Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Τμήμα Εφαρμοσμένης Πληροφορικής και Πολυμέσων

<u>Διπλωματική Εργασία</u>

Bittorrent vs Gnutella: Συγκριτική μελέτη και πειραματισμός με τα δύο σημαντικότερα διομότιμα συστήματα.

Βραχνάκης Εμμανουήλ

Ημ. Παράδοσης _._.2009

Ηράκλειο 2009

ΠΕΡΙΕΧΟΜΕΝΑ

Περίληψη

Τα διομότιμα συστήματα εμφανίστηκαν για πρώτη φορά το 1999, όταν έναν 18χρονος φοιτητής με το όνομα Shawn Fanning, άλλαξε για πάντα τη μουσική βιομηχανία εισάγωντας ένα νέο τρόπο ανταλλαγής αρχείων. Η ιδέα του ήταν απλή: ένα πρόγραμμα που θα επέτρεπε στους χρήστες υπολογιστών να μοιράζονται αρχεία, κυρίως μουσικής, μέσω ενός κεντρικού υπολογιστή αρχείων. Το πρόγραμμα ονομάστηκε Napster. Παρόλο που η υπηρεσία αυτή, έκλεισε μετά από δικαστική απόφαση, το Napster άνοιξε το δρόμο για άλλα είδη διομότιμων συστημάτων όπου θα ήταν πολύ δύσκολο να ελεγθούν. Τα κυριότερα εκ των οποίων είναι το Bittorrent και το Gnutella.

Στη παρούσα έρευνα θα παρουσιάσουμε λεπτομερώς το τρόπο με τον οποίο τα δύο αυτά συστήματα λειτουργούν. Στη συνέχεια θα ακολουθήσει μία συγκριτική μελέτη αυτών, αναφέροντας τις ομοιότητες και τις διαφορές τους. Τέλος θα πραγματοποιηθεί μια επίδειξη στο τρόπο λειτουργίας του πρωτοκόλου Bittorrent.

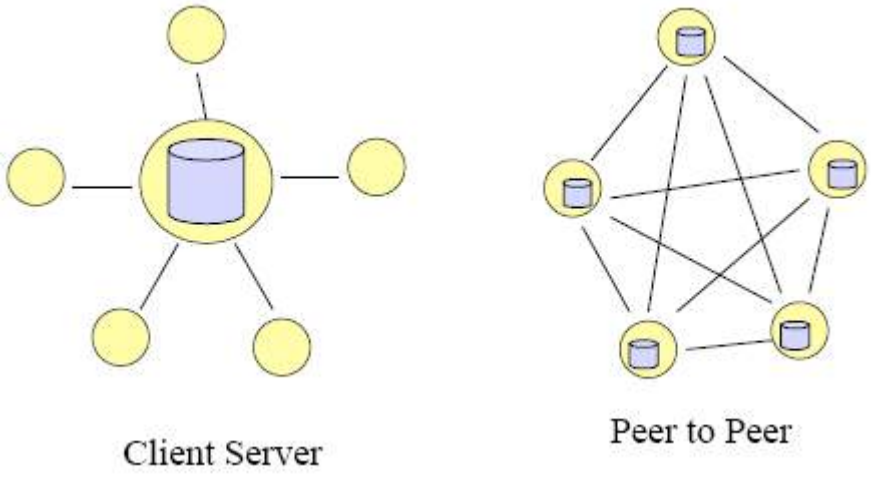
Abstract

Peer-to-Peer systems were presented for first time in 1999, when an 18-year-old college named Shawn Fanning, changed the music industry forever introducing a new way of exchange of files. His idea was simple: a program that allowed computer users to share files, specifically music, through a centralized file server. The program was called Napster. Although the service was shut down by court order, it paved the way for other types of peer-to-peer systems, which have been much harder to control. The main of them are Bittorrent and Gnutella.

In this research we will present in detail the way in which these two systems operate. Then followed a comparative study of these, indicating the similarities and differences. Finally, will be realised a demonstration in the way of operation of Bittorrent's protocol.

ΕΙΣΑΓΩΓΗ

Οι περισσότερες υπηρεσίες που παρέχονται στο διαδίκτυο, ακολουθούν το μοντέλο Πελάτη-Εξυπηρετητή (Client-Server) π.χ. HTTP,FTP,DNS και POP3[1]. Κατά το μοντέλο αυτό, υπάρχει ένας κεντρικός υπολογιστής (ή σύστημα υπολογιστών), server, που παρέχει υπηρεσίες σε άλλους υπολογιστές (ή συστήματα υπολογιστών) που ονομάζονται clients, μέσω ενός δικτύου υπολογιστών. Η επικοινωνία client - server γίνεται μέσω του πρωτοκόλλου TCP/IP. Τα κύρια πλεονεκτήματα του μοντέλου είναι η κεντρικοποιημένη διαχείρηση δεδομένων καθώς και η ασφάλεια αυτών, καθιστώντας έτσι το μοντέλο ιδανικό για πολλές διαδικτυακές εφαρμογές (συστήματα τραπεζών,ηλεκτρονικό ταχυδρομείο, κτλ). Από την άλλη υπάρχουν και σημαντικά μειονεκτήματα, όπως το ότι χρειάζονται ακριβές υπολογιστικές υποδομές (π.χ. το youtube.com αναλώνει 25TB-250TB ανα μήνα,και με κόστος περίπου τα 5 εκ. δολλάρια το μήνα για το bandwidth), σε περίπτωση βλάβης του server,χάνεται η υπηρεσία (single point of failure), απαιτείται συνεχή διαχείρηση (administration), μπορεί επίσης να υπάρξει λογοκρισία και έλεγχος όπως συμβαίνει για παράδειγμα στο κράτος της Κίνας, σχετικά με τα αποτελέσματα των μηχανών αναζήτησης. Τέτοιου είδους μειονεκτήματα που μόλις αναφέρθηκαν, προσπαθούν να ξεπεραστούν με τη δημιουργία και χρήση των διομότιμων συστημάτων ή αλλιώς Peer-to-Peer Systems (συντομ. P2P).



Σχημ. 1 Τοπολογίες των δικτύων Client-Server και P2P.

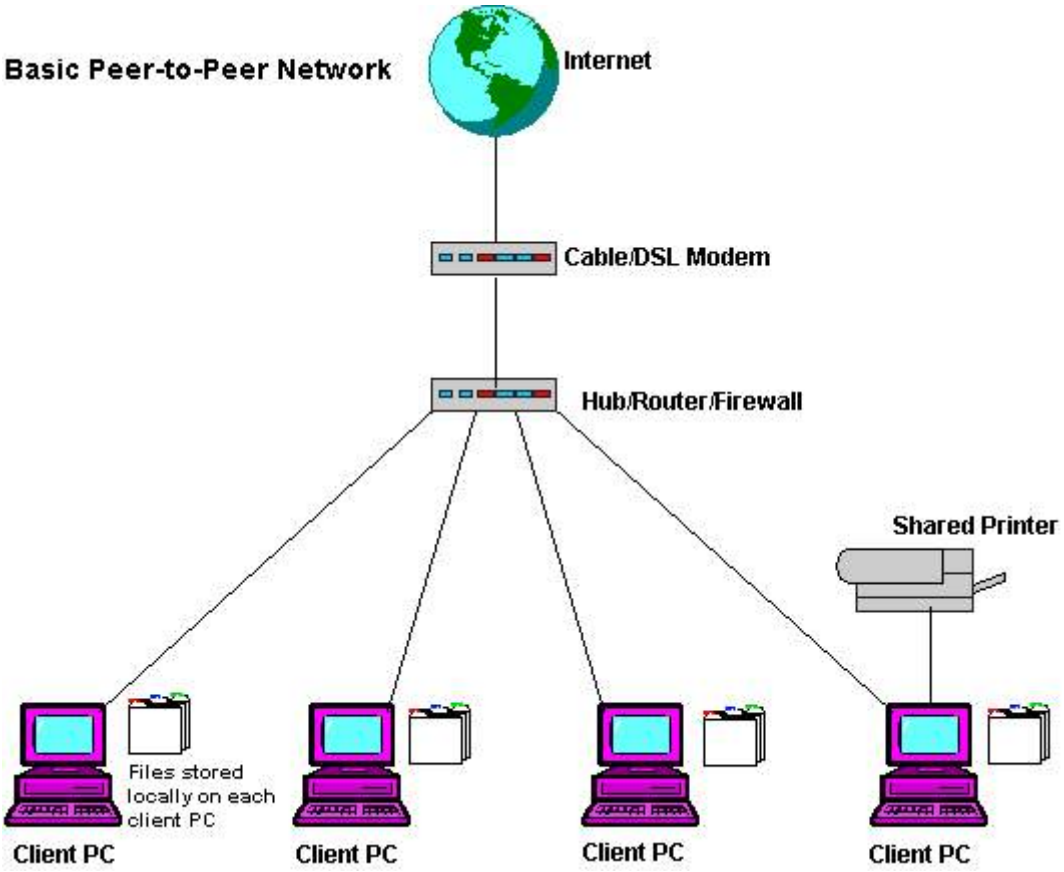
ΚΕΦΑΛΑΙΟ 1ο

1.1. Ορισμός των P2P συστημάτων

Ο ορισμός που δίνεται από τους συγγραφείς είναι: "Τα p2p συστήματα είναι μια κλάση συστημάτων και εφαρμογών που χρησιμοποιούν κατανεμημένους πόρους για την εκτέλεση μιας συνάρτησης με μη κεντρικοποιημένο τρόπο". Ο ορισμός που έχει δώσει η ομάδα της Intel που ασχολείται με P2P δίκτυα έχει ως εξής : " Ένα P2P σύστημα είναι ο διαμοιρασμός υπολογιστικών πόρων και υπηρεσιών με άμεση ανταλλαγή μεταξύ των συστημάτων"[2]. Οι δύο αυτοί ορισμοί μπορούν να θεωρηθούν σχεδόν ταυτόσημοι. Η διαφορά είναι ότι στον δεύτερο τονίζεται η δυνατότητα διαμοιρασμού τον πόρων που υπάρχουν σε κάθε peer, ενώ στον πρώτο τονίζεται η δυνατότητα χρήσης των πόρων που προσφέρονται από διάφορα peer για την εκτέλεση μιας εργασίας το αποτέλεσμα της οποίας θα προκύψει με συνεργασία όλων αυτών των κόμβων (υπολογιστών).

