις web caches με τον ίδιο τρόπο που κάνει για τις IPs. Βασικά, το σύστημα GwebCache είναι ένα δεύτερο δίκτυο τοποθετημένο πάνω από το δίκτυο Gnutella. Είναι αρκετά γενικό σύστημα αφού απλά γνωρίζει κάποιες IPs και URLs, καθώς και σχετικά εύκολο στη οργάνωση.

Εφόσον ο αρχικός ομότιμος βρεί έναν ενεργό κόμβο, ξεκινά μία σύνδεση μεταξύ τους, με σκοπό να ληφθεί η IP διεύθυνση του ενεργού κόμβου. Μόλις ληφθεί η IP διεύθυνση, δημιουργείται μια σύνδεση TCP/IP με βάση το πρωτόκολλο χειραψίας Gnutella. Το πρωτόκολλο αυτό είναι διαφορετικό στις εκδόσεις v0.4 και v0.6. Οποιαδήποτε άλλη απάντηση σημαίνει ότι ο servent δεν είναι πρόθυμος να δεχτεί τη σύνδεση. Οι συνδέσεις μπορούν να απορριφθούν, π.χ. επειδή οι εκδόσεις του πρωτοκόλου δεν είναι συμβατές ή επειδή εκείνος ο συγκεκριμένος servent έχει ήδη πάρα πολλές συνδέσεις.

Από την στιγμή που γίνει η σύνδεση με ένα τουλάχιστον κόμβο στο δικτύο Gnutella, γίνεται αίτηση για μια λίστα από τους κόμβους που ο ομότιμος είναι

συνδεμένος. Παρόμοιο αίτημα γίνεται και στους κόμβους από τη λίστα κοκ (αλγόριθμος “Friend of a Friend”) [27]. Έτσι, μέσω του προγράμματος γίνονται προσπάθειες μέσω διαδοχικών συνδέσεων TCP με τους ομότιμους που προκύπτουν από τις λίστες, μέχρι όμως μιας ορισμένης ποσότητας. Στη προσπάθειά του να συνδεθεί με τους κόμβους αυτούς, ο ομότιμος αποθηκεύει τοπικά τις διευθύνσεις των κόμβων που δεν έχει δοκιμάσει ακόμα να συνδεθεί, και απορρίπτει αυτές που δοκίμασε αλλά ήταν άκυρες .

## 4.3 Πρωτόκολλο Gnutella

Έχοντας μία ή περισσότερες ανοιχτές συνδέσεις με κόμβους συνδεδεμένους στο Gnutella, οι κόμβοι στέλνουν μηνύματα για να επικοινωνήσουν μεταξύ τους. Τα μηνύματα στέλνονται (broadcast) σε όλους τους κόμβους με τους οποίους ο αποστολέας έχει ανοιχτές συνδέσεις και έπειτα προωθούνται προς τα πίσω (back-propagation). Στέλνονται σε μία συγκεκριμένη σύνδεση στην αντίθετη κατεύθυνση από την διαδρομή που ακολούθησε ένα αρχικό broadcasted μήνυμα.

Τα χαρακτηριστικά γνωρίσματα του πρωτοκόλλου που διευκολύνουν τον μηχανισμό broadcast/back-propagation είναι [28]:

- κάθε μήνυμα έχει ένα μοναδικό αριθμό σαν αναγνωριστικό (Globally Unique Identifier - GUID).
- κάθε κόμβος κρατάει στην μνήμη του τα πιο πρόσφατα μηνύματα που δρομολόγησε και χρησιμοποιείται για να αποτρέψει το re-broadcasting και να κάνει εφικτή την υλοποίηση του back-propagation.
- τα μηνύματα έχουν ένα πεδίο Time-to-live (TTL) και ένα πεδίο "Hops taken".

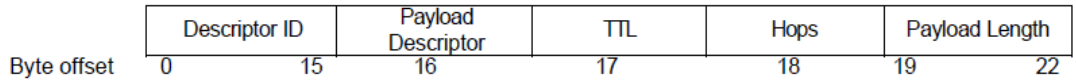
Το πρωτόκολλο Gnutella καθορίζει τον τρόπο με τον οποίο οι servents επικοινωνούν στο δίκτυο. Αποτελείται από ένα σύνολο μηνυμάτων, που χρησιμοποιούνται για τη διαβίβαση δεδομένων μεταξύ των servents, και ένα σύνολο κανόνων για την ανταλλαγή των περιγραφητών μεταξύ των ομοτίμων. Οι περιγραφητές του πρωτοκόλλου, και ουσιαστικά οι τύποι των μηνυμάτων που επιτρέπονται στο δίκτυο είναι:

- ✓ PING,
- ✓ PONG,
- ✓ QUERY,

- ✓ QUERYHIT,
- ✓ PUSH και
- ✓ GET.

### 4.3.1 Δομή πακέτου

Αρχικά, πρίν από τους περιγραφητές, υπάρχει μια επικεφαλίδα περιγραφητή (descriptor header) η οποία ορίζει τον εκάστωτε περιγραφητή πρωτοκόλλου. Κάθε πακέτο Gnutella και έχει τη δομή που απεικονίζεται στο Σχήμα 4.3. Τα επιμέρους πεδία είναι [29]:




**Σχήμα 4.3** – Επικεφαλίδα περιγραφητή [29].

- **Descriptor ID** - ένα 16-byte αλφαριθμητικό που προσδιορίζει τον περιγραφητή στο δίκτυο. Είναι μοναδικό για κάθε μήνυμα. Κατά κανόνα, η λειτουργία της διεύθυνσης του αποστολέα κόμβου χρησιμοποιείται στα μηνύματα Pong και QueryHit ως προσδιοριστικό του προορισμού.
- **Payload Descriptor** - προσδιορίζει το τύπο του μυνήματος. Ανάλογα με την τιμή, αντιστοιχεί και το ανάλογο μήνυμα, έτσι έχουμε:

*0x00 = Ping*
*0x01 = Pong*
*0x40 = Push*
*0x80 = Query*
*0x81 = QueryHit*

- **TTL (Time to live)** - αριθμός που δείχνει πόσες φορές το μήνυμα θα διαβιβαστεί από τους Gnutella servents προτού να απομακρυνθεί από το δίκτυο. Συνήθως η αρχική τιμή για το TTL είναι 7. Κάθε servent ελλατώνει κατά ένα τη τιμή του TTL, πρίν το στείλει προς έναν άλλο servent. Όταν το TTL φθάσει το 0, το μήνυμα δεν θα προωθηθεί περαιτέρω. Το TTL είναι ο μόνος τρόπος να περιοριστούν τα μήνυμα στο δίκτυο και εάν δεν οριστεί εξ αρχής ή παραποιηθεί, το πιθανό είναι να έχει σαν αποτέλεσμα, την υψηλή κυκλοφορία στο δίκτυο και εν τέλει τη κακή απόδοση του δικτύου.
- **Hops** - ο αριθμός που δείχνει πόσες φορές έχει προωθηθεί το συγκεκριμένο μύνημα. Το TTL εξ ορισμού έχει τη τιμή 7. Καθώς το μήνυμα μεταβιβάζεται

από servent σε servent, τα πεδία TTL και Hops της κεφαλίδας πρέπει ικανοποιούν την συνθήκη:

$$TTL(0) = TTL(i) + Hops(i) \qquad (\sigma\chi.4.1)$$

όπου $TTL(i)$ και $Hops(i)$ είναι η τιμή των πεδίων TTL και Hops στη κεφαλίδα, στο i-στο άλμα του μηνύματος, για το $i \geq 0$.

- **Payload length** - είναι το μήκος του μηνύματος. Δεν πρέπει να είναι παραπάνω από 4kb. Ακολουθεί αμέσως μετά τη επικεφαλίδα του μηνύματος. Ο επόμενος περιγραφητής επικεφαλίδας είναι τοποθετημένος τόσα bytes όσο είναι το Payload_Length, από το τέλος αυτής της επικεφαλίδας. Δεν υπάρχει κενό στη ροή δεδομένων του Gnutella. Το πεδίο Payload Length είναι ο μόνος τρόπος για έναν servent να βρεί την αρχή του επόμενου μηνύματος στην εισερχόμενη ροή δεδομένων. Οι servents πρέπει αυστηρά να εγκρίνουν το πεδίο Payload Length για κάθε περιγραφητή που λαμβάνουν, τουλάχιστον για τους καθορισμένου μήκους περιγραφητές. Το πεδίο αυτό πρέπει να ελέγχεται έτσι ώστε ο servent να παραμένει συγχρονισμένος με τη ροή εισόδου του. Αν ο servent είναι εκτός συγχρονισμού με τη ροή εισόδου του, τότε η σύνδεση πέφτει. Ο μόνος τρόπος για την αξιόπιστη ανάγνωση της ροής δεδομένων του δικτύου, που δημιουργείται από τα μηνύματα, είναι εξετάζοντας τη κάθε επικεφαλίδα του Payload Length για να βρεθεί η αρχή του επόμενου περιγραφητή.

Αμέσως μετά την επικεφαλίδα του μηνύματος, ακολουθεί η ωφέλιμη πληροφορία του μηνύματος, που αποτελείται από τους περιγραφητές του πρωτοκόλλου:

- ✓ PING,
- ✓ PONG,
- ✓ QUERY,
- ✓ QUERYHIT,
- ✓ PUSH και
- ✓ GET.

Στην συνέχεια παρουσιάζονται αναλυτικά.

#### 4.3.1.1 PING

Αφού εγκατασταθεί η σύνδεση TCP, ο συνδεδεμένος στο δίκτυο κόμβος στέλνει στους γειτονές του ένα μήνυμα PING με μετρητή βημάτων. Με τον τρόπο αυτό δηλώνει την παρουσία του στο δίκτυο. Οι γειτονικοί κόμβοι προωθούν το μύνημα (flooded - broadcasted) στους δικούς τους γείτονες κοκ μέχρι να εξαντληθεί ο μετρητής βημάτων.

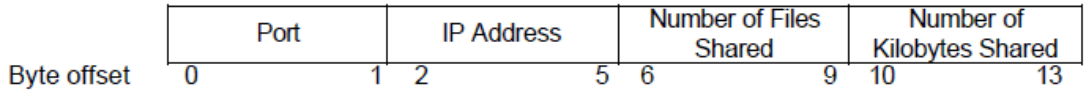
Οι κόμβοι στο Gnutella είναι δυναμικοί, δηλαδή «έρχονται και φεύγουν» συχνά με αποτέλεσμα οι συνδέσεις δικτύων να είναι αναξιόπιστες. Για να αντιμετωπίσει αυτό το περιβάλλον, μετά από την ένταξη του στο δίκτυο, κάθε κόμβος στέλνει περιοδικά PING μηνύματα στους γείτονες του για να ανακαλύψει άλλους συμμετέχοντες κόμβους. Χρησιμοποιώντας αυτές τις πληροφορίες, ένας αποσυνδεδεμένος κόμβος μπορεί πάντα να επανασυνδεθεί στο δίκτυο. Οι κόμβοι αποφασίζουν πού να συνδεθούν βασισμένοι μόνο στις τοπικές πληροφορίες, και διαμορφώνοντας έτσι ένα δυναμικό, αυτοδιοργανούμενο δίκτυο από ανεξάρτητες οντότητες. Αυτό το δίκτυο εικονικού επιπέδου εφαρμογής έχει σαν κόμβους τους Gnutella servents και για συνδέσεις του έχει τις ανοικτές συνδέσεις TCP [29].

Τα μηνύματα Ping δε περιέχουν δεδομένα και άρα είναι μηδενικού μήκους. Αντιπροσωπεύονται απλά από την επικεφαλίδα του μηνύματος όπου το πεδίο Payload_Descriptor έχει τη τιμή 0x00 και ο τομέας Payload_Length είναι 0x00000000. Περιέχει επίσης ένα μετρητή βημάτων TTL, που καθορίζει πόσες φορές το αίτημα μπορεί να διαβιβαστεί σε άλλους υπολογιστές. Ο servent χρησιμοποιεί τα μηνύματα αυτά για να εξετάσει ενεργά το δίκτυο και να αναζητήσει άλλους ομότιμους. Οι servents πρέπει να καταβάλλουν κάθε δυνατή προσπάθεια για να ελαχιστοποιηθεί η κίνηση του Ping στο δίκτυο [29].

#### 4.3.1.2 PONG

Οι ομότιμοι που λαμβάνουν το μήνυμα PING, μπορούν να επιλέξουν να απαντήσουν με ένα μήνυμα PONG. Έτσι ο αρχικός κόμβος λαμβάνοντας αυτό το πλήθος των μηνυμάτων, μπορεί να εγκαταστήσει επιπρόσθετες συνδέσεις TCP με άλλους ομότιμους.

Η δομή ενός περιγραφητή PONG απεικονίζεται στο Σχήμα 4.4, ενώ τα πεδία που το αποτελούν είναι [29]:




**Σχήμα 4.4** –Περιγραφητής PONG [29].

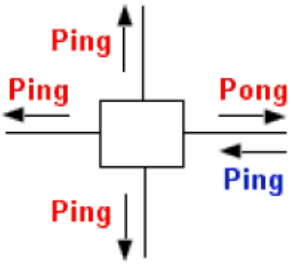
- **Port** - ο αριθμός πόρτας στον οποίο ο host, που έστειλε το PONG μήνυμα, μπορεί να δεχτεί εισερχόμενες συνδέσεις.
- **IP Address** - η IP διεύθυνση σε IPv4 format του αποκρινόμενου host που έστειλε το PONG μήνυμα.
- **Number of files shared** - ο αριθμός αρχείων που ο servent με τη δεδομένη IP διεύθυνση και πόρτα μοιράζεται στο δίκτυο.
- **Number of Kilobytes Shared** - ο αριθμός των δεδομένων σε kilobyte που ο servent με τη δεδομένη IP διεύθυνση και πόρτα μοιράζεται στο δίκτυο.

Ένας περιγραφητής PONG στέλνεται μόνο σε απάντηση σε έναν εισερχόμενο περιγραφητή PING. Όμως μπορεί και περισσότερα από ένα PONG να σταλούν σε απάντηση σε ένα PING, που επιτρέπει στο host να αποθηκεύει και να στέλνει πληροφορίες σχετικα με τις διευθύνσεις των servents.