1.2. Γενική περιγραφή λειτουργίας των P2P συστημάτων.

Με την ευρεία έννοια, η τεχνολογία P2P είναι μια κατανεμημένη αρχιτεκτονική δεδομένων που επιτρέπει σε μεμονωμένους υπολογιστές να συνδεθούν και να επικοινωνήσουν άμεσα με άλλους υπολογιστές . Μέσω αυτής της σύνδεσης, οι χρήστες υπολογιστών (γνωστοί ως "peers") μπορούν να μοιραστούν επικοινωνίες, επεξεργαστική ισχύ, και αρχεία δεδομένων. Όσον αφορά συγκεκριμένα τη διανομή αρχείων (file sharing), η τεχνολογία P2P επιτρέπει την "αποκεντρικοποιημένη" διανομή. Αντί να γίνεται αποθήκευση των αρχείων σε μια κεντρική τοποθεσία , όπως συνέβαινε στο μοντέλο client-server με το οποίο οι μεμονωμένοι υπολογιστές έπρεπε να συνδεθούν για να ανακτήσουν τα αρχεία, η τεχνολογία P2P επιτρέπει στους μεμονωμένους υπολογιστές να μοιράζονται άμεσα μεταξύ τους τα αρχεία που είναι αποθηκευμένα, τοπικά, στους επιμέρους υπολογιστές , όπως δείχνει το παρακάτω σχήμα.



Σχημ. 1.1. Δίκτυο P2P.

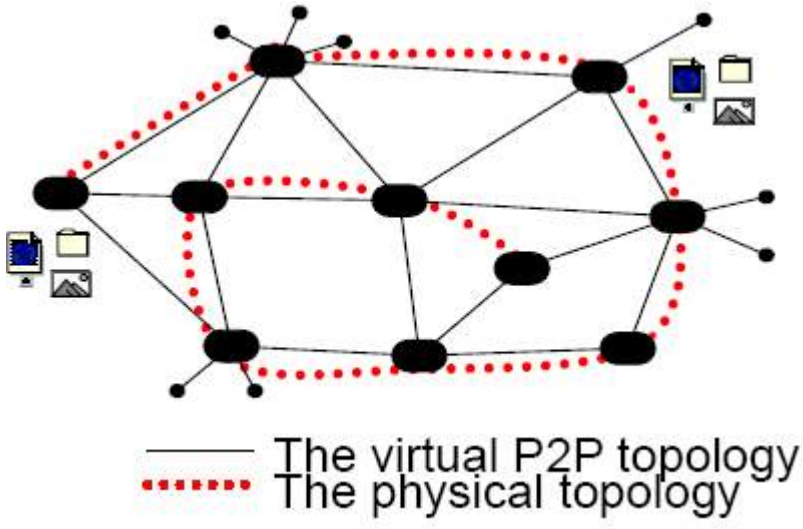
Ορισμένοι *ειδήμονες* σε αυτά τα ζητήματα έχουν υπογραμμίσει ότι, **από μόνα τους τα προγράμματα P2P δεν εκτελούν τη διανομή ή την αντιγραφή αρχείων, αλλά χρησιμοποιούν ένα πρωτόκολλο που διευκολύνει την επικοινωνία μεταξύ των δύο peers που επιθυμούν να μοιραστούν ή να αντιγράψουν δεδόμενα**. Οι χρήστες υπολογιστών μπορούν να μοιραστούν απεριόριστη ποσότητα κάθε τύπου αρχείων, όπως ακουστικού, τηλεοπτικού, λογισμικού, επεξεργασία κειμένου, και φωτογραφιών. Με την εξάλειψη της ανάγκης για ένα κεντρικό σημείο αποθήκευσης αρχείων, το P2P σύστημα επιτρέπει γρηγορότερες μεταφορές των αρχείων και τη συντήρηση του εύρους ζώνης δηλ. ουσιαστικά, της ικανότητας να διαβιβαστούν οι πληροφορίες πρός και από έναν υπολογιστή.

Συνοπτικά, οι εφαρμογές των διομότιμων συστημάτων, αφορούν[1]:

- Ανταλλαγή αρχείων. (π.χ. Napster, Gnutella, Bittorrent)
- Διαδικτυακή τηλεφωνία. (π.χ. Skype)
- Διαδικτυακά Παιχνίδια. (π.χ. Playstation Online Gaming)
- Πάταξη του Spam. (SpamNet)
- Instant Messaging. (IRC, MSN, Yahoo Messengers)
- Content Distribution Networks. (CorelCDN)
- P2P Web Cashing. (Squirrel)
- Application-Level Multicast. (Narada)

Το κάθε P2P δίκτυο είναι δομημένο πάνω από ένα **Overlay Network ή δίκτυο επικάλυψης** [1], και οι συμμετέχοντες σε αυτό ομότιμοι εγκαθιδρύουν TCP ή/και UDP συνδέσεις με άλλους κόμβους. Αυτό δημιουργεί ένα νοητό "virtual" γράφο διασύνδεσης . Πρόκειται ουσιαστικά για τεχνητά, ανεξάρτητα στρώματα επιλεγμένων κόμβων. Η απόδοση ενός peer-to-peer (P2P) δικτύου, εξαρτάται κατά πολύ, από τη καλή συνδεσμολογία της overlay τοπολογίας .

Να αναφέρουμε ότι **ο γράφος του δίκτυου επικάλυψης δεν αντιπροσωπεύει τις πραγματικές συνδέσεις μεταξύ των κόμβων** [1]. (βλ. παρακάτω σχήμα). Τα πλεονεκτήματα του Overlay Network είναι ότι εισαγάγει σημασιολογικές όψεις στο φυσικό δίκτυο και μειώνει το overflooding στο δίκτυο



Σχημ. 1.2. Φυσική και virtual τοπολογία.

Γενικά **τα P2P δίκτυα δεν χρησιμοποιούνται για να μεταφέρουν το περιεχόμενο**. Το P2P δίκτυο χρησιμοποιείται μόνο για αναζήτηση (lookup) του περιεχομένου, δηλ. για να ανακαλύψει σε ποιο κόμβο το ζητούμενο αρχείο είναι διαθέσιμο. Η μετάδοση έπειτα του περιεχομένου, γίνεται άμεσα μεταξύ του ομότιμου και του χρήστη που έκανε την αίτηση, συνήθως μέσω των πρόσθετων HTTP συνδέσεων. Για να είναι σε θέση να εισαχθεί στο ιδεατό δίκτυο, ένας νέος χρήστης πρέπει να ξέρει τουλάχιστον μια διεύθυνση IP ενός κόμβου που είναι ήδη συνδεμένος σε αυτό, όπως συμβαίνει π.χ. στο Gnutella. Διαφορετικά δεν θα μπορεί να συμμετέχει, δεδομένου ότι δεν θα είναι ικανός να εγκαταστήσει οποιεσδήποτε νέες συνδέσεις στο overlay δίκτυο .

1.3. Γενικά χαρακτηριστικά των P2P δικτύων.

Τα χαρακτηριστικά των P2P δικτύων, ασκούν σημαντική επίδραση στην αποτελεσματικότητα και την επέκταση των P2P συστημάτων και των εφαρμογών , τα σημαντικότερα εκ των οποίων είναι τα εξής[2]:

- Decentralized. Η αποκεντρικοποίηση είναι μια από τις σημαντικότερες έννοιες των p2p συστημάτων. Αυτό περιλαμβάνει τη κατανεμημένη αποθήκευση, την επεξεργασία, τη διανομή πληροφοριών και επίσης τις πληροφορίες ελέγχου.
- Self-organized. Όλοι οι κόμβοι συμμετέχουν στην επιλογή του κόμβου που θα εξυπηρετήσει μια αίτηση (request).
- Ad-hoc connectivity. Συχνές αλλαγές στις συνδέσεις (συνδεσμολογία) μεταξύ των κόμβων.
- Discover. Ο κόμβος που θα προσφέρει μία υπηρεσία δεν είναι γνωστός και χρειάζεται εξερεύνηση του δικτύου για να βρεθεί.
- Mobile. Η τοπολογία του δικτύου μπορεί να αλλάξει αν π.χ. κάποιος κόμβος αποσυνδεθεί, και συνδεθεί σε διαφορετική τοποθεσία, ή απλώς δε συνδεθεί καθόλου.

- Mesh. Δεν είναι καθορισμένος ο κόμβος στον οποίο πρεπει να γίνει αναφορά, αλλά καθορίζεται από την ερώτηση και από το μηχανισμό δρομολόγησης του κάθε κόμβου.
- Independent lifetime. Αν κάποιος κόμβος τεθεί εκτός λειτουργίας το δίκτυο συνεχίζει να λειτουργεί κανονικά, με τη διαφορά ότι ο συγκεκριμένος κόμβος δε μπορεί να προσφέρει τις υπηρεσίες του.
- Custom naming. Κάθε κόμβος μπαίνει στο σύστημα με το όνομα που επιθυμεί, και το οποίο το μεταδίδει στο γείτονά του.
- Asynchronous. Η επικοινωνία είναι ασύγχρονη, αφού οι αλλαγές στο δίκτυο είναι πολύ συχνές και οι χρήστες περιοδικά μπαίνουν και βγαίνουν από το σύστημα.

1.4. Ταξινόμηση των P2P συστημάτων.

Για την ταξινόμηση των P2P συστημάτων, λαμβάνουμε υπ΄ όψην μας δύο σημαντικές παραμέτρους, το βαθμό αποκεντρικοποίησης (degree of decentralization), και τη δομή του δικτύου (level of network structure) του Overlay Network [3][4].

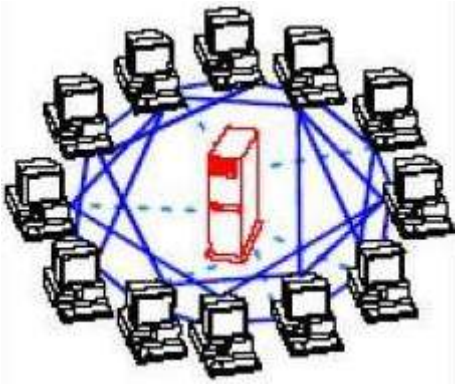
1.4.1. Με βάση το βαθμό αποκεντρικοποίησης.

Με βάση το βαθμό αποκεντρικοποίησης σε ένα P2P σύστημα, μπορούμε να τα ταξινομήσουμε στις παρακάτω τρείς κατηγορίες[3][4]:

I. Κεντρικοποιημένα P2P (Centralized P2P).

Στα συστήματα αυτά, υπάρχει ένας κεντρικός υπολογιστής που διατηρεί τους καταλόγους των πληροφοριών για τους εγγραμμένους χρήστες στο δίκτυο, υπό μορφή meta-data. Επίσης ο κεντρικός υπολογιστής διατηρεί έναν δείκτη (index) των δεδομένων ή των αρχείων που μοιράζονται αυτήν την περίοδο από τους ενεργούς peers. Κάθε peer διατηρεί μια σύνδεση στον κεντρικό υπολογιστή, μέσω του οποίου

στέλνονται οι ερωτήσεις (queries). Η ανταλλαγή δεδομένων γίνεται μεταξύ δύο ομοτίμων. Αυτή η αρχιτεκτονική είχε χρησιμοποιηθεί από το **Napster**. Τέτοια συστήματα είναι απλά και λειτουργούν γρήγορα και αποτελεσματικά για την εύρεση πληροφοριών. Οι αναζητήσεις είναι περιεκτικές και μπορούν να παρέχουν την **εγγύηση στις αναζητήσεις**. Όμως είναι **τρωτά στη λογοκρισία και την κακόβουλη επίθεση** λόγω των κεντρικών υπολογιστών (single point of failure). Τα συστήματα αυτά **δεν είναι εγγενώς κλιμακώσιμα**, λόγω των περιορισμών στο μέγεθος της βάσης δεδομένων και της ικανότητάς των κεντρικών υπολογιστών να ανταποκριθούν στις ερωτήσεις. Δεδομένου ότι οι κεντρικοί κατάλογοι δεν ενημερώνονται πάντα, πρέπει να ανανεώνονται περιοδικά.

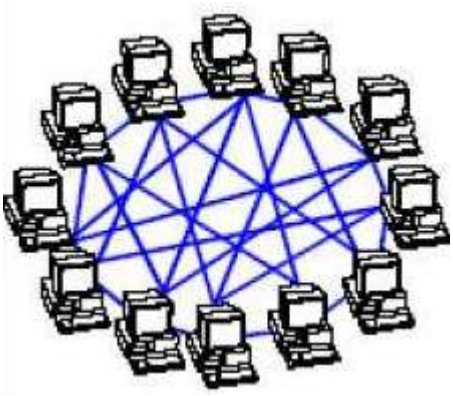


Σχημ. 1.3. Κεντρικοποιημένη τοπολογία δικτύου.

II. Πλήρως αποκεντρικοποιημένα (Purely Decentralized ή pure P2P)

Ένα πλήρως αποκεντρικοποιημένο p2p σύστημα είναι ένα κατανεμημένο σύστημα χωρίς οποιονδήποτε κεντρικό έλεγχο. Σε τέτοια συστήματα όλοι οι κόμβοι είναι ισοδύναμοι σε λειτουργικότητα. Σε αυτά τα δίκτυα οι κόμβοι, ονομάζόνται και ως ”servent” (SERver+cliENT). Ο όρος servent αντιπροσωπεύει την ικανότητα των κόμβων να ενεργήσουν ταυτόχρονα και ως server και ως client.
Το Gnutella v0.4, Freenet , Chord και το CAN είναι περιπτώσεις τέτοιων συστημάτων. Τα pure p2p συστήματα είναι **εγγενώς κλιμακώσιμα**. Η κλιμάκωση (scalability) στο σύστημα περιορίζεται συνήθως από το ποσό των απαραίτητων

κεντρικοποιημένων λειτουργιών και τέτοια συστήματα αποφεύγουν κατά ένα μεγάλο μέρος τις κεντρικές αναφορές ή τους κεντρικούς υπολογιστές (servers). Αυτό το είδος συστημάτων είναι εκ φύσεως **ανθεκτικά στα σφάλματα σύνδεσης** (fault-tolerant), δεδομένου ότι δεν υπάρχει κανένα κεντρικό σημείο αποτυχίας καθώς και η απώλεια ενός ή ακόμα και πολλών ομοτίμων μπορεί εύκολα να αντισταθμιστεί. Έχουν επίσης **μεγαλύτερο βαθμό αυτονομίας ελέγχου των δεδομένων και των πόρων τους**. Από την άλλη, αυτά τα συστήματα παρουσιάζουν μεγάλη χρονική **καθυστέρηση όσον αφορά στην έυρεση των πληροφοριών** και δεν υπάρχει **καμία εγγύηση για την ποιότητα των υπηρεσιών**. Επίσης λόγω έλλειψης μιας σφαιρικής άποψης για το επίπεδο και τη τοπολογία του συστήματος, είναι **δύσκολο να προβλεφθεί η συμπεριφορά του δικτύου**.



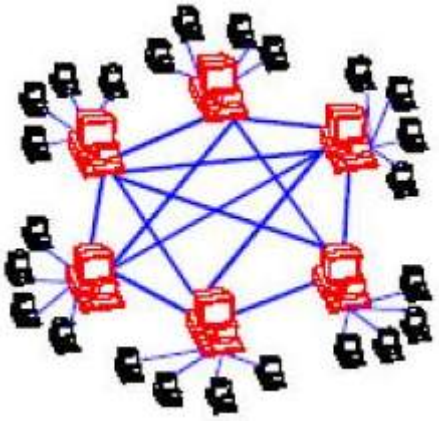
Σχημ. 1.4. Πλήρως αποκεντρικοποιημένη τοπολογία δικτύου.

III. Υβριδικά ή Αποκεντρικοποιημενα P2P (Hybrid ή Decentralized P2P)

Σε αυτά τα συστήματα μερικοί από τους κόμβους υιοθετούν έναν σημαντικότερο ρόλο από τους υπόλοιπους, και ονομάζονται **υπερκόμβοι** (supernodes ή ultrapeers). Πρόκειται για δυναμικές κεντρικές οντότητες. Αυτοί οι κόμβοι διατηρούν δείκτες για τα δεδομένα που διατίθονται στο δίκτυο και μερικά ευρετήρια για τα περιεχόμενα των τοπικών ομοτίμων που συνδέονται με αυτούς. **Τα αιτήματα αναζήτησης πραγματοποιούνται εκ μέρους των supernodes** . Οι ερωτήσεις επομένως στέλνονται στους supernodes, όχι σε άλλους ομότιμους. Σε τέτοια συστήματα οι κόμβοι επιλέγονται αυτόματα ώστε να γίνουν supernodes εάν έχουν ικανοποιητικό εύρος ζώνης (bandwidth) και μεγάλη επεξεργαστική ισχύ. Στη

συνέχεια ένας κεντρικός υπολογιστής παρέχει στους νέους κόμβους έναν κατάλογο ενός ή περισσοτέρων supernodes με τους οποίους θα μπορούν να συνδεθούν. Οι πιό πρόσφατες αρχιτεκτονικές, όπως το **Gnutella v0.6** υιοθετούν αυτή την αρχιτεκτονική. Όταν ένας κόμβος που τηρεί τις προυποθέσεις που αναφέρθηκαν, συνδέεται στο δίκτυο, γίνεται αμέσως supernode και εγκαθιστά συνδέσεις με τους άλλους supernodes, διαμορφώνοντας έτσι ένα επίπεδο μη δομημένο δίκτυο από superpeers.

Σε σύγκριση με τα πλήρως αποκεντρικοποιημένα συστήματα, τα δίκτυα αυτά μειώνουν το χρόνο εύρεσης (discovery time) των δεδομένων όπως επίσης και την κυκλοφορία στα μηνύματα που ανταλλάσσονται μεταξύ των κόμβων με αποτέλεσμα τη καλύτερη διαχείρηση του bandwidth. Σε σύγκριση με τα κεντρικοποιημένα συστήματα, μειώνουν το φόρτο εργασίας στον κεντρικό υπολογιστή αλλά παρουσιάζουν την πιό αργή ανακάλυψη πληροφοριών. Επίσης σε αυτό το είδος συστημάτων δεν υπάρχει μοναδικό σημείο αποτυχίας (point of failure) όπως στα κεντρικοποιημένα συστήματα όπου υπάρχει ένας server. Εάν ένας ή περισσότεροι supernodes τεθούν εκτός λειτουργίας, οι κόμβοι που συνδέονται με αυτούς μπορούν να ξεκινήσουν αμέσως νέες σύνδεσεις, και το δίκτυο θα συνεχίσει να λειτουργεί κανονικά. Στην περίπτωση όπου ένας μεγάλος αριθμός ή ακόμα και όλοι οι supernodes για κάποιο λόγο αποσυνδεθούν από το δίκτυο, οι υπάρχοντες κόμβοι γίνονται αυτόματα supernodes.



Σχημ. 1.5. Τοπολογία δικτύου υβριδικής μορφής.

1.4.2. Με βάση τη δομή δικτύου.