Το πρωτόκολλο Gnutella διευκρινίζει επιπλέον μια πολιτική δρομολόγησης στην οποία ένα μήνυμα PONG μπορεί να σταλλεί μόνο κατά μήκος της ίδιας διαδρομής από την οποία είχε έρθει το αρχικό μύνημα PING (back-propagation), όπως απεικονίζεται στο Σχήμα 4.5.. Για αυτό το λόγο, κάθε servent διατηρεί μια μνήμη (ID cache) όλων των ping IDs που έχει δει σε ένα είδος routing table, με πληροφορίες σχετικά με το ποιά συνδέση το παρέδωσε. Αυτό είναι δυνατό δεδομένου ότι κάθε PING και PONG μήνυμα συνδέεται με ένα παγκοσμίως μοναδικό αναγνωριστικό (Globally Unique Identifier - GUID). Οι servents απλά απορρίπτουν κάθε λαμβανόμενο PONG που δεν ταιριάζει με τα PING αυτά. Επιπλέον, η ID cache επιτρέπει σε έναν servent να απορρίψει τους διπλούς περιγραφητές, που οφείλονται συνήθως στους βρόχους στην ad-hoc τοπολογία δικτύων.

Τα δεδομένα από τα μηνύματα PONG μπορούν να αναφέρονται σε έναν αυθαίρετο κόμβο. Το πρωτόκολλο δεν απαιτεί ότι το PONG πρέπει να αναφέρεται στον ίδιο κόμβο που το εκδίδει. Συνήθως ισχύει, αλλά μπορεί να δείχνει στην πραγματικότητα και σε έναν άλλο host. Ένας servent μπορεί να στέλνει μια σειρά από

μηνύματα PONG σαν απάντηση, συμπεριλαμβανομένων και των μηνυμάτων PONG από άλλους κόμβους.



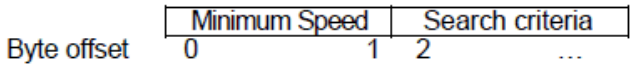

**Σχήμα 4.5** –Δρομολόγηση PING - PONG [29].

Τα μηνύματα PING και PONG χρησιμοποιούνται ουσιαστικά για την συντήρηση του δικτύου. Αντιπροσωπεύουν μια σημαντική μερίδα της συνολικής κυκλοφορίας σε ένα τέτοιου είδους P2P δίκτυο, περίπου το 60%-75% όλων των μηνυμάτων μέσω οποιασδήποτε σύνδεσης. Το πρωτόκολλο Gnutella συνιστά την ελαχιστοποίηση του αριθμού και τη συχνότητα των μυνημάτων PING που στέλνονται από οποιοδήποτε client.

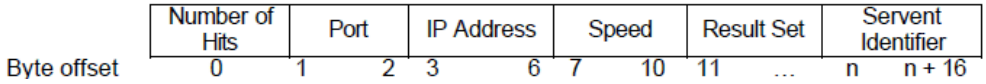
#### 4.3.1.3 QUERY

Τα μηνύματα QUERY στέλνονται για την λειτουργία της αναζήτησης δεδομένων στο δίκτυο Gnutella. Με αυτό το μήνυμα γίνεται η αναζήτηση ενός συγκεκριμένου αρχείου. Πέραν του ονόματος του αρχείου, περιέχει μερικούς περιορισμούς στη ταχύτητας μεταφοράς, π.χ. μπορεί να ζητήσει τουλάχιστον 10 kbyte/sec. Το μέγεθος του μηνύματος είναι στις περισσότερες περιπτώσεις κάτω από 200 bytes. Η δομή του μηνύματος QUERY απεικονίζεται στο Σχήμα 4.6, ενώ τα επιμέρους του πεδία είναι [29]:

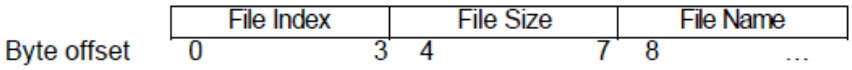
- **Minimum Speed** - η ελάχιστη ταχύτητα (σε kb/second) των servents που πρέπει να αποκριθούν σε αυτό το μήνυμα. Ένας servent λαμβάνοντας έναν περιγραφητή QUERY με την ελάχιστη ταχύτητα των x kb/s θα έπρεπε μόνο να ανταποκριθεί με ένα QUERYHIT εάν είναι σε θέση να επικοινωνήσει με ταχύτητα ≥ x kb/s
- **Search Criteria** - μια μηδενική τιμή (δηλ. 0x00) τερματίζει το αλφαριθμητικό για την αναζήτηση. Το μέγιστο μήκος αυτού του αλφαριθμητικού είναι προδιορισμένο από το πεδίο Payload_Length του περιγραφητή επικεφαλίδας.




**Σχήμα 4.6** – Περιγραφητής QUERY [29].




**Σχήμα 4.7** – Περιγραφητής QUERYHIT [29].




**Σχήμα 4.8** – Δομή πεδίου Number_of_Hits [29].

#### 4.3.1.4 QUERYHIT

Όταν ένας servent λαμβάνει ένα περιγραφητή QUERY, ανταποκρίνεται στέλνοντας ένα QUERYHIT εάν υπάρχει αντιστοιχία στο σύνολο των τοπικών δεδομένων του, δηλαδή αν κάποιος servent έχει τα δεδομένα που ζητήθηκαν στο QUERY.

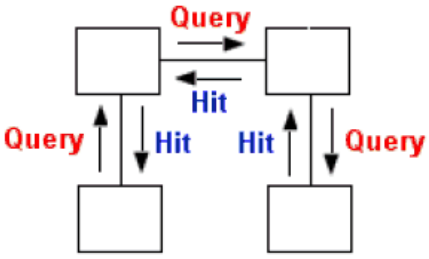
Αυτό το μήνυμα επιστρέφει μια ή περισσότερες πηγές για ένα αρχείο που ζητήθηκε από ένα QUERY. Το σημαντικότερο είναι ότι φέρει τις IP διευθύνσεις όλων αυτών των κόμβων, τη ταχύτητα σύνδεσής τους και ένα προσδιοριστικό αρχείων. Αυτό το προσδιοριστικό χρησιμοποιείται όταν ζητήθει το αρχείο μέσω HTTP για να αποθηκευτεί. Η δομή ενός περιγραφητή QUERYHIT απεικονίζεται στο Σχήμα 4.7 [29], όπου τα επιμέρους πεδία αναφέρονται σε:

- **Number of Hits** - ο αριθμός των QUERY hits που βρίσκονται στο result set.
- **Port** - ο αριθμός πόρτας στην οποία ο αποκρινόμενος host μπορεί να δεχτεί εισερχόμενες συνδέσεις.
- **IP Address** - η IP διεύθυνση του αποκρινόμενου host.
- **Speed** - η ταχύτητα (σε kb/second) του αποκρινόμενου host.
- **Result Set** - το μέγεθος του result set οριοθετείται από το μέγεθος του πεδίου Payload_Length στον περιγραφητή επικεφαλίδας. Είναι ένα σύνολο απαντήσεων στο αντίστοιχο ερώτημα QUERY. Η δομή του απεικονίζεται στο Σχήμα 4.8, περιλαμβάνει τα στοιχεία Number_of_Hits και τα επιμέρους πεδία του αναφέρονται σε [29]:

- **File Index** - ένας αριθμός που προσδιορίζει το κάθε ταίριασμα αρχείου με το αντίστοιχο ερώτημα. Ο αριθμός αυτός είναι μοναδικός κάθε φορά και ορίζεται από τον αποκρινόμενο host.
- **File Size** - το μέγεθος (σε bytes) του αρχείου του οποίου δείκτης είναι ο File_Index.
- **File Name** – η διπλή μηδενική τιμή (δηλ. 0x0000) με την οποίη τελειώνει το όνομα του αρχείου του οποίου δείκτης είναι File_Index.

➢ **Servent Identifier** - Ένα 16-byte αλφαριθμητικό που προσδιορίζει μοναδικά τον αποκρινόμενο servent στο δίκτυο. Αυτό είναι συνήθως κάποια λειτουργία της διεύθυνσης δικτύου του servent. Ο Servent Identifier συμβάλλει στη λειτουργία του περιγραφητή PUSH, όπως θα περιγράψουμε στην συνέχεια.

Τα μηνύματα QUERYHIT στέλνονται μόνο σε απάντηση σε ένα εισερχόμενο μήνυμα QUERY, όπως απεικονίζεται στο Σχήμα 4.9. Ένας servent πρέπει μόνο να απαντήσει σε ένα QUERY με ένα QUERYHIT εάν περιέχει δεδομένα που ικανοποιούν αυστηρά τα κριτήρια αναζήτησης ερώτησης (Query Search Criteria).




**Σχήμα 4.9** – Δρομολόγηση μηνυμάτων QUERY και QUERYHIT [29].

Το πεδίο Descriptor_Id που βρίσκεται στο περιγραφητή επικεφαλίδας για το QUERYHIT πρέπει να περιέχει την ίδια τιμή με αυτόν του αντίστοιχου περιγραφητή QUERY. Αυτό επιτρέπει σε έναν servent να προσδιορίσει τους περιγραφητές QUERYHIT που συνδέονται με τους περιγραφητές QUERY που ο servent δημιούργησε.

### 4.3.2 Μηχανισμός αναζήτησης αρχείου

Ο βασικός μηχανισμός αναζήτησης που υιοθετείται από το Gnutella είναι η περιορισμένη πλημμύρα (Limited Flooding) [30]. Στην πλημμύρα, ένας ομότιμος που αναζητεί ένα αρχείο θέτει ένα ερώτημα και το στέλνει σε όλους τους ομότιμους του γείτονες. Ο κόμβος που λαμβάνει το ερώτημα το διαβιβάζει σε όλους τους δικούς του γείτονες εκτός από το κόμβο από τον οποίο έλαβε το μήνυμα. Με αυτόν τον τρόπο, ένα ερώτημα διαδίδεται μέχρι ένα προκαθορισμένο αριθμό αλμάτων (TTL) από τον αρχικό ομότιμο. Το TTL που εφαρμόζεται από το Gnutella είναι γενικά για τις δημοφιλείς αναζητήσεις αν και το TTL είχε εισαχθεί για τις σπάνιες αναζητήσεις.

### 4.3.3 Λήψη αρχείου

Απο τη στιγμή που ένας servent λαμβάνει ένα μήνυμα QUERYHIT, μπορεί να αρχίσει η άμεση λήψη ενός από τα αρχεία που περιγράφονται από το Result Set [29]. Τα δεδομένα των αρχείων δεν μεταφέρονται μέσα από το δίκτυο Gnutella. Η ανταλλαγή δεδομένων, γίνεται μέσω μιας απευθείας σύνδεσης μεταξύ των ομοτίμων.

Το πρωτόκολλο που χρησιμοποιείται για τη λήψη των αρχείων είναι το standard HTTP. Ο servent που ξεκινάει τη λήψη στέλνει ένα αίτημα αλφαριθμητικό στον target server της ακόλουθης μορφής:

*GET /get/<File Index>/<File Name>/ HTTP/1.0\r\n*
*Connection: Keep-Alive\r\n*
*Range: bytes=0-\r\n*
*User-Agent: Gnutella\r\n*
*\r\n*

όπου <File Index> και <File Name> είναι ένα από τα ζεύγη File Index / File Name από το Result Set που βρίσκονται στο QUERYHIT. Για παράδειγμα, εάν το Result Set από έναν περιγραφητή QUERYHIT περιείχε τα:

| | |
|---|---|
| **File Index** | *2468* |
| **File Size** | *4356789* |
| **File Name** | *Foobar.mp3\x00\x00* |

τότε η αίτηση λήψης για το αρχείο θα ήταν:

*GET /get/2468/Song.mp3/ HTTP/1.0\r\n*

*Connection: Keep-Alive\r\n*
*Range: bytes=0-\r\n*
*User-Agent: Gnutella\r\n*
*\r\n*

Ο server που λαμβάνει το αίτημα αποκρίνεται με τις ανάλογες επικεφαλίδες HTTP 1.0 όπως:
*HTTP 200 OK\r\n*
*Server: Gnutella\r\n*
*Content-type: application/binary\r\n*
*Content-length: 4356789\r\n*
*\r\n*

Ακολουθούν τα δεδομένα των αρχείων, συμπεριλαμβανομένου τον αριθμό των bytes που διευκρινίζεται στο Content-length που παρέχεται στην απάντηση HTTP του κεντρικού υπολογιστή.

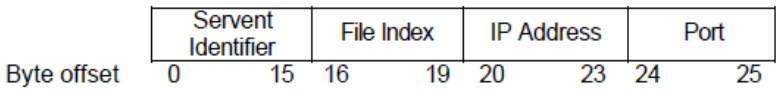
Το πρωτόκολλο Gnutella παρέχει υποστήριξη για την παράμετρο HTTP Range, έτσι ώστε οι λήψεις που για κάποιο λόγο διακόπηκαν να μπορούν να συνεχιστούν από το σημείο στο οποίο είχαν σταματήσει.

#### 4.3.3.1 PUSH

Η δομή ενός μηνύματος PUSH απεικονίζεται στο Σχήμα 4.10, όπου τα επιμέρους πεδία είναι [29]:

- **Servent Identifier** - ένα 16-byte αλφαριθμητικό που προσδιορίζει μοναδικά τον servent στο δίκτυο από τον οποίο ζητήθηκε να προωθήσει το αρχείο με το δείκτη File_Index. Ο servent που αρχίζει το αίτημα προώθησης πρέπει να θέσει αυτό το πεδίο στο Servent_Identifier που επιστρέφεται στον αντίστοιχο περιγραφητή QUERYHIT. Αυτό επιτρέπει στον παραλήπτη ενός αιτήματος προώθησης να καθορίσει ή όχι αν είναι ο στόχος εκείνου του αιτήματος.
- **File Index** - ο δείκτης που προσδιορίζει μοναδικά το αρχείο που προωθείται από το στόχο servent. Ο servent που αρχίζει το αίτημα προώθησης πρέπει να θέσει σε αυτό τον πεδίο μια τιμή εκ των πεδίων του File_Index από το Result Set του αντίστοιχου περιγραφητή QUERYHIT.
- **IP Address** - η IP διεύθυνση του host στον οποίο το αρχείο με το File_Index πρέπει να προωθηθεί.

- **Port** - η πόρτα στην οποία το αρχείο με το File_Index πρέπει να προωθηθεί.