Όπως έχει αναφερθεί, τα συστήματα P2P δημιουργούν ένα overlay network, και κάθε ομότιμος διατηρεί μερικές point-to-point συνδέσεις με άλλους κόμβους. Η τοπολογία αυτού του νοητού δικτύου είναι ιδιαίτερα δυναμική καθώς οι κόμβοι εισάγονται και εξέρχονται από το δίκτυο σε τακτά διαστήματα. Αυτή η τοπολογία μπορεί να οργανωθεί για να χρησιμοποιεί το δίκτυο πιό αποτελεσματικά ή για να εντοπίζει τα περιεχόμενα γρηγορότερα, ή με τρόπο *ad-hoc*, εντελώς ανεξάρτητη από τις φυσικές συνδέσεις και τα περιεχόμενα των hosts . Όταν υπάρχει μια μορφή συσχέτησης μεταξύ των περιεχομένων των κόμβων και της τοπολογίας, λέμε ότι το P2P σύστημα είναι δομημένο. Μπορούμε λίγο πολύ να ταξινομήσουμε τα P2P συστήματα σε τρεις ομάδες, σχετικά με το επίπεδο δομής τους[3][4].

I. *Μη δομημένα δίκτυα (Unstructured networks).*

Στα μη δομημένα δίκτυα, δεν υπάρχει αυστηρά ορισμένη δομή του δικτύου. Η τοποθέτηση των δεδομένων είναι απολύτως ανεξάρτητη από το overlay network και οι ομότιμοι συνδέονται απευθείας ο ένας με τον άλλον χωρίς να έχουν καμία πληροφορία για τα δεδομένα των υπόλοιπων peers. Σε αυτά τα συστήματα, η εύρεση δεδομένων ανέρχεται σε τυχαία αναζήτηση, στην οποία οι διάφοροι κόμβοι εξετάζονται και ρωτούνται εάν έχουν οποιαδήποτε αντιστοιχία για την ερώτηση. Για παράδειγμα το δίκτυο Gnutella ανήκει σε αυτή τη κατηγορία. Τα μη δομημένα συστήματα είναι **εύκολο να εφαρμοστούν** και επίσης **απαιτούν λίγη συντήρηση**, όμως **στερούνται κλιμακωσιμότητας** (scalability). Δεδομένου ότι ο αριθμός των peers που συμμετέχουν στο δίκτυο αυξάνεται, αυτό έχει σαν αποτέλεσμα να αυξάνεται και ο αριθμός των μηνυμάτων που ανταλλάσσονται για εύρεση δεδομένων. Τα μη δομημένα P2P δίκτυα για την αναζήτηση αρχείων χρησιμοποιούν το πρωτόκολλο πλημμύρας (Flooding protocol) το οποίο είναι πολύ ευαίσθητο σε κάποιες ακραίες περιπτώσεις . Εάν π.χ. ο αριθμός συνδέσεων είναι πολύ μικρός , όλοι οι κόμβοι δεν θα μπορούν να εντοπιστούν, για ένα λογικό χρονικό διάστημα. Αντιθέτως, εάν υπάρχουν πάρα πολλές συνδέσεις, τα πολυάριθμα ίδια αντίγραφα του μηνύματος από την ερώτηση θα φθάσουν σε πολλούς κόμβους από διαφορετικές κατευθύνσεις, με συνέπεια τη σπατάλη του εύρους ζώνης. Τα δίκτυα αυτά είναι

εύκολα στη δημιουργία και στη συντήρησή τους και είναι κατάλληλα για αναζήτηση δημοφιλών αντικειμένων. Μειονέκτημα, ότι η τοπολογία δεν είναι βέλτιστη, με πιθανό αποτέλεσμα τις **μη αποδοτικές ανάζητήσεις**.

II. *Δομημένα δίκτυα (Structured networks).*

Στα δομημένα δίκτυα, η τοπολογία ελέγχεται στενά και τα δεδομένα τοποθετούνται σε συγκεκριμένες θέσεις στο σύστημα ομοτίμων. Αυτά τα συστήματα παρέχουν μια ένα είδους χαρτογράφησης μεταξύ του προσδιοριστή δεδομένων (data identifier) και της θέσης, υπό μορφή ενός κατανεμημένου πίνακα δρομολόγησης (Distributed Hash Tables - DHTs), έτσι ώστε οι ερωτήσεις να μπορούν να καθοδηγηθούν αποτελεσματικά στον κόμβο με τα επιθυμητά δεδομένα, και αποτελεσματικότερα απο ότι στη μη δομημένη περίπτωση. Στα δομημένα P2P συστήματα οι peers διατηρούν τις πληροφορίες για τους πόρους που οι γειτονικοί peers μπορούν να προσφέρουν. Πλεονέκτημα αυτών δικτύων αποτελεί η γρήγορη αναζήτηση και ότι καθιστούνται κατάλληλα για αναζήτηση συγκεκριμένων αντικειμένων. Το κύριο μειονέκτημα των δομημένων δικτύων είναι η δυσκολία στη διατήρηση του πίνακα δρομολόγησης με τις συχνές αφίξεις και τις αναχωρήσεις των ομοτίμων και επιπλέον δεν επιτρέπουν στο σύστημα να φτάσει σε μια σταθερή κατάσταση. Παραδείγματα αυτών των δικτύων, είναι τα **Chord**, **CAN** και το **Tapestry**.

III. *Τα μερικώς δομημένα δίκτυα (Loosely structured networks)*

Τέλος, τα μερικώς δομημένα δίκτυα είναι υβριδικές λύσεις μεταξύ των δομημένων και μη δομημένων δικτύων. Σε τέτοια συστήματα, μια χαρτογράφηση υπάρχει μεταξύ της θέσης των αρχείων και της τοπολογίας, αλλά δεν διευκρινίζεται εντελώς και μπορεί να οδηγήσει στην αποτυχία αναζήτησης (η αναζήτηση συντελείται έπειτα σαν το δίκτυο ήταν μη δομημένο). Μπορούμε να αναφέρουμε το δίκτυο **Freenet** σαν αντιπροσωπευτικό παράδειγμα ενός τέτοιου δικτύου.

1.5. Προκλήσεις στα P2P συστήματα.

Τα peer-to-peer συστήματα προσφέρουν διάφορα πλεονεκτήματα σε διάφορους τομείς, έναντι των συμβατικών συστημάτων client-server, όπως στο θέμα της κλιμακωσιμότητας (scalability), ανοχή σφαλμάτων (fault tolerance), απόδοσης (performance). Εντούτοις, υπάρχουν μερικές προκλήσεις που αυτά τα συστήματα θα πρέπει να εξετάζουν[3]:

- *Ασφάλεια*. Οι κατανεμημένες εφαρμογές δημιουργούν πρόσθετες προκλήσεις για την ασφάλεια έναντι της αρχιτεκτονικής client-server. Δεδομένου ότι στα P2P συστήματα το σύνολο ενεργών peers είναι δυναμικό, το **να επιτευχθεί ένα υψηλό επίπεδο ασφάλειας είναι δυσκολότερο**. Οι παραδοσιακοί μηχανισμοί ασφάλειας , όπως τα firewalls, για να προστατεύσουν τα δεδομένα και τα συστήματα από τους εισβολείς και τις επιθέσεις, δεν αρκούν για να προστατεύσουν τα συστήματα αυτά δεδομένου ότι είναι ουσιαστικά παγκοσμίως κατανεμημένα. και επίσης αυτοί οι μηχανισμοί μπορούν ακόμα και να εμποδίσουν την P2P επικοινωνία. Επομένως απαιτούνται νέες έννοιες στο θέμα ασφάλειας που να επιτρέπουν την αλληλεπίδραση και τη κατανεμημένη επεξεργασία (distributed processing) στα διομότιμα συστήματα.

- *Αξιοπιστία*. Αξιόπιστο σύστημα είναι ένα σύστημα που μπορεί να ανακτηθεί όταν εμφανίζεται μια αποτυχία. Οι παράγοντες που πρέπει να ληφθούν υπόψη για την αξιοπιστία είναι η αντιγραφή των δεδομένων, ανίχνευση και αποκατάσταση του προβληματικού κόμβου , ύπαρξη πολλαπλάσιων εγγυήσεων και επαληθεύσεων για τις θέσεις πληροφορίας ώστε να αποφευχθεί ένα single point of failure και η διαθεσιμότητα πολλαπλών διαδρομών στα δεδομένα Υπάρχουν δύο στρατηγικές για την αντιγραφή, η αντιγραφή ιδιοκτητών και η αντιγραφή διαδρομών. Τέλος θα πρέπει να λαμβάνεται υπ'όψην, αν το δίκτυο είναι δομημένο, ή μη δομημένο.

- *Ευελιξία* (Flexibility). Μια από τις σημαντικές πτυχές στα P2P συστήματα είναι η αυτονομία των ομοτίμων έτσι ώστε μπορούν να συνδέονται και να αποσυνδέονται με τη θέλησή τους. Τα πρόσφατα P2P συστήματα χαρακτηρίζονται από τον αποκεντρικοποιημένο έλεγχο, τη μεγάλη κλιμάκωση (large scale) και τον ακραίο δυναμισμό του λειτουργικού περιβάλλοντός τους, κάνοντας τις ιδιότητες της προσαρμογής (adaptation) και της αυτο-οργάνωσης (self-organization) πολύ σημαντικές στη δημιουργία των P2P συστημάτων. Επίσης, όπως και προηγουμένως, θα πρέπει να λαμβάνεται υπ'όψην, αν πρόκειται για δίκτυο δομημένο, ή μη δομημένο.

- *Εξισορρόπηση φόρτου* (Load Balancing). Ο όρος Load Balancing είναι γνωστός και ως Load Transfer ή Process Migration, είναι η μεταφορά δεδομένων από κόμβους με μεγάλο φόρτο κίνησης δεδομένων πρός άλλους κόμβους που έχουν μικρότερο φόρτο. Η κατανομή των δεδομένων ή των υπολογισμών που αποθηκεύονται και πραγματοποιούνται από τους κόμβους είναι ένα κρίσιμο ζήτημα για την αποδοτική λειτουργία των peer-to-peer δικτύων. Οι τεχνικές για την εξισορρόπηση φόρτου, όπως και στην ευελιξία και στην αξιοπιστία, διαφέρουν ανάλογα αν το δίκτυο είναι δομημένο ή μη.

1.6. Μηχανισμοί αναζήτησης – πρωτόκολλα, στα P2P συστήματα.

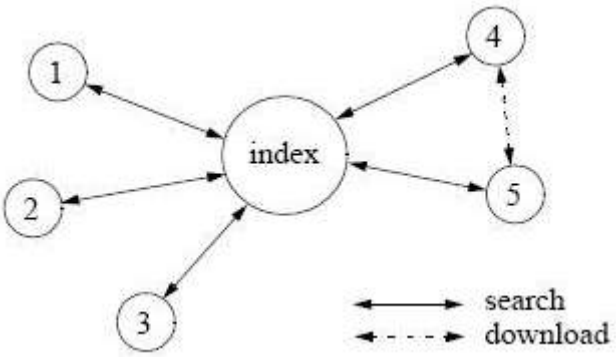
Τα κατανεμημένα διομότιμα δίκτυα απαιτούν συχνά έναν μηχανισμό αναζήτησης για να εντοπίζουν συγκεκριμένα δεδομένα μέσα στο σύστημα. Τα P2P συστήματα έχουν εξελιχθεί από τις κεντρικοποιημένες δομές πρώτης γενειάς στη δεύτερη γενειά flooding-based και την τρίτη έπειτα γενειά συστημάτων βασισμένα στα distributed hash tables[3].

1.6.1. Centralized directory model.

Αυτός ο μηχανισμός χρησιμοποιείται στα υβριδικά συστήματα. Σε αυτό το μοντέλο οι ομότιμοι του συστήματος συνδέονται με τους κεντρικοποιημένους directory servers ή σε super-nodes αν υπάρχουν, οι οποίοι αποθηκεύουν όλες τις πληροφορίες σχετικά με τη θέση και τη χρήση των πόρων. Όταν ένας ομότιμος κάνει αίτηση, ο central index ταιριάζει την αίτηση με τον "καλύτερο " ομότιμο από το κατάλογό του, έπειτα τα δεδομένα ανταλλάσονται απευθείας μεταξύ των δύο ομότιμων. Ο "καλύτερος" ομότιμος, θα μπορούσε να είναι ο φθηνότερος, ο γρηγορότερος, ο κοντινότερος, ή ο πιο διαθέσιμος, ανάλογα με τις απαιτήσεις του χρήστη.

Ο centralized directory server διατηρεί ένα δείκτη (index) με meta-data (όνομα αρχείου,ημερομηνία δημιουργίας κ.τ.λ.) όλων των αρχείων στο δίσκο, ένα πίνακα με πληροφορίες για τη σύνδεση όλων των εγγεγραμμένων χρηστών (IP Address, connection speed κ.α.), καθώς και ένα πίνακα με όλα τα αρχεία που ο χρήστης διατηρεί και μοιράζεται στο δίκτυο.

Επειδή στον μηχανισμό αυτό απαιτούνται οι κεντρικοί υπολογιστές καταλόγων, αυτό μπόρει να περιορίσει τη δυνατότητα κλιμάκωσης του συστήματος, επείδη μπορεί να απαιτηθούν μεγαλύτεροι servers, αν αριθμός των αιτήσεων αυξηθεί, καθώς και μεγαλύτερο χώρο αποθήκευσης αν ο αριθμός των χρηστών αυξηθεί[3].

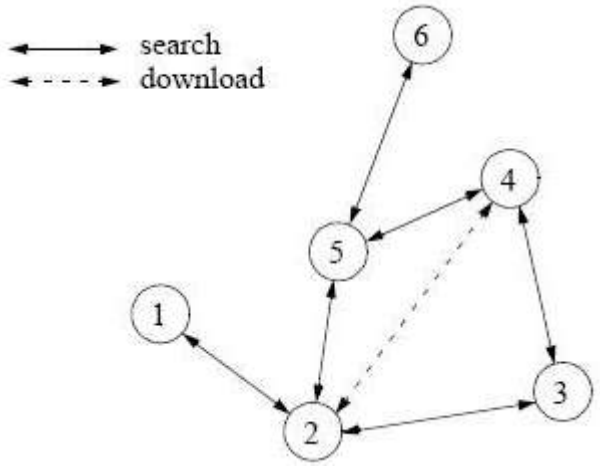


Σχημ. 1.6. Central Index Algorithm.

1.6.2. Flooded requests model.

Εφαρμόζεται στα πλήρως αποκεντρικοποιημένα συστήματα (π.χ. Gnutella v0.4). Στο μοντέλο αυτό κάθε ομότιμος δεν διατηρεί κεντρικό κατάλογο, και κάθε ομότιμος δημοσιεύει πληροφορίες για τα περιεχόμενα που μοιράζεται στο P2P δίκτυο. Εφόσον δεν είναι γνωστές οι πηγές του δικτύου, κάθε ομότιμος που αναζητεί κάποια δεδομένα , πλημμυρίζει το overlay network με ερωτήσεις (queries) για ανακάληψη της πηγής . Κάθε αίτηση από έναν ομότιμο γίνεται flooded (broadcasted) στους άμεσα-απευθείας συνδεδεμένους με αυτόν peers, οι οποίοι στη συνέχεια το προωθούν στους ομότιμους που συνδέονται κ.ο.κ. , μέχρι το αίτημα να απαντηθεί ή να εξαντληθεί ο μέγιστος αριθμός των flooding steps. Χαρακτηριστικό των δικτύων που κάνουν χρήση αυτού του μηχανισμού είναι η ad-hoc συνδεσμολογία. Διάφορες broadcast πολιτικές εχούν εφαρμοστεί για βελτίωση της αναζήτησεις στα P2P δίκτυα. Το δίκτυο Gnutella στην αρχική του μορφή χρησιμοποιούσε αυτή το μηχανισμό και λειτουργούσε ως ένα κατανεμημένο σύστημα αποθήκευσης αρχείων.

Το **flooding protocol δίνει βέλτιστα αποτελέσματα για δίκτυο με μικρό εώς μεσαίο αριθμό ομοτίμων** π.χ. company network, αλλά δε συνιστάται για πολύ μεγάλο αριθμό ομοτίμων. Το πρωτόκολλο αυτό απαιτεί μεγάλο bandwidth δικτύου, λόγω της πληθώρας των μυνημάτων που προκύπτουν κατά τη διαδικασία αναζήτησης. Όλα αυτά κάνουν τη κλιμάκωση ένα πολύ δύσκολο θέμα[3].



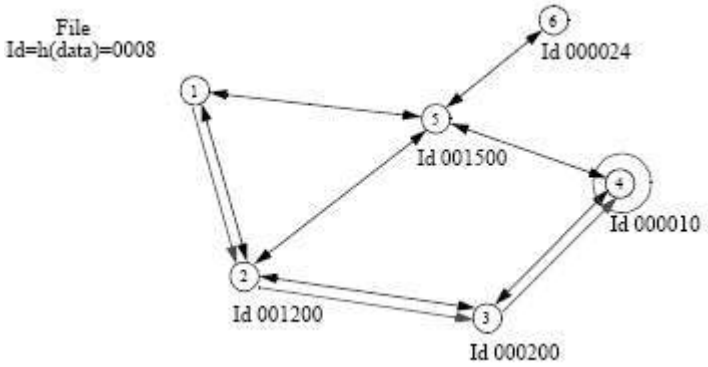
Σχημ. 1.7. Flooded Requests Algorithm.

1.6.3. Document Routing Model.

Αυτό το πρότυπο δρομολόγησης προσθέτει, την έννοια της δομής, στον τρόπο που αποθηκεύονται οι πληροφορίες για τους πόρους, χρησιμοποιώντας distributed hash tables. Το πρωτόκολλο αυτό παρέχει μια χαρτογράφηση μεταξύ του προσδιοριστή των πόρων και της θέσης, υπό μορφή κατανεμημένου πίνακα δρομολόγησης, έτσι ώστε οι ερωτήσεις να μπορούν να καθοδηγηθούν αποτελεσματικά στον κόμβο με τον επιθυμητό πόρο (αποτελέσμα). Το πρωτόκολλο μειώνει τον αριθμό των p2p αλμάτων που πρέπει να ληφθεί για να εντοπιστεί ένας πόρος. Το look-up service εφαρμόζεται οργανώνοντας τους ομότιμους σε ένα δομημένο overlay network, και δρομολογώντας ένα μήνυμα μέσω του overlay network πρός τον αρμόδιο ομότιμο.

Τα κυριότερα δίκτυα που χρησιμοποιούν το document routing model είναι τα: FreeNet, Chord, CAN, Pastry, Tarestry.

Οι στόχοι κάθε αλγορίθμου είναι παρόμοιοι: να μειωθεί ο αριθμός των P2P αλμάτων για να εντοπιστεί ένα έγγραφο ενδιαφέροντος και να μειωθεί το ποσό κατάστασης δρομολόγησης που πρέπει να κρατηθεί σε κάθε ομότιμο[3].