**Σχήμα 4.10** – Περιγραφητής PUSH [29].

Όταν ένας servent λαμβάνει ένα περιγραφητή PUSH μπορεί να ενεργήσει ύστερα από το αίτημα προώθησης αν και μόνο αν το πεδίο servent_Identifier περιέχει την τιμή του δικού του αναγνωριστικού του servent. Το πεδίο Descriptor_Id στην επικεφαλίδα του περιγραφητή του περιγραφητή επικεφαλίδας δεν πρέπει να περιέχει την ίδια τιμή με αυτόν του αντίστοιχου περιγραφητή QUERYHIT, αλλά πρέπει να περιέχει μια νέα τιμή που παράγεται από τον αλγόριθμο παραγωγής Descriptor_Id του servent [29].

#### 4.3.3.2 Firewalled Servents

Δεν είναι πάντοτε δυνατή η εγκαθίδρυση μιας άμεσης σύνδεσης σε ένα Gnutella servent στη προσπάθεια να αρχίσει η λήψη ενός αρχείου [29]. Ο servent μπορεί, π.χ. να είναι πίσω από ένα firewall που δεν επιτρέπει τις εισερχόμενες συνδέσεις στην αντίστοιχη θύρα του Gnutella client. Τότε ο firewalled servent στέλνει ένα πακέτο PUSH στον κόμβο ο οποίος διαθέτει το ζητούμενο αρχείο, μέσω της διαδρομής που το πακέτο QUERYHIT είχε σταλθεί αρχικά. Το πακέτο PUSH πληροφορεί αυτό τον κόμβο ότι ο firewalled servent επιθυμεί να λάβει το αρχείο αλλά δεν μπορεί να αρχίσει η σύνδεση HTTP. Τότε ο κόμβος με το ζητούμενο αρχείο θα είναι αυτός που θα ξεκινήσει τη σύνδεση και θα προσπαθήσει να συνδεθεί απευθείας με τον firewalled servent . Αν ούτε και αυτό είναι εφικτό, σημαίνει ότι και αυτός είναι πίσω από ένα firewall. Σε αυτήν την περίπτωση, η μεταφορά αρχείων δεν μπορεί να πραγματοποιηθεί.

Εάν η σύνδεση είναι εφικτή από τον firewalled servent προς στο servent που άρχισε το αίτημα PUSH , τότε ο firewalled servent θα έπρεπε αμέσως να στείλει το εξής:

*GIV <File Index>:<Servent Identifier>/<File Name>\n\n*

όπου <File Index> και <Servent Identifier> είναι οι τιμές των πεδίων File Index και Servent Identifier αντίστοιχα, από το λαμβανόμενο αίτημα PUSH. Το <File Name>

είναι το όνομα του αρχείου στον τοπικό πίνακα αρχείων που διαθέτει ο κάθε ομότιμος, το οποίο προσδιορίζεται από το <File Index>. Ο servent λαμβάνοντας την επικεφαλίδα του αιτήματος GIV, δηλαδή του αιτήματος PUSH, θα έπρεπε να εξάγει τα <File Index> και <File Name> από την επικεφαλίδα για να κατασκευάσει ένα HTTP αίτημα GET της ακόλουθης μορφής:

*GET /get/<File Index>/<File Name>/ HTTP/1.0\r\n*
*Connection: Keep-Alive\r\n*
*Range: bytes=0-\r\n*
*User-Agent: Gnutella\r\n*
*\r\n*

Από κει και έπειτα, η διαδικασία λήψης του αρχείου συνεχίζεται κατά τα γνωστά, όπως έχει αναφερθεί σε προηγούμενη παράγραφο.

### 4.3.4 Δρομολόγηση μηνυμάτων

Ο peer-to-peer χαρακτήρας του δικτύου Gnutella απαιτεί από τους servents να δρομολογούν την κυκλοφορία δικτύων (αιτήματα QUERY, απαντήσεις σε αιτήματα QUERY, αιτήματα PUSH, κ.λπ.) με κατάλληλο τρόπο. Μέχρι στιγμής έχουμε αναφέρει κάποιους κανόνες δρομολόγησης μηνυμάτων για ένα τυπικό Gnutella servent. Στην συνέχεια συνοψίζουμε αυτούς τους κανόνες δρομολόγησης των περιγραφητών του πρωτοκόλλου [29]:

- Τα μηνύματα PONG μπορούν να σταλλούν μόνο κατά μήκος της ίδιας διαδρομής που μετέφερε τον εισερχόμενο περιγραφητή PING. Αυτό εξασφαλίζει ότι μόνο εκείνοι οι servents που δρομολόγησαν το μήνυμα PING θα λάβουν το μήνυμα PONG σαν απάντηση. Ένας servent που λαμβάνει έναν περιγραφητή PONG με Descriptor ID=n, αλλά δεν έχει δει ένα μήνυμα PING με το ίδιο Descriptor ID, θα πρέπει να απομακρύνει το μήνυμα PONG από το δίκτυο.
- Τα μηνύματα QUERYHIT μπορούν να σταλθούν μόνο κατά μήκος της ίδιας διαδρομής από την οποία ήρθε ο εισερχόμενος περιγραφητής QUERY. Αυτό εξασφαλίζει ότι μόνο εκείνοι οι servents που δρομολόγησαν το QUERY θα λάβουν το QUERYHIT σαν απάντηση. Ένας servent που λαμβάνει ένα QUERYHIT με Descriptor ID=n, αλλά δεν έχει δει περιγραφητή QUERY με

το ίδιο Descriptor ID θα πρέπει να απομακρύνει τον περιγραφητή QUERYHIT από το δίκτυο.

- Τα μηνύματα PUSH μπορούν μόνο να σταλλούν κατά μήκος της ίδιας διαδρομής από την οποία ήρθε το QUERYHIT. Αυτό εξασφαλίζει ότι μόνο εκείνοι οι servents που δρομολόγησαν τον περιγραφητή QUERYHIT θα μπορούν να δουν τον περιγραφητή PUSH. Ένας servent που λαμβάνει μήνυμα PUSH με Servent_Identifier=n, αλλά δεν έχει δει περιγραφητή QUERYHIT με το ίδιο προσδιοριστικό, πρέπει να απομακρύνει το πακέτο PUSH από το δίκτυο. Οι περιγραφητές PUSH δρομολογούνται από τον Servent_Identifier, όχι από τον Descriptor_Id.
- Ένας servent θα διαβιβάσει τα εισερχόμενα μηνύματα PING και QUERY σε όλους τους άμεσα συνδεδεμένους σε αυτόν, servents, εκτός από αυτόν που παρέδωσε το εισερχόμενο PING ή QUERY.
- Ένας servent θα μειώσει το πεδίο TTL του περιγραφητή επικεφαλίδας, και θα αυξήσει το Hops πεδίο του, πρίν να διαβιβάσει το μήνυμα σε οποιοδήποτε γειτονικό του servent. Εάν, μετά από τη μείωση της επικεφαλίδας του πεδίου TTL, ο τομέας TTL βρεθεί να είναι μηδέν, ο περιγραφητής δεν διαβιβάζεται κατά μήκος οποιασδήποτε σύνδεσης και απομακρύνεται από το δίκτυο.
- Ένας servent που λαμβάνει ένα πακέτο με το ίδιο Payload Descriptor και Descriptor ID όπως αυτό που έλαβε πριν, πρέπει να προσπαθήσει να αποφύγει να διαβιβάσει τον περιγραφητή σε οποιοδήποτε συνδεδεμένο servent. Οι προοριζόμενοι παραλήπτες έχουν λάβει ήδη έναν τέτοιο περιγραφητή και στέλνοντας τον πάλι σπαταλά μόνο το εύρος ζώνης δικτύων.

### 4.3.5 Το πρωτόκολλο Gnutella v0.6.

Τα πλήρως αποκεντρικοποιημένα και αδόμητα P2P δίκτυα όπως το Gnutella, είναι ίσως τα δημοφιλέστερα overlay δίκτυα για τη διανομή αρχείων [31]. Η απουσία μιας καθορισμένης δομής και ενός κεντρικού ελέγχου καθιστά τέτοια συστήματα πιό ανθεκτικά και αυτοδιοργανούμενα έναντι των δομημένων συστημάτων.

Ωστόσο, το κύριο πρόβλημα σε αυτό το είδος δικτύων είναι η δυνατότητα κλιμάκωσης (scalability) [30]. Πρώτον, λόγω της παραγωγής του μεγάλου αριθμού περιττών μηνυμάτων που προκύπτουν με τη μέθοδο πλυμμήρας και δεύτερον, η χρήση των τιμών TTL στα μηνύματα, που εισήχθησαν για να ανακουφίσουν την

επίδραση της πλημμύρας, έτεινε να μειώνει τον αριθμό των ομοτίμων στους οποίους γινόταν έλεγχος κατά τη διαδικασία της αναζήτησης, όπου συνήθως ελέγχονταν περίπου 10.000 κόμβοι. Δεδομένου ότι αυτά τα δίκτυα γίνονται ολοένα και δημοφιλέστερα, η ποιότητα της υπηρεσίας υποβιβαθμίζεται με ραγδαίο ρυθμό [35].

Για να παρέχει συνεπώς το δίκτυο δυνατότητα κλιμάκωσης, το Gnutella αναβαθμίζει συνεχώς κάποια χαρακτηριστικά γνωρίσματά του και εισάγει νέες έννοιες. Έτσι οι κύριοι υπεύθυνοι για την ανάπτυξη των Gnutella clients μαζί με υπόλοιπους συμμετέχοντες στο ανοικτού κώδικα πρωτόκολλο Gnutella πρόσθεσαν μια σειρά από τροποποιήσεις στο αρχικό πρωτόκολλο, όπως και κάποιες πολύ ουσιαστικές αλλαγές. Το νέο πρωτόκολλο αποκαλείται Gnutella v0.6 και εμφανίστηκε τον Οκτώβριο του 2000, επτά μήνες μετά την αρχική έκδοση του πρωτοκόλλου (έκδοση v0.4 - Μάρτιος 2000).

Οι βελτιώσεις αυτές μπορούν να ταξινομηθούν σε δύο ευρείες περιοχές:

- βελτιώσεις που αφορούν τις τεχνικές αναζήτησης και
- βελτιώσεις που αφορούν τη τροποποίηση της δομής στη τοπολογία του overlay δικτύου, με γενικότερο σκοπό να ενισχυθεί η αποδοτικότητα στην αναζήτηση ενός αρχείου.

Στις βελτιωμένες τεχνικές αναζήτησης, έχουν εισαχθεί το:

- Dynamic QUERY (DQ),
- QUERY Routing Protocol (QRP) και
- ο μηχανισμός PONG-caching για καλύτερη διαχείρηση του εύρους ζώνης και μείωση της περιττής κίνησης στο δίκτυο.

Η παρουσίαση της έννοιας των ultrapeers και leafpeers αποτέλεσε μια από τη σημαντικότερη τροποποίηση στη τοπολογία του αδόμητου δίκτυου, δίνοντας μία ιεραρχική δομή στο δίκτυο μαζί με μια two-tier τοπολογία δικτύων [30]. Να αναφερθεί ότι σχεδόν το 99% πλέον των Gnutella clients χρησιμοποιούν το πρωτόκολλο Gnutella v0.6 .

#### 4.3.5.1 Εισαγωγή

Το νέο Gnutella (έκδοση v0.6) ανήκει στη κατηγορία των υβριδικών P2P συστημάτων. Πρόκειται για ένα two-tier overlay δίκτυο όπου στο ένα επίπεδο (Top-level overlay) υπάρχουν οι συνδέσεις των ultrapeers, και στο άλλο επίπεδο βρίσκονται οι leaf-peers. Οι ιεραρχικές αρχιτεκτονικές χωρίζουν τους ομοτίμους στο

δίκτυο σε δύο ομάδες: στους ultrapeers και leaf-peers ή leaf nodes. Υπάρχει ωστόσο ακόμα ένας τύπος ομοτίμων που καλούνται legacy-peers, τα οποία είναι παρόντα στο ultra-peer επίπεδο και δεν δέχονται οποιεσδήποτε συνδέσεις με τους leaf-peers. (Σχήμα 4.11) [31].

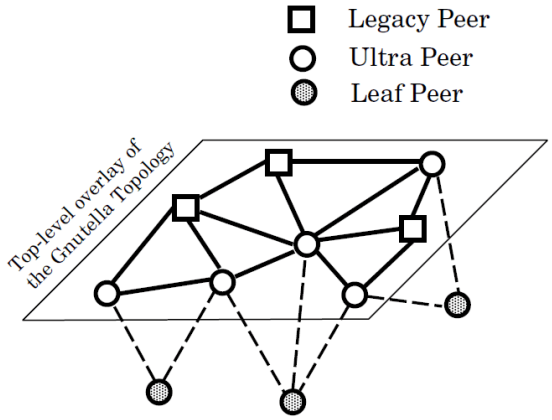
Όπως στα κεντρικοποιημένα P2P δίκτυα, οι leaf-peers συνδέονται με τους ultrapeers και στους οποίους φορτώνουν μια λίστα με τα ονόματα των αρχείων που αυτοί μοιράζονται. Κατά συνέπεια, οι ultrapeers διατηρούν έναν κατάλογο των περιεχομένων που διαμοιράζονται οι τοπικά συνδεδεμένοι κόμβοι τους. Οι ultrapeers συνδέονται μεταξύ τους χρησιμοποιώντας ένα παρόμοιο πρωτόκολλο με αυτό που χρησιμοποιείται στο Gnutella v0.4, επιτρέποντας στους ultrapeers να προωθούν αιτήματα QUERY στους γειτονικούς τους ultrapeers, επεκτείνοντας με αυτόν τον τρόπο την επικοινωνία στο δικτύου. Επίσης, το bootstrapping γίνεται πλέον μέσω των ultrapeers [30].

Σε ένα υβριδικό δίκτυο υπάρχουν κάποιοι κανόνες και περιορισμοί όσον αφορά τις συνδέσεις που πραγματοποιούνται στο δίκτυο [32]:

- οι leaf nodes δεν έχουν εξερχόμενες συνδέσεις με άλλους leaf nodes. Δεν υπάρχει καμία άμεση σύνδεση μεταξύ δύο οποιωνδήποτε leaf-peers στο overlay δίκτυο.
- οι leaf nodes δεν δέχονται εισερχόμενες συνδέσεις.
- οι ultra-peers πραγματοποιούν εξερχόμενες συνδέσεις μόνο με άλλους super peers.

Για την αποφυγή των σημείων συμφόρησης όσον αφορά την ανανέωση και την διερεύνηση του καταλόγου, οι ultrapeers δέχονται μόνο έναν περιορισμένο αριθμό συνδέσεων από τους leaf nodes. Για την επιλογή τους, οι ultrapeers πρέπει να έχουν μεγάλο εύρος ζώνης και μεγάλη υπολογιστική ισχύ προκειμένου να αποφευχθούν τα προβλήματα κλιμάκωσης στην επεξεργασία των ερωτημάτων από τους γειτονικούς ultrapeers. Επιλέγονται από τον Gnutella client αφού αποδείξουν ότι παραμένουν στο δίκτυο για σχετικά μεγάλες χρονικές περιόδους. Εάν λάβουν τον ελάχιστα απαιτούμενο αριθμό συνδέσεων με τους client nodes μέσα σε έναν καθορισμένο χρόνο, παραμένουν ως ultrapeers. Διαφορετικά μετατρέπονται σε κανονικούς servents. Εάν κανένας superpeer δεν είναι διαθέσιμος, θα ξαναγίνει προσπάθεια επιλογής για μια άλλη περίοδο δοκιμασίας. Στην πραγματικότητα, οι ultrapeers προστατεύουν τους leaf nodes από τη λήψη των περιττών μηνυμάτων που

μπορούν να κατακλύζουν τις συνδέσεις τους και να προκαλέσουν τα προβλήματα που παρατηρήθηκαν στο Gnutella v0.4 [33] .




**Σχήμα 4.11** – Two-tier τοπολογία Gnutella v0.6 [36].

#### 4.3.5.2 PONG caching

Τα μηνύματα PING και PONG διαμόρφωναν ένα σημαντικό ποσοστό της κυκλοφορίας στην αρχική μορφή του Gnutella (έκδοση v0.4). Για να μειωθεί αυτή η κατανάλωση εύρους ζώνης έχουν εισαχθεί μηχανισμοί caching για τα μηνύματα PONG, που περιλαμβάνουν εφαρμογή του Gnutella PONG cache και το σύστημα μείωσης των PING.

Στο δίκτυο Gnutella v0.6, μόνο οι ultrapeers δέχονται συνδέσεις από άλλους υπολογιστές. Τα μηνύματα PONG «διαφημίζουν» τους υπολογιστές που ένας ομότιμος μπορεί να συνδεθεί, έτσι όλα τα PONG που κυκλοφορούν στο δίκτυο θα πρέπει να αφορούν τους ultrapeers. Έτσι μόνο οι ultrapeers διατηρούν πίνακα PONG μηνυμάτων, ο οποίος ανανεώνεται περιοδικά. Αντί ο ultrapeer να προωθεί τα μηνύματα PING από τους leaf peers του στους υπόλοιπους ultrapeers, τους στέλνει σαν απάντηση τα μηνύματα PONG (τυπικά 10) που έχει αποθηκεύσει στην ανταλλαγή των PING-PONG μηνυμάτων με άλλους ultrapeers. Αυτό συνεπάγεται πολύ λιγότερα μηνύματα PING στο δίκτυο.

Ένας ultrapeer του LimeWire διαβιβάζει τα μηνύματα PONG στους leaf peers του, ακόμη και αν δεν έχουν στείλει μηνύματα PING, για να εξασφαλιστεί ότι θα ανακαλύψουν αρκετούς ultrapeers [34]. Η προσωρινή αποθήκευση PONG συνεπάγεται:

- τη μείωση του όγκου της κίνησης στο δίκτυο και

- οι απαντήσεις τείνουν να είναι περισσότερο αντιπροσωπευτικές καθώς η κάθε προσωρινή αποθήκευση μεταφέρει μηνύματα PONG από τους ομότιμους κατά μήκος του δικτύου [35].

#### 4.3.5.3 QUERY

Στην δομή του μηνύματος QUERY προστέθηκαν τα ακόλουθα πεδία [29]:

- **Optional extensions** - οι επιτρεπόμενοι τύποι επέκτασης ειναι:
  - HUGE (Hash/URN Gnutella Extensions),
  - XML
  - GGEP
- Στο πεδίο Result Set υπάρχουν τρία επιπλέον πεδία:
  - **11ο byte** - τύποι επέκτασης (HUGE, GGEP, plain),
  - **12ο byte** - προτεινόμενο EQHD block, που περιέχει επιπρόσθετες πληροφορίες και
  - **13ο byte** - Private vendor specific data.

#### 4.3.5.4 PUSH

Στην δομή του μηνύματος PUSH προστέθηκε το πεδίο Optional GGEP extension [29].

#### 4.3.5.5 Τεχνικές αναζήτησης

Το νέο πρωτόκολλο ακολουθεί την μέθοδο αναζήτησης ερωτημάτων βασισμένη στην περιορισμένη πλημμύρα (limited flood based query search). Ένα ερώτημα ενός ultra-peer προωθείται στους leaf-peers του με TTL (0) και σε όλους τους ultra-peer γείτονές του με TTL μειωμένο κατά ένα, προφανώς μόνο όταν TTL > 0. Ένας leaf-peer δεν διαβιβάζει οποιοδήποτε λαμβανόμενο ερώτημα. Από την άλλη πλευρά, οι ultra-peers πραγματοποιούν την αναζήτηση ερωτήματος εκ μέρους των leaf-peers τους. Το ερώτημα ενός leaf-peer στέλνεται αρχικά στους συνδεδεμένους με αυτόν ultra-peers, οι οποίοι στη συνέχεια το προωθούν ταυτόχρονα στους γειτονικούς τους ultra-peers μέχρι έναν περιορισμένο αριθμό αλμάτων (hops). Τονίζεται ότι η

μέθοδος της περιορισμένης πλυμμήρας εφαρμόζεται πλέον μόνο μεταξύ των ultra-peers [30].

Το νέο πρωτόκολλο χρησιμοποιεί κάποιες άλλες τεχνικές με σκοπό την μείωση της κίνησης των περιττών μηνυμάτων στο δίκτυο και της αποτελεσματικότερης αναζήτησης. Πιο αξιοσημείωτες είναι [36]:

- **Dynamic Query (DQ)** - τεχνική αναζήτησης ερωτήματος κατά την οποία ένας ultra-peer διαβιβάζει σταδιακά ένα ερώτημα σε 3 βήματα (TTL (1), TTL (2), TTL (3) αντίστοιχα) μέσω κάθε TCP σύνδεσης, μετρώντας παράλληλα την ανταπόκριση σε εκείνο το ερώτημα. Ο ultra-peer μπορεί να σταματήσει την προώθηση του ερωτήματος σε οποιοδήποτε βήμα εάν λάβει ένα ικανοποιητικό αριθμό από QUERYHIT, το οποίο γεγονός μειώνει σημαντικά το ποσό της κίνησης που προκαλείται από δημοφιλείς αναζητήσεις. Συνεπώς αυτή η τεχνική θα χρησιμοποιήσει το TTL (3) μόνο για τις σπάνιες αναζητήσεις.
- **Query Routing Protocol (QRP)** - τεχνική αναζήτησης ερωτήματος κατά την οποία ένας ultra-peer προωθεί ένα ερώτημα μόνο στους συνδεδεμένους με αυτόν leaf-peers, οι οποίοι ενδέχεται να έχουν ένα αποτέλεσμα. Αυτό συνεπάγεται την αποδοτικότερη αναζήτηση στα πιο σπάνια αρχεία. Ένας leaf-peer δημιουργεί ένα πίνακα hash όλων των αρχείων που μοιράζεται και το στέλνει σε όλους τους ultra-peers που έχει συνδεθεί. Κατά συνέπεια, όταν φθάνει ένα ερώτημα σε έναν ultra-peer, αυτό διαβιβάζεται μόνο σε εκείνους τους συνδεδεμένους leaf-peers που θα έχουν μηνύματα QUERYHIT και που θα έχουν δηλαδή το αρχείο. Έτσι η αναζήτηση πραγματοποιείται μόνο στο στρώμα ultra-peer, δεδομένου ότι οι ultra-peers περιέχουν τους δείκτες των «παιδιών» τους. Ο ultrapeer αποφασίζει ποιο ερώτημα θα προωθήσει, χρησιμοποιώντας αυτόν το μηχανισμό.

#### 4.3.5.6 Πρωτόκολλο χειραψίας

Πολλοί clients χρησιμοποιούνται για τη πρόσβαση στο δίκτυο με το νέο πρωτόκολλο Gnutella, όπως για π.χ. Limewire, Bearshare, gtk-Gnutella. Μέσω της χειραψίας ένας ομότιμος εγκαθιστά τη σύνδεση ή συνδέσεις με έναν ή περισσότερους ultra-peers [32]. Για να ξεκινήσει το πρωτόκολλο χειραψίας ένας ομότιμος αρχικά

συλλέγει τη διεύθυνση ενός ultra-peer που είναι εκείνη τη στιγμή συνδεμένος στο δίκτυο μέσα από μια ομάδα από online ultra-peers. Ένας ομότιμος μπορεί να συλλέξει τη λίστα με τις διευθύνσεις των online ultra-peers από το σύστημα GwebCache ή μέσω του PONG caching ή ακόμα από το σκληρό του δίσκο, ο οποίος είχε λάβει μια τέτοια λίστα από τη προηγούμενη φορά που έτρεξε το πρόγραμμα και συνδέθηκε στο δίκτυο. Το πρωτόκολλο χειραψίας χρησιμοποιείται για να δημιουργούνται νέες συνδέσεις TCP. Μια χειραψία αποτελείται από 3 ομάδες επικεφαλίδων. Τα βήματα της χειραψίας διαμορφώνονται στην συνέχεια ως εξής [30]:

- Ο ομότιμος που αρχικοποιεί τη σύνδεση στέλνει την πρώτη ομάδα επικεφαλίδων. Αυτή αναφέρει στον απομακρυσμένο ομότιμο για τα χαρακτηριστικά γνωρίσματά του και τη κατάσταση, που υποδηλώνει τον τύπο γείτονα (leaf ή ultra) που θέλει να είναι.
- Ο ομότιμος που λαμβάνει τη σύνδεση αποκρίνεται με μια δεύτερη ομάδα επικεφαλίδων που μεταβιβάζει ουσιαστικά το μήνυμα εάν συμφωνεί με την πρόταση του ομοτίμου που ξεκίνησε τη σύνδεση ή όχι.
- Τέλος, ο αρχικός ομότιμος στέλνει μια τρίτη ομάδα επικεφαλίδων για να επιβεβαιώσει και να εγκαθιδρύσει τη σύνδεση.

Το αρχικό πρωτόκολλο χειραψίας του Gnutella (έκδοση 0.4) περιγράφεται ως εξής:

1. Ο client εγκαθιστά μια σύνδεση TCP με τον server.
2. Ο client στέλνει : "GNUTELLA CONNECT/0.4<lf><lf> ".
3. Ο server αποκρίνεται με "GNUTELLA OK<lf><lf> ".
4. Ο client και ο server στέλνουν δυαδικά μηνύματα κατά βούληση.

Το πρόβλημα με αυτό το σύστημα είναι ότι δεν παρέχει κανέναν τρόπο προσαρμογής και καλύτερης εκμετάλευσης των servents στο πρωτόκολλο Gnutella.
Οι Servents δεν μπορούν να ανακαλύψουν τυχόν προηγμένες ικανότητες και βελτιώσεις στους άλλους ομοτίμους αλλά και αναβαθμίσεις του ίδιου του πρωτοκόλλου. Το νέο πρωτόκολλο Gnutella (έκδοση 0.6) επιτρέπει στους Servents να ανταλλάσουν πρόσθετες πληροφορίες και είναι γενικά ασφαλέστερο. Η περιγραφή του νέου πρωτοκόλλου δίνεται παρακάτω [32]:

1. Ο client εγκαθιστά μια σύνδεση TCP με τον server.
2. Ο client στέλνει " GNUTELLA CONNECT/0.6<cr><lf> ".

3. Ο client στέλνει σε όλες τις επικεφαλίδες ικανότητας όπου κάθε μια ολοκληρώνεται από το "<cr><lf>", με ένα πρόσθετο "<cr><lf>" στο τέλος.
4. Ο server αποκρίνεται με " GNUTELLA/0.6 200 OK<cr><lf> ".
5. Ο server στέλνει όλες τις επικεφαλίδες του, με την ίδια διαμόρφωση όπως στο βήμα 3.
6. Ο client στέλνει " GNUTELLA/0.6 200 OK<cr><lf>, με την ίδια διαμόρφωση όπως στο βήμα 4.
7. Ο client και ο server στέλνουν δυαδικά μηνύματα κατά βούληση, χρησιμοποιώντας τις πληροφορίες που λαμβάνονται στα βήματα 3 και 5.

Να σημειωθεί ότι καμία χειραψία επικεφαλίδων του νέου πρωτοκόλλου δεν μπορεί να χρησιμοποιηθεί στη χειραψία της έκδοσης 0.4. .

#### 4.3.5.7 BYE

Το μήνυμα BYE χρησιμοποιείται για να ενημερώσει τους servents ότι ένας κόμβος που συνδέεται μαζί τους ετοιμάζεται να αποσυνδεθεί από το δίκτυο και θα τερματιστεί η σύνδεση. Από την στιγμή που η εφαρμογή αυτού του μηνύματος είναι προαιρετική, μια ειδική επικεφαλίδα πρέπει να σταλλεί κατά τη διάρκεια της χειραψίας για τη δημιουργία σύνδεσης.

Οι servents δεν πρέπει να στείλουν αυτό το μήνυμα στους ομοτίμους που δεν έχουν δείξει ότι το υποστηρίζουν. Το πεδίο TTL στο μήνυμα πρέπει να τεθεί ίσο με τιμή 1, ώστε να αποφευχθεί μια κατά λάθος μετάδοση του μηνύματος. Με την παραλαβή του μηνύματος ο servent κλείνει αμέσως τη σύνδεση. Ο ομότιμος που στέλνει το μήνυμα πρέπει να περιμένει μερικά δευτερόλεπτα ώστε ο απομακρισμένος servent να κλείσει τη σύνδεση πρίν τον ομότιμο. Κανένα δεδομένο δεν μπορεί να σταλλεί μετά από το μήνυμα αυτό [36].

Τα πεδία του μυνήματος είναι:

- **Bytes 0-1** - Code, επιστρέφει κωδικούς προσδιορισμένους από το SMTP (Simple Mail Transfer Protocol). Για παράδειγμα, η τιμή 200 σημαίνει ότι όλα είναι εντάξει.
- **Byte 2** - περιγραφητής αλφαριθμητικό.