Σχημ. 1.8. Document Routing Algorithm,
(όπως εφαρμόζεται στο δίκτυο Freenet)

ΚΕΦΑΛΑΙΟ 2ο

2.1. Προεπισκόπηση

Το BitTorrent είναι ένα peer-to-peer file sharing πρωτόκολλο που χρησιμοποιείται για τη **διανομή μεγάλης ποσότητας δεδομένων**. Είναι ένα από τα πιό κοινά πρωτόκολλα για τη μεταφορά των μεγάλων αρχείων με σχετικά εύκολο τρόπο με την εφαρμογή μιας διαφορετικής προσέγγισης από ότι υπήρχε πρίν. **Ένας χρήστης μπορεί να λάβει πολλά αρχεία ταυτόχρονα** χωρίς οποιαδήποτε ιδιαίτερη απώλεια του ποσοστού μεταφοράς (transfer rate). Λέγεται ότι είναι πολύ καλύτερο από τις συμβατικές μεθόδους μεταφοράς αρχείων λόγω μιας διαφορετικής αρχής που ακολουθείται από αυτό το πρωτόκολλο. Εξισώνει επίσης τον τρόπο που ένα αρχείο διαμοιράζεται με το να επιτρέπει σε έναν χρήστη όχι μόνο για να λαμβάνει αλλά και να το μοιράζεται με άλλους. Κατά συνέπεια όσο μεγαλύτερος ο αριθμός χρηστών τόσο ευκολότερα και γρηγορότερα ένα αρχείο μπορεί να μεταφερθεί μεταξύ των ομοτίμων.

Το πρωτόκολλο BitTorrent έχει στηριχτεί σε μια τεχνολογία που **καθιστά εφικτή τη διανομή μεγάλης ποσότητας δεδομένων χωρίς την ανάγκη ενός κεντρικού υπολογιστή υψηλού αποθηκευτικού χώρου και "ακριβού" εύρους ζώνης.** Η μεταφορά των αρχείων δεν εξαρτάται από μια ενιαία πηγή που υποτίθεται είναι το αρχικό αντίγραφο του αρχείου αλλά αντ' αυτού το φορτίο θα διανεμηθεί σε διάφορες πηγές. Στη συγκεκριμένη περίπτωση όχι μόνο οι πηγές είναι αρμόδιες για τη μεταφορά αλλά και οι clients ή οι χρήστες που θέλουν να λάβουν το αρχείο περιλαμβάνονται σε αυτήν την διαδικασία. Αυτό έχει σαν αποτέλεσμα την **ομοιόμορφη κατανομή του φορτίου ανάμεσα στους χρήστες** και να καταστήσει έτσι την κύρια πηγή μερικώς απαλλαγμένη από αυτήν την διαδικασία που θα μειώσει την κυκλοφορία δικτύου που επιβάλλεται εκ φύσεως. Λόγω αυτού, το BitTorrent έχει γίνει ένας από τους δημοφιλέστερους μηχανισμούς μεταφοράς αρχείων στο σημερινό κόσμο. Αν και ο ίδιος ο μηχανισμός δεν είναι τόσο απλός, έχει κερδίσει τη δημοτικότητά του λόγω της πολιτικής διανομής που επιβάλλει στους χρήστες του, για να "κατεβάσεις" ένα αρχείο θα πρέπει παράλληλα να "προσφέρεις". Τέλος να

αναφερθεί ότι **το BitTorrent δεν προσφέρει τη δυνατότητα αναζήτησης ενός αρχείου, λειτουργεί μόνο ως πρωτόκολλο μεταφοράς αρχείων**[6]**.**

2.2. History

Το BitTorrent δημιουργήθηκε από το προγραμματιστή **Bram Cohen**, τον Απρίλιο του 2001 και η πρώτη έκδοση του εμφανίστηκε τον Οκτωβρίο του 2002. **Πριν τη δημιουργία και εφαρμογή του Bittorrent , υπήρχαν και άλλες τεχνικές διανομής αρχείων αλλά δεν χρησιμοποιούσαν το εύρος ζώνης αποτελεσματικά.** Το εύρος ζώνης είχε γίνει μια δυσχέρεια (bottleneck) σε τέτοιες μεθόδους. Επίσης σε παρόμοια συστήματα που προυπήρχαν όπως το Napster και το Kazaa είχε παρατηρηθεί ότι **οι περισσότεροι από τους χρήστες μπορούσαν απλά να "κατεβάσουν" τα αρχεία χωρίς την ανάγκη να δίνουν (upload)**. **Έτσι αυξανόταν ο φόρτος του δικτύου στις αρχικές πηγές των δεδομένων,** δηλαδή σε ένα μικρό αριθμό χρηστών. Αυτό οδηγούσε στη μη εκμετάλευση του εύρους ζώνης των υπόλοιπων ομοτίμων. Αυτό ήταν η κύρια πρόθεση πίσω από την εφεύρεση του Cohen, δηλ., να γίνει η **μέγιστη χρησιμοποίηση του εύρους ζώνης όλων των χρηστών** που περιλαμβάνονται στη διανομή των αρχείων. Με αυτόν τον τρόπο, καθένας που ήθελε να κατεβάσει ένα αρχείο έπρεπε να συμβάλει και στην upload διαδικασία. Για το σκοπό αυτό βοήθησε πολύ και το γεγονός ότι στο πρωτόκολλο, το **κάθε αρχείο χωρίζεται σε μικρότερα κομμάτια** (εκτενή αναφορά στη συνέχεια του κεφαλαίου). Αυτή η νέα έννοια του Cohen γέννησε ένα νέο peer to peer πρωτόκολλο αποκαλούμενο BitTorrent[6].

2.3. Η αρχιτεκτονική του Bittorrent.

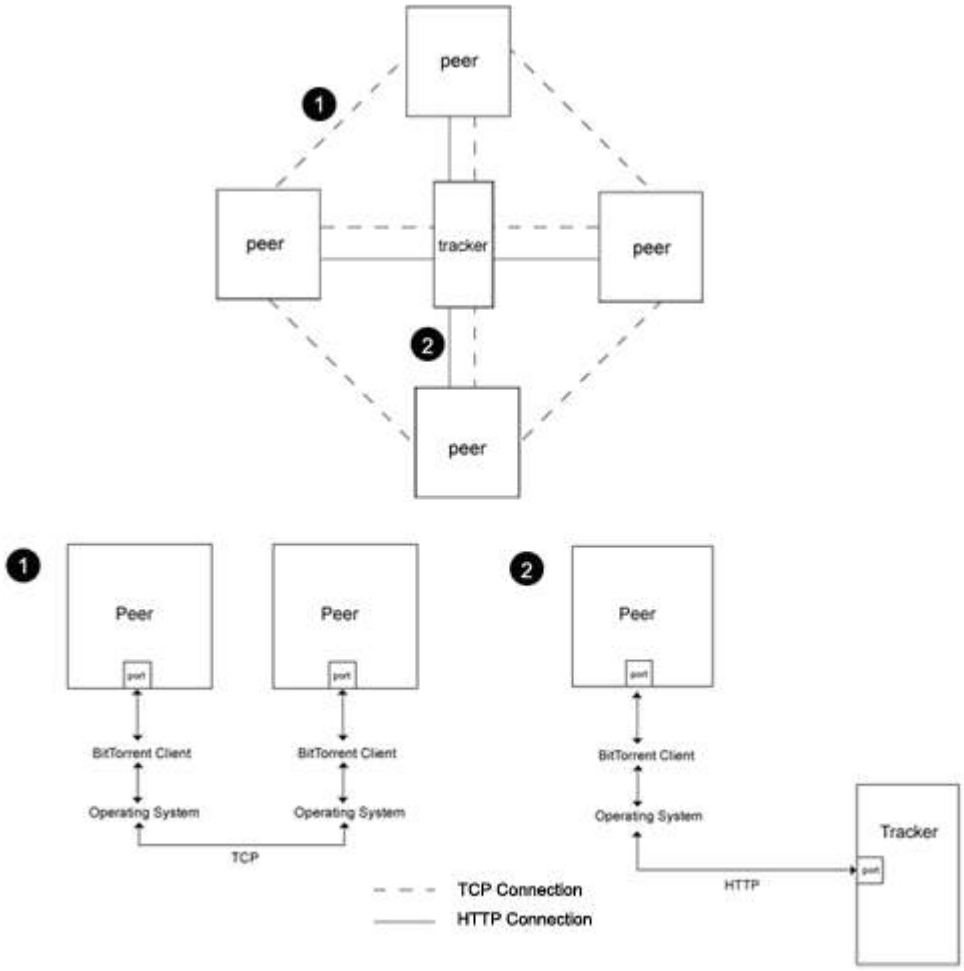
Το πρωτόκολλο Bittorrent μπορεί να αναλυθεί στα παρακάτω πέντε σημεία[6]:

- **Metainfo File** – ένα αρχείο που περιέχει όλες τις απαραίτητες λεπτομέρειες για να λειτουργήσει το πρωτόκολλο.
- **Tracker** – ένας server που βοηθάει στη διαχείρηση του πρωτοκόλλου Bittorrent .
- **Peers** – Χρήστες που ανταλλάσουν δεδομένα μέσω του πρωτοκόλλου Bittorrent .
- **Data** – Τα αρχεία που μεταφέρονται μέσω του πρωτοκόλλου.
- **Client** – Το πρόγραμμα που εγκαθιστάται στον υπολογιστή ενός ομότιμου και εφαρμόζει - ακολουθεί το πρωτόκολλο.

Οι ομότιμοι χρησιμοποιούν το πρωτόκολλο TCP (Transport Control Protocol) για να επικοινωνούν και να ανταλλάσουν δεδομένα. Αυτό το πρωτόκολλο είναι προτιμότερο από άλλα όπως το UDP (User Datagram Protocol) καθώς 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.

Επίσης το πρωτόκολλο BitTorrent μπορεί να περιγραφεί και από την άποψη δύο υπο-πρωτοκόλλων: ένα που περιγράφει τις αλληλεπιδράσεις μεταξύ του tracker και των ομοτίμων (THP: Tracker HTTP Protocol), και ένα που περιγράφει όλες τις αλληλεπιδράσεις μεταξύ των ομοτίμων (PWR: Peer Wire Protocol). Η χρήση και ο ακριβής τρόπος λειτουργίας των πρωτοκόλλων αυτών, θα δωθεί στη συνέχεια του κεφαλαίου.