## 4.4 Gnutella crawler

Ο Gnutella crawler [37] είναι ένα πρόγραμμα λογισμικού που χρησιμοποιείται για τη συλλογή στατιστικών πληροφοριών του δικτύου διανομής αρχείων Gnutella, όπως τον αριθμό χρηστών, την πίτα αγοράς των διαφόρων clients και τη γεωγραφική κατανομή της βάσης των χρηστών. Οι αρχικοί crawlers χρησιμοποιούσαν τα Ping/Pong μηνύματα για να ανακαλύψουν τους συνδεδεμένους hosts με το δίκτυο. Παρά το γεγονός ότι αυτή η μέθοδος είναι ακόμα χρησιμοποιήσιμη, είναι πολύ αργή για να συλλέξει αρκετά στοιχεία για μια τοπολογική επισκόπηση του δικτύου Gnutella καθώς απαιτεί την αρχικοποίηση των πλήρων συνδέσεων Gnutella. Μια επέκταση έχει προστεθεί στο πρωτόκολλο Gnutella για να επιτρέπει στους crawlers να έχουν γρήγορη πρόσβαση γρήγορα στους ultrapeers. Αυτή τη στιγμή, δεν υπάρχει κάποιος δημόσιος προσβάσιμος, online crawler στο δίκτυο Gnutella, αφού ο τελευταίος που φιλοξενούταν από τον Lime Wire LLC έχει αποσυρθεί.

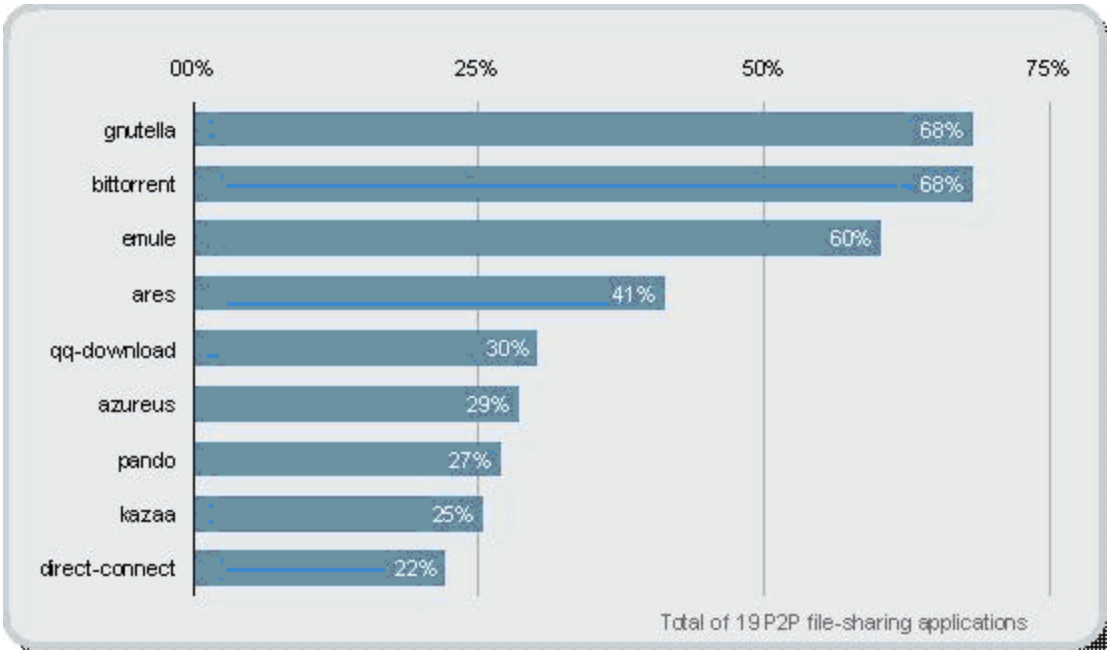
# Κεφάλαιο 5

# Σύγκριση BitTorrent - Gnutella

## 5.1 Εισαγωγή

Σύμφωνα με το [1], οι BitTorrent και Gnutella clients αποτελούν τους πιο διαδεδομένους αυτήν την στιγμή, αφού το 68% των χρηστών τα χρησιμοποιεί για τον διαμοιρασμό αρχείων, όπως απεικονίζεται στο Σχήμα 5.1.

Στην συνέχεια αυτής της ενότητας διεξάγουμε μια συγκριτική μελέτη των πρωτοκόλλων / clients που μελετήσαμε στα προηγούμενα κεφάλαια.




**Σχήμα 5.1** – Δημοτικότητα clients [1].

## 5.2. Τοπολογία δικτύου

Στο εδάφιο αυτό συγκρίνουμε το δίκτυο BitTorrent με τις δυο αρχιτεκτονικές δικτύων (v0.4 και v0.6) του Gnutella.

Το δίκτυο BitTorrent έχει ως κύριο χαρακτηριστικό την ύπαρξη του tracker. Λόγω της παρουσίας του, το δίκτυο κατατάσσεται στην κατηγορία των ημικεντρικοποιημένων P2P δικτύων. Δεν αποκαλείται κεντρικοποιημένο, όπως το Napster, για το λόγο ότι κάθε ομότιμος είμαι ικανός να θέσει σε λειτουργία έναν τέτοιο server. Το BitTorrent είναι πιο απλό στην υλοποίηση και εξαιτίας της κεντρικής οντότητας υπάρχει σφαιρική άποψη για ολόκληρο το P2P δίκτυο. Είναι πιο ευέλικτο σε θέματα βελτιστοποίησης του δικτύου επικάλυψης, ενώ οι τροποποιήσεις μπορούν να εφαρμοστούν ευκολότερα, χρησιμοποιώντας την σφαιρική γνώση του P2P δικτύου [38].

Αντίθετα, το Gnutella v0.4 αποτελεί ένα πλήρως αποκεντρικοποιημένο δίκτυο. Όλες οι λειτουργίες του δικτύου καθορίζονται μόνο μεταξύ των ομοτίμων και του πρωτοκόλλου, χωρίς την παρουσία κάποιας κεντρικής οντότητας. Διαθέτει μεγαλύτερο βαθμό αυτονομίας ελέγχου των δεδομένων και των πόρων του, όμως λόγω έλλειψης σφαιρικής άποψης στο επίπεδο του συστήματος, είναι δύσκολο να προβλεφθεί η συμπεριφορά του [30].

Όσον αφορά την δομή του δικτύου, τόσο το BitTorrent όσο και το Gnutella v0.4 ανήκουν στη κατηγορία των αδόμητων δικτύων, αφού δεν υπάρχει μία αυστηρά ορισμένη δομή σε αυτά τα δίκτυα. Η τοποθέτηση των δεδομένων είναι ανεξάρτητη από το overlay δίκτυο και οι ομότιμοι συνδέονται απευθείας ο ένας με τον άλλον, χωρίς να έχουν καμία πληροφορία για τα δεδομένα των υπόλοιπων ομοτίμων. Επίσης, στα δίκτυα αυτά οι ομότιμοι είναι ισοδύναμοι μεταξύ τους και μπορούν να συνδέονται και να αποσυνδέονται από το δίκτυο ανά πάσα στιγμή. Οι συνδέσεις των ομοτίμων γίνονται μέσω του πρωτοκόλου TCP.

Όσον αφορά το δίκτυο Gnutella v0.6, ανήκει στην κατηγορία των υβριδικών P2P συστημάτων. Το γεγονός αυτό οφείλεται στην παρουσία των ultrapeers, οι οποίοι έχουν κάποιες επιπλέον αρμοδιότητες συγκριτικά με τους ομότιμους. Ο ρόλος των ultrapeers και των trackers είναι τελείως διαφορετικός και αυτό οφείλεται στην φύση των πρωτοκόλλων τα οποία εξυπηρετούν. Όμως αποτελούν και τα δυο δυναμικές κεντρικές οντότητες για τα συστήματα τους. Στο σημείο αυτό λοιπόν το δίκτυο

Gnutella v0.6 μοιάζει με το BitTorrent. Άρα και τα δυο αυτά συστήματα δεν είναι πλήρως αποκεντρικοποιημένα.

Σε ότι αφορά τους ομότιμους, ενώ στο BitTorrent είναι απολύτως ισοδύναμοι, στο Gnutella v0.6 δεν συμβαίνει το ίδιο. Επίσης, κάθε ομότιμος στο BitTorrent μπορεί να συνδεθεί με έναν άλλον αν φυσικά ανήκουν στο ίδιο σμήνος, ενώ στο Gnutella v0.6 οι ομότιμοι δεν συνδέονται απευθείας μεταξύ τους παρά μόνο με κάποιον ultrapeer.

Το BitTorrent και το Gnutella v0.6 αποτελούν αδόμητα δίκτυα. Το BitTorrent σχηματίζει ένα διαφορετικο δίκτυο επικάλυψης για κάθε αρχείο, ενώ το Gnutella σχηματίζει ένα μεγάλης έκτασης.

## 5.3 Bootstrapping

Bootstrapping, όπως αναφέραμε σε προηγούμενα κεφάλαια, είναι η διαδικασία με την οποία οι ομότιμοι εντάσσονται στο εκάστοτε δίκτυο.

Τόσο το BitTorrent όσο και το Gnutella (v0.4 και v0.6) χρειάζονται τη βοήθεια ενός προγράμματος client ώστε οι ομότιμοι να μπορούν να ενταχθούν στο δίκτυο. Χωρίς αυτό, δε θα ήταν δυνατή η επικοινωνία μεταξύ των ομοτίμων και κατ' επέκταση η ανταλλαγή των δεδομένων. Ωστόσο υπάρχουν σημεία στον ακριβή τρόπο ένταξης των ομοτίμων στο δίκτυο που διαφοροποιούν τα δύο αυτά συστήματα.

Στο BitTorrent ο ομότιμος πρέπει να συνδεθεί με έναν tracker. Σε περίπτωση που ο tracker τεθεί εκτός λειτουργίας, οι ομότιμοι θα μπορούν να ανταλλάσουν δεδομένα, όμως θα είναι αδύνατο για ένα νέο ομότιμο να συμμετάσχει στο συγκεκριμένο σμήνος. Επίσης, μετά τη σύνδεση και την εγγραφή στο tracker, θα δοθεί στον ομότιμο μία τυχαία λίστα με τους γείτονές του, με τους οποίους θα πραγματοποιήσει τις συνδέσεις και την ανταλλαγή των δεδομένων.

Στο Gnutella, με την εφαρμογή του προγράμματος πραγματοποιείται άμεση σύνδεση του ομοτίμου με έναν τουλάχιστον ενεργό κόμβο, όπου στην περίπτωση του Gnutella v0.6 θα είναι ένας ultrapeer. Πρόσθετα, στο Gnutella γίνεται προσπάθεια από τον ίδιο τον ομότιμο (μέσω του client) ώστε να εγκαθιδρύσει και άλλες συνδέσεις, μετά την σύνδεση με τον ενεργό κόμβο, έτσι ώστε να ανακαλύψει τους υποψήφιους γείτονές του.

## 5.4. Δεδομένα

Ο βασικός λόγος ύπαρξης των P2P συστημάτων είναι η ανταλλαγή δεδομένων. Για τον λόγο αυτό εξετάζουμε τους μηχανισμούς δημοσίευσης, αναζήτησης και αποθήκευσης και στις δυο περιπτώσεις.

### 5.4.1. Δημοσίευση

Τα δίκτυα BitTorrent και Gnutella έχουν διαφορετικό μηχανισμό δημοσίευσης δεδομένων.

Στο BitTorrent η δημοσίευση των δεδομένων γίνεται με τη δημιουργία του .torrent αρχείου, το οποίο στη συνέχεια καταχωρείται σε ένα tracker server. Η δημοσίευση αρχίζει απλά με την εφαρμογή του server. Το κάθε .torrent αρχείο είναι συγκεκριμένο για τα δεδομένα που δημοσιεύονται.

Αντίθετα, στο Gnutella έχουμε έχουμε διαφοροποίηση ανάλογα με την έκδοση. Το Gnutella v0.4 κάθε κόμβος διατηρεί τοπικά τη λίστα με τα αρχεία που μοιράζεται, αφού δεν υπάρχει κάποια μορφή κεντρικής οντότητας στο δίκτυο αυτό. Ο Στο Gnutella v0.6 οι leaf-peers παρέχουν μια λίστα με τα ονόματα των αρχείων που μοιράζονται σε κάθε ultrapeer που έχουν συνδεθεί [30].

### 5.4.2. Μηχανισμός ανάζητησης

Το πρωτόκολλο BitTorrent είναι ένα πρωτόκολλο ανταλλαγής αρχείων καθώς εστιάζει στην αποτελεσματική προσκόμιση (fetching) και όχι στην αναζήτηση (searching) δεδομένων. Ο ομότιμος θα πρέπει να επισκεφθεί ιστοσελίδες με αποθηκευμένα .torrent αρχεία. Η αναζήτηση γίνεται ουσιαστικά μέσω αυτών των σελίδων όπου ο χρήστης μπορεί να αναζητήσει το επιθυμητό αρχείο. Έτσι, μέσω ενός BitTorrent client, θα δοθεί μια τυχαία λίστα με τους ομότιμους που θα έχουν το αρχείο. Η διαδικασία της αναζήτησης, λόγω παρουσίας του tracker, λειτουργεί με αποδοτικότερο και ασφαλέστερο τρόπο, σε σχέση με το Gnutella. Η λίστα με τους ομοτίμους που παρέχεται από τον tracker στον χρήστη, αποτελεί εγγύηση ότι ο ενδιαφερόμενος ομότιμος θα λάβει τα δεδομένα που επιθυμεί, και όχι κάποια τυχόν εσφαλμένα αρχεία. Αντίθετα στο Gnutella, δεν υπάρχει καμία εγγύηση [39].

Στο Gnutella v0.4 κάθε ομότιμος υποβάλει ερώτημα το οποίο προωθείται στους γείτονές του, που στη συνέχεια το προωθούν στους γειτονικούς τους κόμβους κοκ., εώς ότου μηδενιστεί η τιμή στο πεδίο TTL. Οι ομότιμοι που έχουν το αρχείο που ζητείται, στέλνουν ανάλογο μήνυμα στον αποστολέα. Στο Gnutella v0.6, η αναζήτηση των δεδομένων από τους ομοτίμους γίνεται μέσω των ultrapeers με τους οποίους συνδέονται. Γενικά, στο Gnutella γίνεται προσπάθεια μέσω των ομοτίμων και του δικτύου ώστε να βρεθεί το “κατάλληλο” αρχείο. Κάθε αναζήτηση έχει σαν αποτέλεσμα τη δημιουργία μεγάλου όγκου μηνυμάτων. Υπάρχει μεγαλύτερη καθυστέρηση στις αναζητήσεις. Λειτουργεί όμως καλύτερα σε μικρής κλίμακας δίκτυα. Επίσης, για τη συντήρηση του δικτύου υπάρχουν τα ζωτικής σημασίας μηνύματα PING-PONG, τα οποία δημιουργούν προβλήματα στη κλιμάκωση του δικτύου. Αυτό οδηγεί σε μη αποδοτικές αναζητήσεις καθώς στο βωμό της κλιμάκωσης, και για να μην επιβαρυνθεί επιπλέον το σύστημα από μηνύματα, δεν γίνεται αναζήτηση και έλεγχος σε όλους τους κόμβους, καθώς υπάρχει και ο περιορισμός από τη τιμή του TTL, ζήτημα που δεν υφίσταται στο BitTorrent.

Συνεπώς , όσον αφορά το ζήτημα αναζήτησης δεδομένων, παρατηρούμε ότι τα πρωτόκολλα BitTorrent και Gnutella δεν παρουσιάζουν κάποια ομοιότητα.

### 5.4.3. Μηχανισμός λήψης και αποθήκευσης

Και στα δύο πρωτόκολλα οι ανταλλαγές δεδομένων γίνονται απευθείας μεταξύ των κόμβων. Αν για κάποιο λόγο διακοπεί η διαδικασία λήψης ενός αρχείου, υπάρχουν μηχανισμοί, και στις δύο περιπτώσεις, ώστε να συνεχιστεί η λήψη από το σημείο που διακόπηκε. Στο BitTorrent, σε αντίθεση με το Gnutella, για να μπορεί ένας ομότιμος να κάνει λήψη ενός αρχείου, θα πρέπει υποχρεωτικά να συνεισφέρει παράλληλα. Αυτό έχει σαν αποτέλεσμα την καλύτερη εκμετάλευση του εύρους ζώνης των ομοτίμων, το οποίο οδηγεί τις περισσότερες φορές σε αύξηση του ρυθμού λήψης (download rate) [40]. Δεν ισχύει κάτι αντίστοιχο στο Gnutella. Λόγω της διαδικασίας της πλυμμήρας των ερωτημάτων αλλά και των υπολοίπων μήνυμάτων γίνεται σπατάλη του μεγαλύτερου ποσοστού του εύρους ζώνης.

Το πρωτόκολλο BitTorrent έχει αναπτύξει μηχανισμούς όπως το chocking, optimistic unchocking και anti-snubbing, που σε συνδιασμό με τη στρατηγική tit-for-tat, οδηγούν στην αποδοτικότερη και πιο δίκαιη λήψη των δεδομένων [30].

Η ειδοποιός διαφορά ανάμεσα στα δύο πρωτόκολλα είναι ότι στο BitTorrent τα δεδομένα χωρίζονται σε κομμάτια και γίνεται ανταλλαγή αυτών μεταξύ των ομοτίμων που βρίσκονται στο σμήνος, σε αντίθεση με το Gnutella, όπου γίνεται λήψη ολόκληρου του αρχείου από άλλους ομοτίμους.

Το BitTorrent προσφέρεται για τη λήψη αρχείων μεγάλου μεγέθους, ενώ το Gnutella για τη λήψη αρχείων μικρού μεγέθους. Εξάλλου ένας από τους λόγους δημιουργίας του πρωτοκόλλου BitTorrent ήταν και η λήψη μεγάλης ποσότητας δεδομένων.

Επιπρόσθετα, στο BitTorrent σε περίπτωση όπου δεν υπάρχει seeder σε ένα συγκεκριμένο σμήνος, παγώνει η διαδικασία λήψης ολόκληρου του αρχείου. Στη χειρότερη περίπτωση, όπου δεν υπάρχουν καθόλου seeders σε ένα torrent, δε μπορεί ποτέ να ολοκληρωθεί η λήψη του αρχείου. Για να συμβεί κάτι αντίστοιχο στο Gnutella, θα πρέπει το αρχείο αυτό να μη συμπεριλαμβάνεται στα προς ανταλλαγή δεδομένα όλων των ομοτίμων.

## 5.5. Θέματα ασφάλειας

Γενικά στα P2P συστήματα η επίτευξη της ασφάλειας δεν είναι εύκολο έργο [30]. Οι κλασικοί τρόποι αντιμετώπισης, όπως τα firewall, για τη προστασία των δεδομένων και των συστημάτων από τους εισβολείς και τις επιθέσεις, όχι μόνο δε μπορούν να βοηθήσουν, αλλά πιθανόν και να εμποδίσουν την επικοινωνία μεταξύ των ομοτίμων.

Η έννοια της ασφάλειας είναι γενική και εμπεριέχει άλλες έννοιες όπως [30]:

- **διαθεσιμότητα** (availability) των δεδομένων και των ομοτίμων. Και στα δυο πρωτόκολλα η διαθεσιμότητα ενός περιεχομένου δεν μπορεί να προβλεφθεί και να εγγυηθεί [39]. Για παράδειγμα, στο δίκτυο BitTorrent όταν και ο τελευταίος seeder αποσυνδεθεί, το περιεχόμενο παύει να είναι διαθέσιμο. Η διάρκεια ζωής ενός περιεχομένου είναι πολύ σημαντικό θέμα, επειδή η διαθεσιμότητά του αποτελεί το κύριο μέλλημα των ομοτίμων στο δίκτυο και επιρρεάζει άμεσα την απόδοση του συστήματος. Στο BitTorrent, η διαθεσιμότητα ενός περιεχομένου εξαρτάται σε πολύ μεγάλο βαθμό και από την δημοτικότητά του. Για τη βελτίωση του ζητήματος αυτού στο BitTorrent έχουν αναπτυχθεί οι

multiple trackers και τα DHTs [41], που υπήρχαν στα δομημένα P2P δίκτυα.

- έλεγχος της **ακεραιότητας** (integrity) των δεδομένων που παρέχονται από τους ομότιμους. Στο Gnutella δεν υπάρχει εγγύηση για την ακεραιότητα των δεδομένων, καθώς η επαλήθευση του περιεχομένου μπορεί να γίνει αφού έχει ολοκληρωθεί η λήψη του. Ωστόσο σε κάποιους Gnutella clients π.χ. Limewire, επιτρέπεται σε μερικές περιπτώσεις η προεπισκόπηση του αρχείου. Αντίθετα στο BitTorrent, η ακεραιότητα είναι εξασφαλισμένη μέσω του ελέγχου (checksum με αλγόριθμο SHA1) που πραγματοποιείται για κάθε κομμάτι του αρχείου που ανταλλάσεται. Άν ένα κομμάτι είναι ημιτελές ή κατεστραμμένο, δε γίνεται η λήψη του [42].
- **ανωνυμία** (anonymity) των ομοτίμων. Και τα δυο πρωτόκολλα δε προσφέρουν ανωνυμία στους χρήστες τους, καθώς και στις δύο περιπτώσεις δεν αποκρύπτονται οι IP διευθύνσεις των ομοτίμων. Στο δίκτυο BitTorrent, οι ομότιμοι που συμμετέχουν σε ένα σμήνος είναι γνωστοί. Από τη στιγμή που το δίκτυο δημιουργεί απευθείας συνδέσεις μεταξύ των ομοτίμων, οι IP διευθύνσεις σχετίζονται με τη διαδικασία λήψης και είναι ορατές. Ομοίως στο Gnutella, είναι γνωστές οι διευθύνσεις των ομοτίμων που συμμετέχουν στην διαδικασία λήψης ενός αρχείου. Εξάλλου στο πακέτο QUERYHIT, υπάρχει το πεδίο IP address, δηλαδή η διέυθυνση του ομότιμου που έχει τα επιθυμητά αρχεία. Όσοι κόμβοι λαμβάνουν αυτό το πακέτο, μπορούν να γνωρίζουν εξ' αρχής τη διεύθυνση του ομοτίμου.
- έλεγχος της **αυθεντικότητας** (authenticity) των ομοτίμων. Και τα δυο πρωτόκολλα εμφανίζονται εκτεθημένα στο θέμα της αυθεντικότητας εφόσον δε διαθέτουν κάποιο μηχανισμό για την αξιολόγηση και επικύρωση των ομοτίμων. Στο Gnutella v0.6, το bootstrapping γίνεται από τους ultrapeers, εξαρτάται δηλαδή από τον εκάστοτε ultrapeer να εμποδίσει τη σύνδεση στο δίκτυο ενός ομότιμου που θεωρεί επικίνδυνο [42].
- μηχανισμούς κατά των **κακόβουλων λογισμικών**. Το BitTorrent, λόγω παρουσίας του tracker αλλά και του checksum που πραγματοποιείται, δεν αποτελεί εύκολο στόχο για ιούς, ad-ware,

spyware και pop-ups, σε αντίθεση με το Gnutella που εμφανίζεται πιο εκτεθημένο σε κακόβουλα λογισμικά [40]. Επιπλέον ο αλγόριθμος της πλημμύρας που χρησιμοποιείται βοηθάει στην περαιτέρω όξυνση και εξάπλωση του προβλήματος.

Γενικά το πρωτόκολλο Gnutella, συγκριτικά με το BitTorrent, δεν έχει δώσει ιδιαίτερη προσοχή στο θέμα της ασφάλειας [43] και αυτό στο βωμό της απλότητάς. Υποστηρίζεται [43] ότι πολλές από τις αδυναμίες του πρωτοκόλλου θα είχαν αποφευχθεί αν οι υπέυθυνοι είχαν λάβει υπ'όψην τους ότι είναι πιθανό κάποιοι ομότιμοι να θέλουν να υιοθετήσουν μία κακόβουλη συμπεριφορά.

## 5.6. Κλιμάκωση

Το Gnutella παράγει μεγάλο όγκο μηνυμάτων κατά τις αναζητήσεις και συνήθως δε μπορούν να χειριστούν ικανοποιητικά πάνω από μερικές χιλιάδες κόμβους. Αυτό έχει σαν αποτέλεσμα η αναζήτηση να γίνεται μόνο σε ένα μικρό μέρος του δικτύου. Με τη παρουσία των ultrapeers στο Gnutella v0.6, το πρόβλημα της κλιμάκωσης στο δίκτυο βελτιώθηκε κατά ένα μεγάλο βαθμό, αλλά όχι τόσο ώστε να μη θεωρείται πλέον πρόβλημα [44].

Από την άλλη μεριά, το BitTorrent δε περιέχει κάποιο ενσωματωμένο μηχανισμό αναζήτησης και θεωρείται αποδοτικότερο στο θέμα της κλιμάκωσης, όπου είναι αισθητή η διαφορά κυρίως στα μεγάλα αρχεία. Παρόλα αυτά δεν μπορεί να ειπωθεί ότι το BitTorrent δεν αντιμετωπίζει κανένα πρόβλημα με τη κλιμάκωση, καθώς ο tracker υποστηρίζει μόνο ένα περιορισμένο αριθμό ομοτίμων. Ο αριθμός αυτός μπορεί επιπλέον να περιοριστεί σημαντικά ανάλογα με το ποιά πρωτόκολλα χρησιμοποιεί (TCP/HTTP/HTTPS/UDP) [38].

Το θέμα της κλιμάκωσης αλλά και της ανθεκτικότητας στο BitTorrent, συνήθως επιλύεται με τους πολλαπλούς trackers [41]. Οι trackers τοποθετούνται σε groups ή επίπεδα με έναν tracker να επιλέγεται τυχαία για να εφαρμοστεί από το top tier. Αν όλοι οι tracker από το top tier αποτύχουν, το σύστημα δομιμάζει τους trackers από το αμέσως παρακάτω επίπεδο [44].

## 5.7. Free-riding

Το θέμα του free-riding είναι ένα πολύ σημαντικό ζήτημα στα P2P δίκτυα. Όταν οι ομότιμοι υιοθετούν μια «εγωιστική» συμπεριφορά, αυτό επιρεάσει κατά ένα πολύ μεγάλο ποσοστό την συνολική απόδοση του συστήματος. Η εξάλειψη όμως του free-riding δεν είναι καθόλου εύκολη υπόθεση. Οφείλεται και στην αλτρουιστική φύση των συστημάτων αυτών καθώς δεν είναι υποχρεωτική η παραμονή των ομοτίμων στο δίκτυο.

Το BitTorrent πλεονεκτεί έναντι του Gnutella στο θέμα του free-riding, καθώς έχει ενσωματώσει στο πρωτόκολλο του μηχανισμούς που παροτρύνουν τους ομότιμους να προσφέρουν περισσότερο στους υπολοίπους. Εξάλλου ένα από τα κύρια μελλήματα του πρωτοκόλλου ήταν και η πάταξη του free-riding. Παρ'όλα αυτά δεν έχει καταφέρει ακόμα να δώσει κίνητρα στους ομότιμους ώστε να μένουν στο δίκτυο και μετά την ολοκλήρωση της λήψης του αρχείου.

## 5.8. Ανθεκτικότητα

Στο Gnutella δεν υφίσταται μορφή κεντρικής οντότητας και συνεπώς δεν υπάρχει «αδύναμος κρίκος» όπως ο tracker στην περίπτωση του BitTorrent [38]. Αν τεθεί εκτός λειτουργίας, θα είναι αδύνατο για τους νέους ομότιμους να ενταχθούν στο δίκτυο ή για τους ήδη υπάρχοντες να επικοινωνήσουν μεταξύ τους. Το πρόβλημα όμως μπορεί να λυθεί με την χρήση πολλαπλών trackers [39]. Το Gnutella είναι εκ φύσεως ανθεκτικότερο στην ανοχή σφαλμάτων καθώς η απώλεια ενός ή ακόμα και πολλών ομοτίμων μπορεί εύκολα να αντισταθμιστεί, αλλά και στο Gnutella v0.6 η απώλεια ενός ή πολλών ultrapeers.

## Βιβλιογραφία

[1] http://www.paloaltonetworks.com/researchcenter/2009/07/traffic-analysis-p2p-found-92-of-the-time/

[2] H. Balakrishnan, M. Frans Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking Up Data in P2P Systems," Communications of the ACM, 2003.

[3] S. Androutsellis-Theotokis, D. Spinellis, “A Survey of Peer-to-Peer Content Distribution Technologies”, ACM Computing Surveys, 2004.

[4] Ν. Κρεμμύδας, “Επισκόπηση στα Συστήματα Ομότιμων Κόμβων”, Ε.Μ.Π., 2005.