Το ακόλουθο διάγραμμα επεξηγεί πώς οι ομότιμοι αλληλεπιδρούν ο ένας με τον άλλον, και επικοινωνούν με έναν κεντρικό tracker.



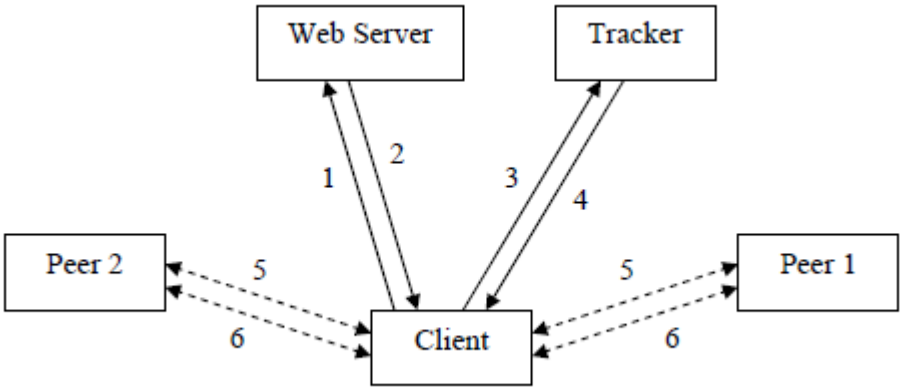
Σχημ. 2.1. 1 : *Επικοινωνία μεταξύ ομοτίμων.* 2 : *Επικοινωνία μεταξύ ομότιμου και tracker.*

2.4. Metainfo file.

Όταν κάποιος επιθυμεί να δημοσιεύσει δεδομένα χρησιμοποιώντας το πρωτόκολλο BitTorrent, πρέπει να δημιουργήσει ένα αρχείο metainfo ή απλά, αρχείο torrent. 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 διεύθυνση IP του tracker που θα συνδεθεί .

Ένας tracker είναι ένας κεντρικός υπολογιστής που διαχειρίζεται τα torrent και θα συζητηθεί αναλυτικά μετέπειτα. Στο αρχείο δίνεται μια επέκταση ".torrent", και τα δεδομένα εξάγονται από το αρχείο από έναν BitTorrent client, ένα πρόγραμμα που τρέχει στον υπολογιστή του χρήστη, και υιοθετεί το πρωτόκολλο bittorrent.

Δεδομένου ότι το πρωτόκολλο BitTorrent δεν διαθέτει λειτουργία αναζήτησης, **τα αρχεία .torrent βρίσκονται συνήθως μέσω ιστοσελίδων (HTTP)** και μέσω των μηχανών αναζήτησης ή των μηχανών αναζήτησης trackers[6].



Σχημ. 2.2 Ένα τυπικό Bittorrent σύστημα.

Κάθε torrent αρχείο πρέπει να περιέχει τις ακόλουθες πληροφορίες, (ή "keys")[5]:

- info: Ένα λεξικό που περιγράφει το αρχείο-α του torrent. Είτε για ένα αρχείο, είτε τη δομή καταλόγου για περισσότερα αρχεία. Οι κρυπτογραφήσεις για κάθε κομμάτι (piece) δεδομένων, σε SHA 1 format αποθηκεύονται εδώ.
- announce: Ανακοινώνεται η διεύθυνση URL του tracker ως string (συμβολοσειρά)

Τα παρακάτω είναι προαιρετικά keys που μπορούν επίσης να χρησιμοποιηθούν :

- announce-list: Χρησιμοποιείται για απαρίθμηση των backup (εφεδρικών) trackers.
- ημερομηνία δημιουργίας: Ο χρόνος δημιουργίας του torrent με το UNIX time stamp (integer seconds since 1-Jan-1970 00:00:00 UTC)
- comment: Σχόλια από το δημιουργο του torrent.
- created by: Όνομα και έκδοση του προγράμματος για τη δημιουργία του metainfo file.

Οι πληροφορίες που περιλαμβάνονται στο torrent αρχείο κωδικοποιούνται προτού να σταλθούν. Η κωδικοποίηση γίνεται χρησιμοποιώντας μια συγκεκριμένη μέθοδο του bittorrent γνωστή ως **"bencoding"**[5]. 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 byte strings, ακέραιους αριθμούς, λίστες και λεξικά .

2.4.1. Διανομή του Metainfo file.

Επειδή όλες οι πληροφορίες που απαιτούνται για το torrent περιλαμβάνονται σε ένα αρχείο, αυτό το αρχείο μπορεί εύκολα να διανεμηθεί μέσω άλλων πρωτοκόλλων, και καθώς το αρχείο αντιγράφεται, ο αριθμός των ομοτίμων μπορεί να αυξηθεί πολύ γρήγορα. Η δημοφιλέστερη μέθοδος διανομής είναι χρησιμοποιώντας

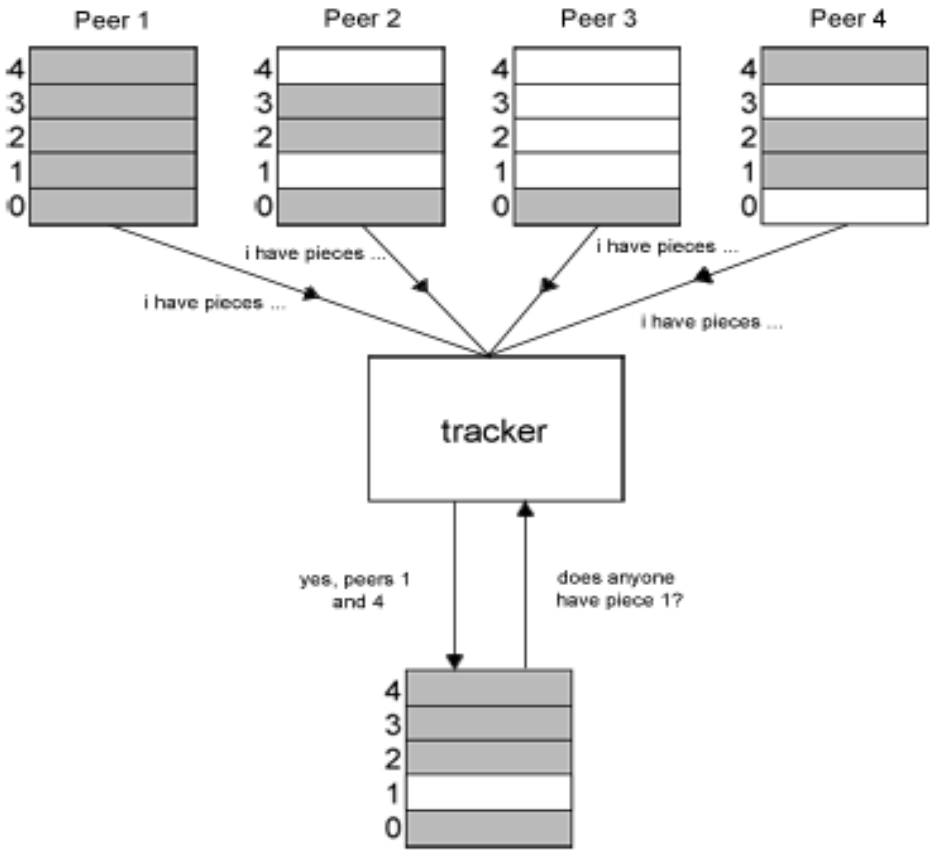
ένα public indexing site που φιλοξενεί τα αρχεία torrent. Ένας seeder (ομότιμος που διαθέτει ολόκληρο το αρχείο) θα φορτώσει το αρχείο, και έπειτα άλλοι μπορούν να κατεβάσουν ένα αντίγραφο αυτού μέσω του πρωτοκόλλου HTTP και να συμμετέχουν στο torrent.

2.5. Tracker.

Στο BitTorrent οι χρήστες συμμετέχουν ενεργά στη διανομή των αρχείων μαζί με τους κεντρικούς υπολογιστές. Αυτό είναι η μοναδικότητα αυτού του πρωτοκόλλου. Επίσης αυτό χρειάζεται μια εφαρμογή ενός αφοσιωμένου κεντρικού υπολογιστή αποκαλούμενου ως tracker για να χειριστεί τους ομότιμους που συνδέονται στο δίκτυο. Το φορτίο κατανέμεται μέσω του δικτύου μεταξύ των ομοτίμων και των κεντρικών υπολογιστών.

Ένας tracker, πρόκειται για έναν **software server**, χρησιμοποιείται για να διαχειριστεί τους χρήστες που συμμετέχουν σε έναν torrent. Αποθηκεύει στατιστικά για το torrent, αλλά **ο κύριος ρόλος του είναι να επιτρέπει στους ομοτίμους " να βρούν ο ένας τον άλλον " και να αρχίσει την επικοινωνία μεταξύ τους** , δηλ. να βρεί τους ομότιμους που έχουν τα δεδομένα που ζητούνται. Αρχικά οι ομότιμοι δεν γνωρίζουν τίποτα ο ένας για τον άλλον έως ότου υπάρξει μια απάντηση από τον tracker.

Όποτε ένας ομότιμος έρχεται σε επαφή με τον tracker , αναφέρει ποια κομμάτια (pieces) ενός αρχείου έχει στη κατοχή του. Με αυτό το τρόπο, όταν ένας άλλος ομότιμος "κάνει μια ερώτηση" στον tracker, αυτός του **παρέχει μια τυχαία λίστα ομοτίμων που συμμετέχουν στο torrent**, οι οποίοι έχουν τα απαιτούμενο(α) κομμάτι(α), όπως φαίνεται και στο παρακάτω σχήμα [6]:



Σχημ. 2.3. Tracker.

Ο ρόλος του tracker τελειώνει μόλις βρούν οι ομότιμοι "ο ένας τον άλλον". Από εκεί και έπειτα, η επικοινωνία γίνεται άμεσα μεταξύ των ομοτίμων, και ο tracker δεν εμπλέκεται πλέον.