[5] Computer για όλους, “Peer To Peer Computing”, Αριθμός Τεύχους: 202, 1/6/2001.

[6] Σ.Μαργαρίτη, “Data Management in Peer-to-Peer Systems”, Technical report, Πανεπιστήμιο Ιωαννίνων, 2005.

[7] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, “Peer-to-Peer Computing”, HP Laboratories 2002.

[8] http://en.wikipedia.org/wiki/BitTorrent_(protocol)

[9] http://www.codecon.org/

[10] http://www.slideshare.net/SridharBR/bit-torrent-protocol-report

[11] Αικατερίνη Γ. Δόκα, Μεταφορά Αρχείων σε Υπολογιστικά Πλέγματα με Χρήση Τεχνολογίας Ομότιμων Κόμβων, Διπλωματική Εργασία, Ε.Μ.Π., 2005.

[12] http://www.digitalnews.gr/torrents-—-θεωρία-πράξη-συμβουλές

[13] Σημειώσεις μαθήματος «Δίκτυα Υπολογιστών», Ε.Μ.Π., 2009.

[14] http://wiki.theory.org/BitTorrentSpecification#Choking_and_Optimistic_Unchoking

[15] http://www.bittorrent.org/bittorrentecon.pdf

[16] http://en.wikipedia.org/wiki/Comparison_of_BitTorrent_clients

[17] http://www.aol.com/

[18] http://www.nullsoft.com/

[19] http://www.winamp.com/

[20] http://www.gnu.org/licenses/gpl.html

[21] http://www.isee.gr/issues/05/special/index.html#3

[22] http://en.wikipedia.org/wiki/Opennap

[23] http://en.wikipedia.org/wiki/LimeWire

[24] http://www.wired.com/gadgets/portablemusic/news/2002/03/50858

[25] http://en.wikipedia.org/wiki/MLDonkey

[26] K. Sripanidkulchai, The popularity of Gnutella queries and its implications on scalability, 2001.

[27] I.Αικατερινίδης, Ανάπτυξη συστημάτων Δημοσίευσης Συνδρομών σε Δομημένα Δίκτυα Ομότιμων Εταιριών, Διδακτορική Διατριβή, 2008.

[28] Matei Ripeanu, Ian Foster, "Mapping the Gnutella Network: Macroscopic Properties of Large-Scale Peer-to-Peer Systems", 2008.

[29] "The Gnutella Protocol Specification v0.41", http://www.clip2.com

[30] Santosh Kumar Shaw, "Topology Formation and Replication Strategies for Gnutella Network", 2008.

[31] D.Defigueiredo, A.Garcia, B.Kramer, "Analysis of Peer-to-Peer Network Security using Gnutella", Department of Computer Science, Univercity of California, Berkeley.

[32] I.Varsandan, "A Peer to Peer Network for Distributed Case Based Reasoning", Jacobs University Bremen, 2007.

[33] A.Asvanund, R.Krishnan, M.Smith, R.Telang, "Interest-Based Self-Organizing Peer-to-Peer Networks: A Club Economics Approach", Carnegie Mellon University, 2005.

[34] http://wiki.limewire.org/index.php?title=Pong_Caching

[35] R.Motta, W. Nienaber, J.Jenkins, "Gnutella: Integrating Performance And Security In Fully Decentralized P2P Models", Florida State University, 2003.

[36] http://en.wikipedia.org/wiki/Gnutella

[37] http://wiki.theory.org/BitTorrentSpecification

[38] J.Li, "On peer-to-peer (P2P) content delivery", 2007.

[39] K.Lua, J.Crowcroft, M.Pias, R.Sharma, S.Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes", IEEE COMMUNICATIONS SURVEY AND TUTORIAL, 2004.

[40] http://www.websitedesignpartners.com/WhatisbittorrentandhowdoIuseit.html

[41] G.Neglia, G.Reina, H.Zhang, D.Towsley, A.Venkataramani, J.Danaher, "Availability in BitTorrent Systems", University of Palermo and Massachusetts.

[42] http://cseweb.ucsd.edu/~rmotta/my_papers/Gnutella_Integrating_Perfor

[43] D.Zeinalipour-Yazti, “Exploiting the Security Weaknesses of the Gnutella Protocol”, Department of Computer Science, University of California.

[44] http://en.wikipedia.org/wiki/BitTorrent_(protocol)

[45] http://en.wikipedia.org/wiki/Port_forwarding

[46] A.Crespo and H.Garcia-Molina, “Semantic overlay networks for p2p systems”, Technical report, Computer Science Department, Stanford University, 2002.

[47] http://kennethhunt.com/archives/000198.html

[48] D.Stutzbach, R.Rejaie, “Capturing Accurate Snapshots of the Gnutella Network”, Department of Computer and Information Science, University of Oregon.

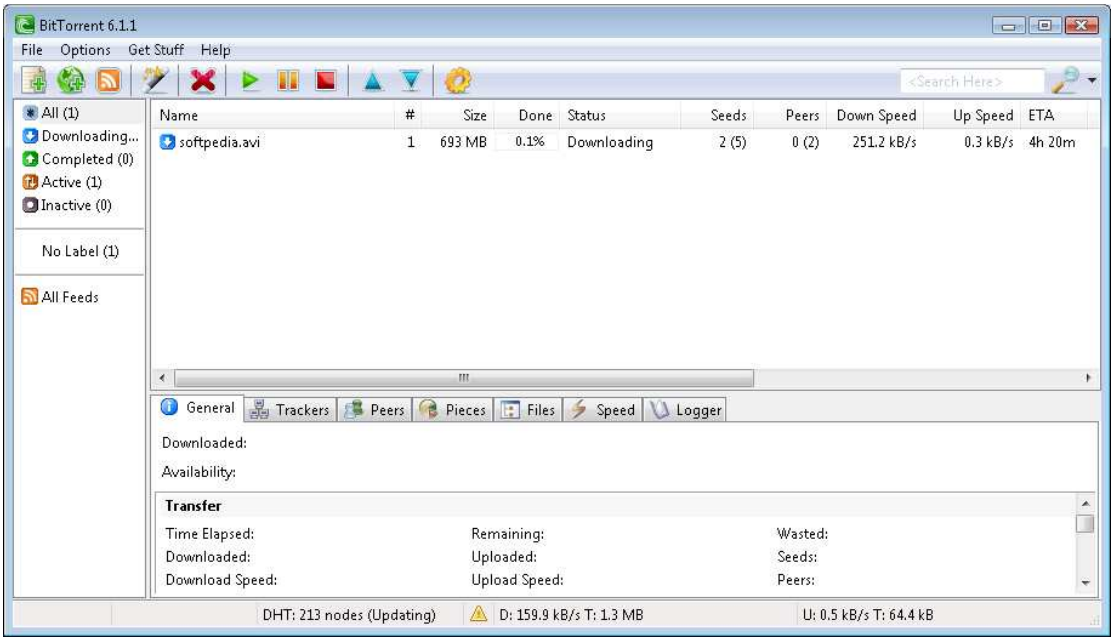
[49] http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html

# Παράρτημα Α’

## Α1. Σύντομη παρουσίαση προγράμματος BitTorrent

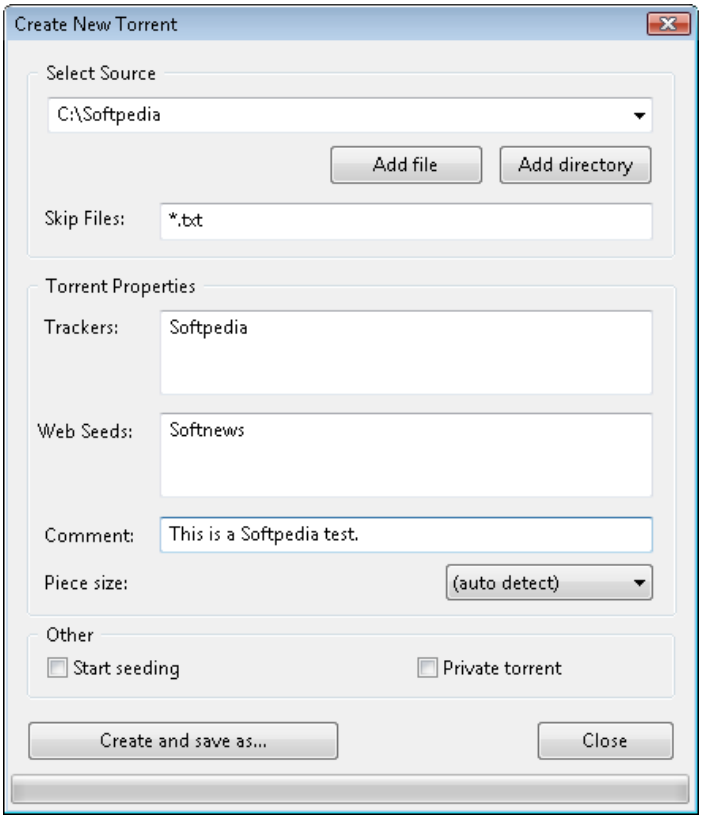
Στο εδάφιο αυτό παρουσιάζουμε το πρόγραμμα BitTorrent. Το κατεβάσαμε ως ελεύθερο λογισμικό (freeware) από το διαδίκτυο και το εγκαταστήσαμε τοπικά στον υπολογιστή μας, για να διεξάγουμε μια δοκιμαστική χρήση. Η έκδοση που χρησιμοποιήσαμε είναι η v.6.1.1. .

Από την στιγμή που ο χρήστης κατεβάσει και εγκαταστήσει το πρόγραμμα στον υπολογιστή του, ξεκινώντας το θα λάβει την εικόνα του Σχήματος Α.1. Η διεπαφή αυτή παρέχει την δυνατότητα στον χρήστη να εποπτεύει την πρόοδο των αρχείων που κατεβάζει.




**Σχήμα Α.1** – Κεντρικό παράθυρο προγράμματος BitTorrent.

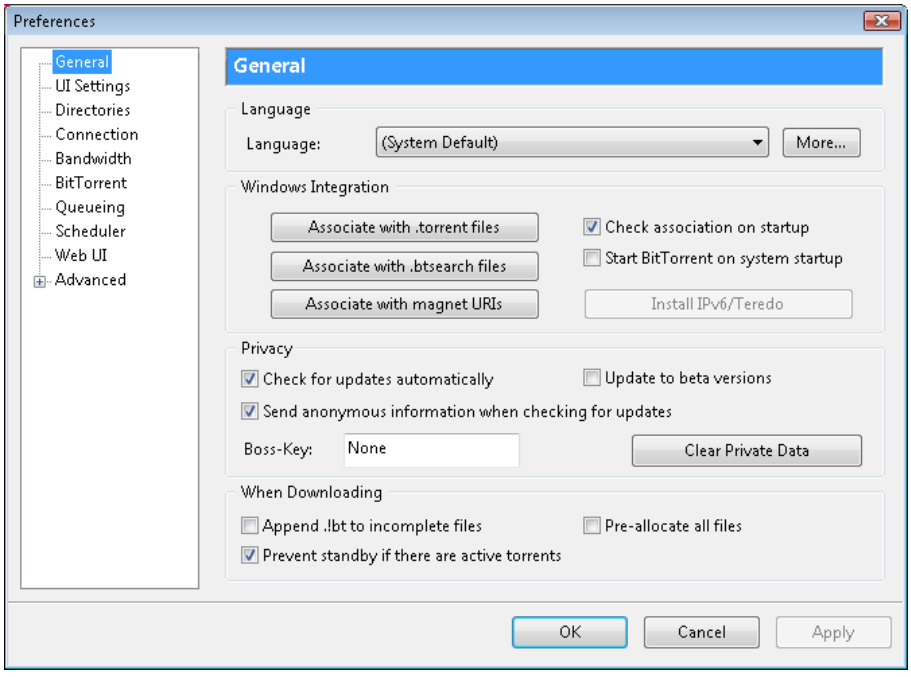
Το πρόγραμμα BitTorrent παρέχει στον χρήστη την δυνατότητα να δημιουργήσει τα δικά του αρχεία torrent, επιλέγοντας τον κατάλογο με το επιθυμητό υλικό, τους trackers και τους web seeds. Στην περίπτωση που το torrent πρόκειται να αναρτηθεί σε site ιδιωτικής κοιτότητας, μπορεί να χαρακτηριστεί ως private torrent. Η διεπαφή που παρέχει αυτή την δυνατότητα στον χρήστη απεικονίζεται στο Σχήμα Α.2.




**Σχήμα Α.2** – Φόρμα δημιουργίας νέου torrent.

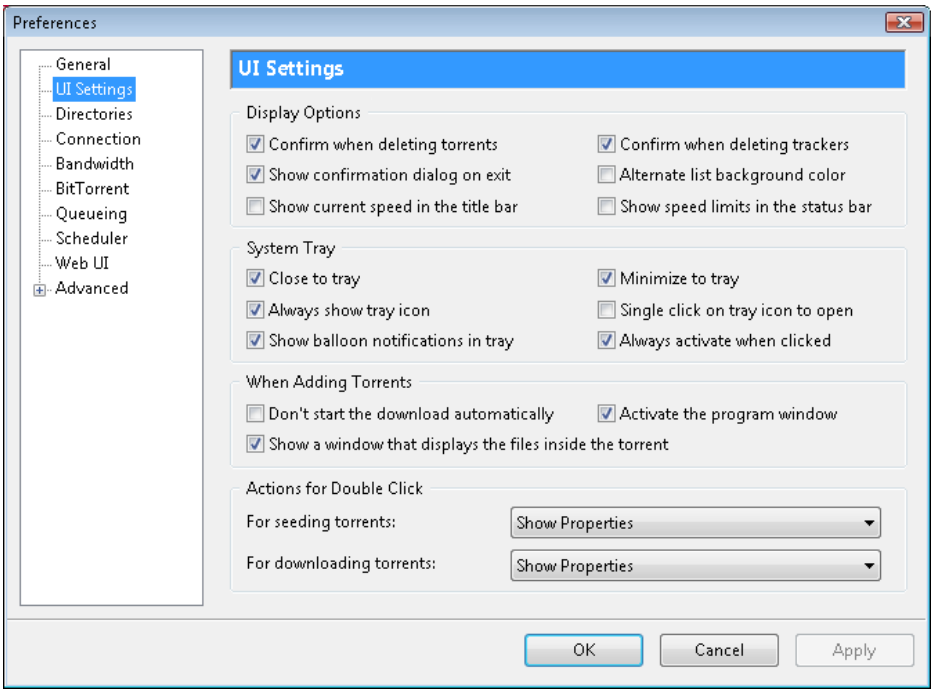
Το πρόγραμμα BitTorrent είναι πλήρως παραμετροποιήσιμο, επιτρέποντας στον χρήστη να καθορίζει τις επιθυμητές ρυθμίσεις. Οι ρυθμίσεις αυτές αφορούν:

- ✓ **ρυθμίσεις γενικού περιεχομένου**



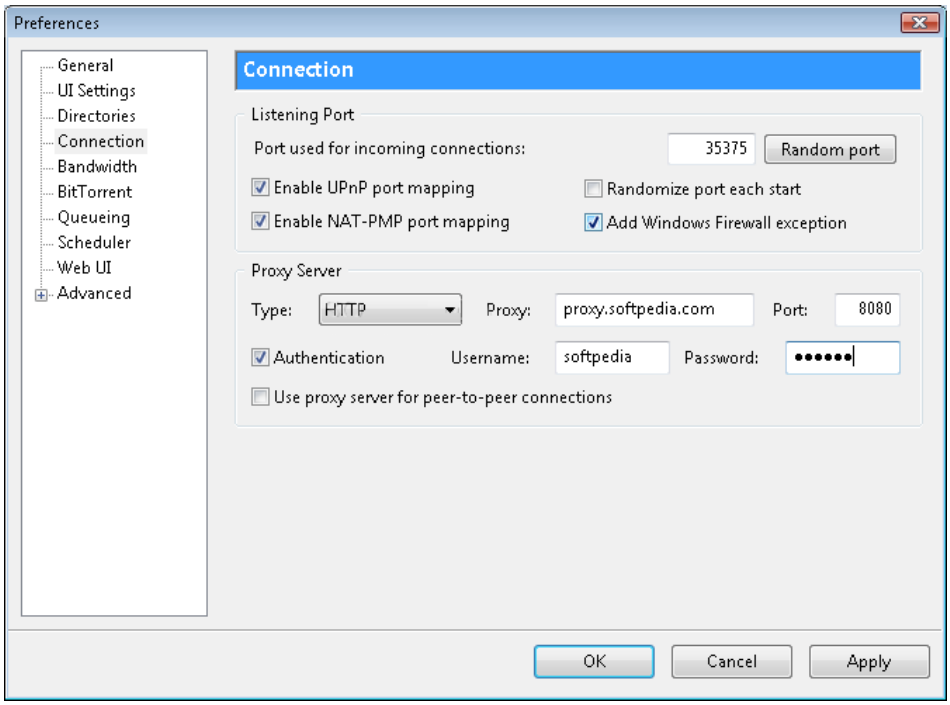

**Σχήμα Α.3** – Φόρμα ρυθμίσεων γενικού περιεχομένου.

✓ **ρυθμίσεις γραφικού περιβάλλοντος**




**Σχήμα Α.4** – Φόρμα ρυθμίσεων γραφικού περιβάλλοντος.

✓ **ρυθμίσεις σύνδεσης**

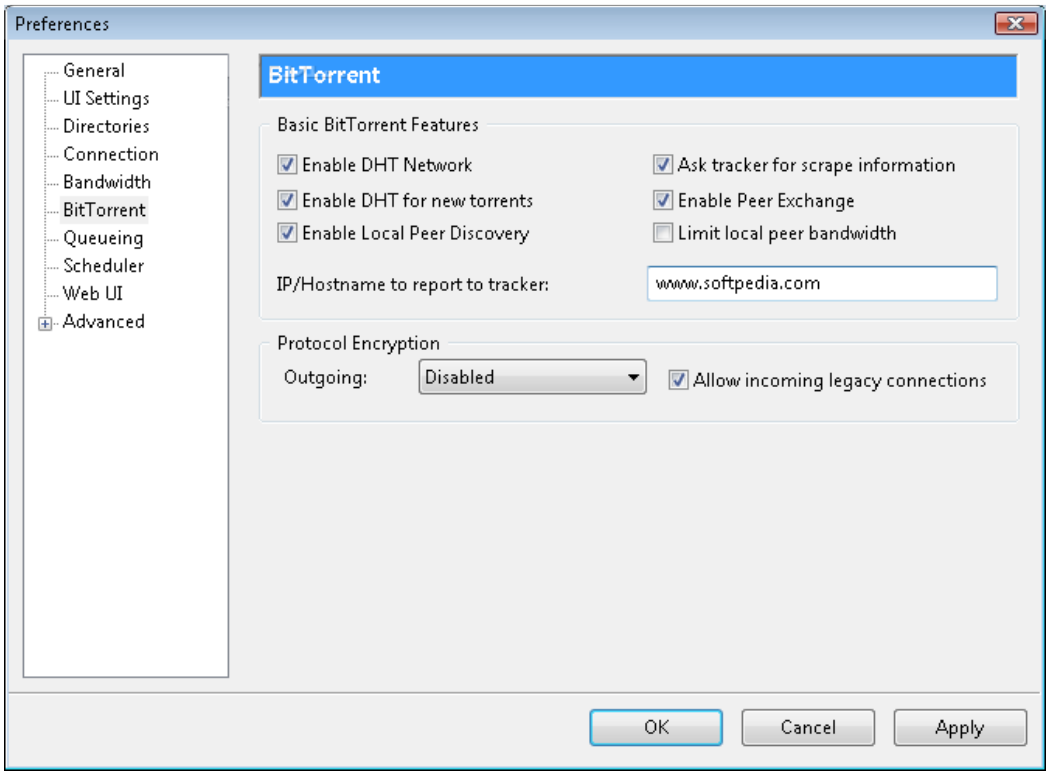



**Σχήμα Α.5** – Φόρμα ρυθμίσεων σύνδεσης.

Στην διεπαφή αυτή γίνεται το port forwarding [45]. Είναι μια απαραίτητη διαδικασία που πρέπει να κάνουν όσοι έχουν router για να αυξήσουν την ταχύτητα των torrents. Με την τεχνική αυτή επιτρέπονται οι επικοινωνίες από εξωτερικά hosts με υπηρεσίες

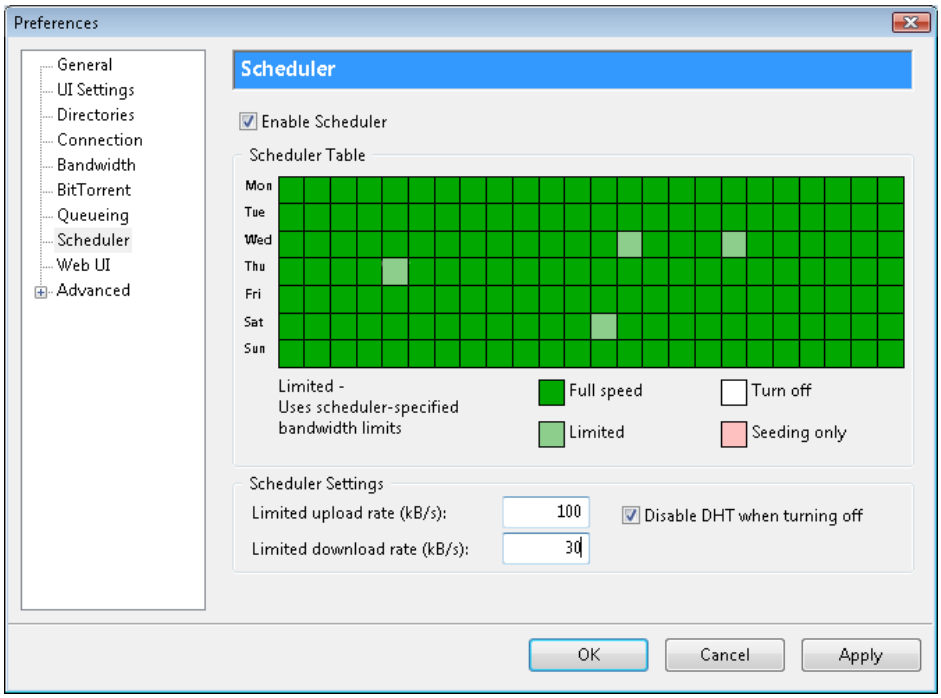
που παρέχονται μέσα από ένα ιδιωτικό τοπικό δίκτυο. Ουσιαστικά ανοίγουμε την πόρτα στην οποία δουλεύει το BitTorrent και να την κάνουμε forward στην IP του router.

- ✓ **ρυθμίσεις πρωτοκόλλου BitTorrent**



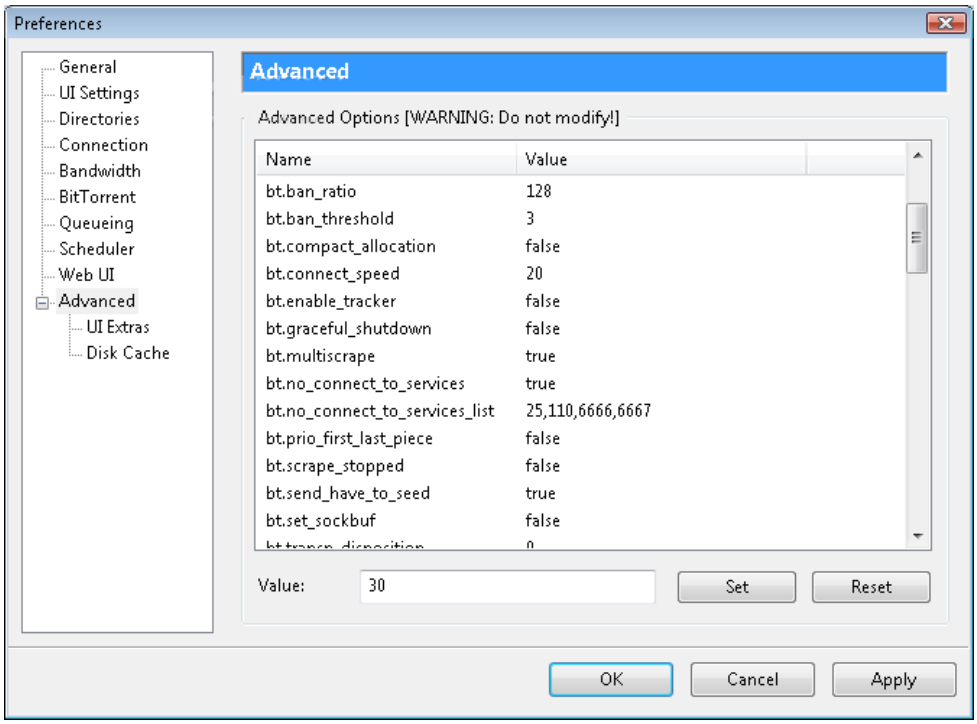

**Σχήμα Α.6** – Φόρμα ρυθμίσεων πρωτοκόλλου BitTorrent.

- ✓ **ρυθμίσεις χρονοπρογραμματισμού λειτουργίας**

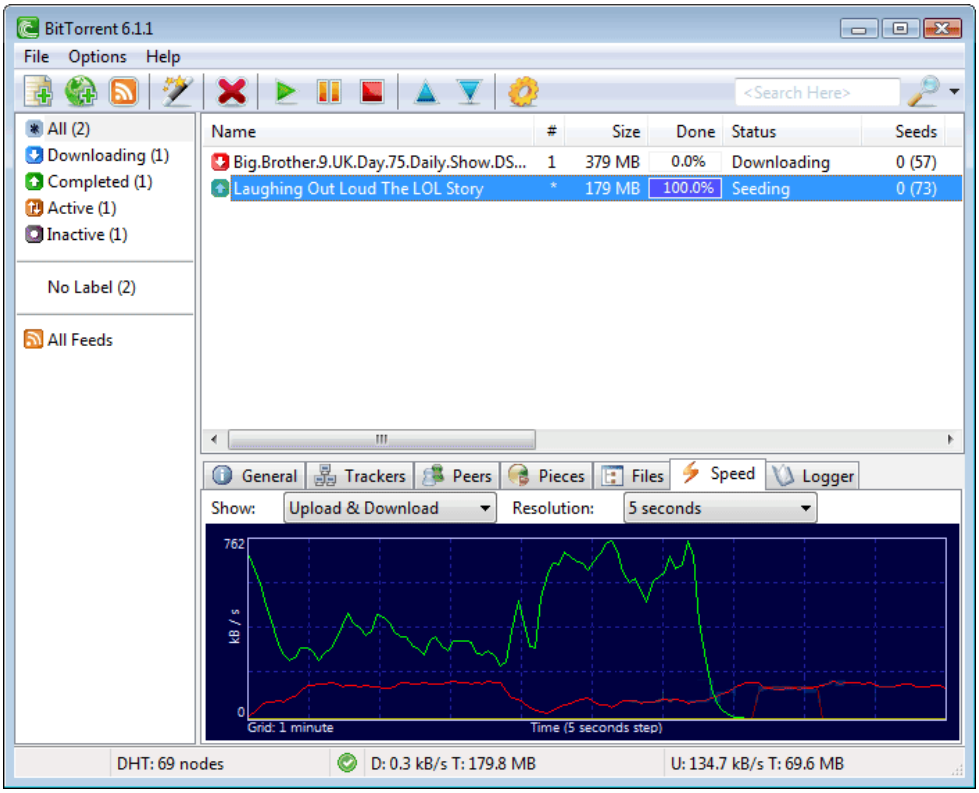



**Σχήμα Α.7** – Φόρμα ρυθμίσεων χρονοπρογραμματισμού λειτουργίας.

✓ **ρυθμίσεις παραμέτρων πρωτοκόλλου BitTorrent**




**Σχήμα Α.8** – Φόρμα ρυθμίσεων παραμέτρων πρωτοκόλλου BitTorrent.




**Σχήμα Α.9** – Φόρμα παρακολούθησης προόδου κατεβάσματος torrents.

Καθορίζοντας τις ρυθμίσεις λειτουργίας, το πρόγραμμα είναι διαθέσιμο για κατέβασμα αρχείων. Χρειάζεται απλά να επιλέξει ο χρήστης το επιθυμητό torrent και το πρόγραμμα αναλαμβάνει τα υπόλοιπα. Στην εικόνα του Σχήματος Α.9 παρατηρούμε ότι το πρώτο torrent που κατεβάζουμε δεν έχει seeds για να κατεβάσει, ενώ το δεύτερο torrent έχει κατέβει και το διαθέτουμε με την σειρά μας στους υπόλοιπους χρήστες.