Ο tracker είναι στην ουσία μια υπηρεσία HTTP/HTTPS που ανταποκρίνεται σε HTTP GET requests. Η διεύθυνση του tracker που διαχειρίζεται ένα torrent διευκρινίζεται στο metainfo αρχείο, ένας μόνο tracker μπορεί να διαχειριστεί πολλά torrents. Οι multiple trackers μπορούν επίσης να χαρακτηριστούν, ως backups (αντίγραφα ασφαλείας), τα οποία διαχειρίζονται από το πρόγραμμα BitTorrent client που τρέχει στον υπολογιστή του κάθε χρήστη. Οι BitTorrent clients επικοινωνούν με τον tracker χρησιμοποιώντας HTTP GET requests (αιτήματα), πρόκειται για μια standard CGI μέθοδο. Αυτό αποτελείται από την επισύναψη του "?", στο URL, και χωρίζοντας τις παραμέτρους με το "&"[5][6].

2.5.1. THP: Tracker HTTP Protocol.

Ο tracker εφαρμόζει 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 HTPP ή HTTPS server, και έχει τη συμπεριφορά που περιγράφεται παρακάτω[6]:

1. **Ο ομότιμος (client) στέλνει ένα GET request στον URL tracker**, με ορισμένες CGI μεταβλητές και τιμές που προστίθενται στο URL . Αυτό γίνεται με τον τυποποιημένο τρόπο, δηλ., εάν η βάση 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" έγγραφο, που περιέχει ένα κωδικοποιημένο (bencoded) λεξικό. Αυτό το λεξικό έχει όλες τις πληροφορίες για τον ομότιμο. **Ο tracker επιστρέφει μία τυχαία λίστα με 50 ομότιμους.** Κάθε ομότιμος περιγράφεται με 6 bytes, 4 bytes για την IP και 2 για το port.

3. Έπειτα ο ομότιμος στέλνει εκ νέου αιτήματα (re-requests), είτε σε τακτά χρονικά διαστήματα, είτε όταν εμφανίζεται ένα συμβάν, και ο tracker αποκρίνεται.

Οι παράμετροι που χρησιμοποιούνται στο GET request από τον client προς τον tracker είναι οι[5]:

- info_hash: 20-byte SHA1 hash του info key από το metainfo file.
- peer_id: 20-byte string που χρησιμοποιείται σαν μια μοναδική ID για τον client.
- port: Η πόρτα στη οποία ακούει ο client .
- uploaded: Η συνολική ποσότητα δεδομένων (σε ten based ASCII) που έχει μοιραστεί από το client, από τη στιγμή που έστειλε το 'started' event στο tracker.
- downloaded: Η συνολική ποσότητα δεδομένων (σε ten based ASCII) που έχει "κατεβεί" από το client, από τη στιγμή που έστειλε το 'started' event στο tracker.

- left: Ο αριθμός των bytes (σε ten based ASCII)που υπολοίπονται να κατέβουν από το client.
- compact: Υποδεικνύει ότι ο client δέχεται τις συμπιεσμένες απαντήσεις. Η λίστα από τους ομότιμους μπορεί έπειτα 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: started, stopped, completed.
- ip: (προαιρετικό) Η IP διεύθυνση του client machine, σε dotted format.
- numwant: (προαιρετικό) Ο αριθμός των ομοτίμων που ο client επιθυμεί να λάβει από τον tracker.
- key: (προαιρετικό) Επιτρέπει σε ένα client να προσδιοριστεί αν η IP address των ομοτίμων αλλάξει.
- trackerid: (προαιρετικό) Αν ένα προηγούμενο announce περιείχε το trackerid, θα πρέπει να καθοριστεί εδώ.

Ο tracker στη συνέχεια ανταποκρίνεται με ένα "text/plain" document μαζί με ένα κωδικοποιημένο (bencoded) λεξικό με τις παρακάτω πληροφορίες[5]:

- failure message: Άν υπάρχει, δε θα υπάρχουν περαιτέρω πληροφορίες. Η τιμή του είναι ένα error message, που εξηγεί γιατί το request απέτυχε.
- warning message: Παρόμοιο με το failure message, αλλά εξακολουθεί να γίνεται η επεξεργασία του response.
- interval: Ο αριθμός των δεπτερολέπτων που ένας client θα πρέπει να περιμένει μεταξύ των αιτήσεων που στέλνει στον tracker.
- tracker id: Ένα string που ο client θα πρέπει να στείλει πίσω με την επόμενή του ανακοίνωση.
- complete: Αριθμός των ομοτίμων που έχουν ολόκληρο το αρχείο, seeders.
- incomplete: Αριθμός ομοτίμων που δεν είναι seeders. Ονομαζονται lechers.
- peers: Μια λίστα από dictionaries που συμπεριλαμβάνουν: peer id, IP και ports όλων των ομοτίμων

2.5.2. Scraping.

Ο tracker υποστηρίζει μία υπηρεσία που ονομάζεται “scrape”. Ο σκοπός του είναι να **παρέχει πληροφορίες για τα torrents που φιλοξενεί**[5][6].

Το scraping είναι η διαδικασία υποβολής ερωτήσης 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 στο announce URL, στο τελευταίο '/' αντί για τη λέξη 'announce' να αντικατασταθεί με το 'scrape'.
π.χ. Announce URL : http://example.com/announce
Scrape URL : http://example.com/scrape

Ο tracker τότε ανταποκρίνεται ένα “text/plain” document με bencoded πληροφορίες (files, complete, downloaded, incomplete, name).

2.6. Ομότιμοι.

Οι ομότιμοι είναι οι χρήστες που συμμετέχουν σε ένα torrent, και μπορούν να διαθέτουν ή ένα μέρος του αρχείου, ή το πλήρες αρχείο (γνωστό ως seed - σπόρος). Τα κομμάτια ζητούνται από τους ομότιμους , αλλά δεν είναι εγγυημένο ότι θα σταλθούν. Εξαρτάται από τη κατάσταση του κάθε ομότιμου.

2.6.1. Επικοινωνία μεταξύ ομοτίμων.

Το Bittorrent χρησιμοποιεί το **TCP** (Transmission Control Protocol) πρωτόκολλο **για την αποστολή μηνυμάτων και δεδομένων μεταξύ των ομοτίμων**, και αντίθετα από άλλα πρωτόκολλα, δεν χρησιμοποιεί το UDP (User Datagram Protocol). Ωστόσο το πρόγραμμα μTorrent, κάνει χρήση και του UDP πρωτοκόλλου.

2.6.2. PWR: Peer Wire Protocol.

Το peer wire (peer to peer) sub-protocol εφαρμόζεται πάνω από το πρωτόκολλο TCP. Το μήνυμα που περνάει είναι συμμετρικό, δηλ. τα μηνύματα είναι τα ίδια που στέλνονται και στις δύο κατευθύνσεις. Όταν ένας ομότιμος θέλει να αρχίσει μια σύνδεση, εγκαθιδρύει 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[6].

Handshake message.

Το μύνημα χειραψίας αποτελείται από πέντε μέρη[6]:

- Ένα single byte, που περιέχει τη τιμή 19 (base 10 ASCII). Αυτό είναι το μήκος του string χαρακτήρων που ακολουθεί αυτό το byte.
- Το string χαρακτήρων “BitTorrent protocol” , το οποίο περιγράφει το πρωτόκολλο. Τα νεώτερα πρωτόκολλα πρέπει να ακολουθήσουν 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 εφαρμογές.
- Μια κωδικοποίηση (bencoded) SHA1 hash των 20 byte , της τιμής info key του αρχείου torrent. Αυτό είναι ίδιο hash που στέλνεται στον tracker στη μεταβλητή info_hash.
- Το string χαρακτήρων των 20 byte που αντιπροσωπεύει την peer id. Αυτό είναι η ίδια τιμή που στέλνεται στον tracker.

Handshaking.

Η "χειραψία" πραγματοποιείται ως εξής[6]:

1. Η χειραψία αρχίζει με το χαρακτήρα 19 (base 10 ASCII) ακολουθούμενο από το string 'BitTorrent Protocol'.
2. Ένα 20 byte SHA1 hash της bencoded info τιμής από το metainfo, στέλνεται στη συνέχεια. Αν τα hash μεταξύ των ομοτίμων δε ταιριάζουν, η σύνδεση κλείνει.
3. Μια peer id (20 byte) στέλνεται, η οποία χρησιμοποιείται στη συνέχεια για τις αιτήσεις στον tracker και περιλαμβάνεται στις στις αιτήσεις του ομότιμου. Εάν η id του ομότιμου δεν ταιριάζει με αυτή που αναμενόταν, η σύνδεση κλείνει.

Εάν ένας ομότιμος είναι ο πρώτος παραλήπτης σε μια χειραψία, και το info_hash δεν ταιριάζει με οποιοδήποτε torrent που εξυπηρετεί, η σύνδεση κλείνει. Εάν ο ομότιμος που ξεκίνησε τη σύνδεση, λάβει μια χειραψία όπου το peer id δεν ταιριάζει με την id που παραλαμβάνεται 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 interested είτε not interested για έναν άλλο ομότιμο, και είτε choke είτε not choke για τον άλλο ομότιμο. Το choking (πνίξιμο) σημαίνει ότι κανένα αίτημα δεν θα απαντηθεί, και interested (ενδιαφερόμενος) σημαίνει ότι ο ομότιμος ενδιαφέρεται για να κατεβάσει κομμάτια του αρχείου από τον άλλο ομότιμο.

Αυτό σημαίνει ότι κάθε ομότιμος χρειάζεται τέσσερις 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 από τη χειραψία είναι δομημένα ως εξής:
[μήκος μηνύματος σε ακέραιο αριθμό] [ένα byte που περιγράφει τον τύπο μηνύματος] [ωφέλιμο φορτίο]

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

2.6.3. Message stream.

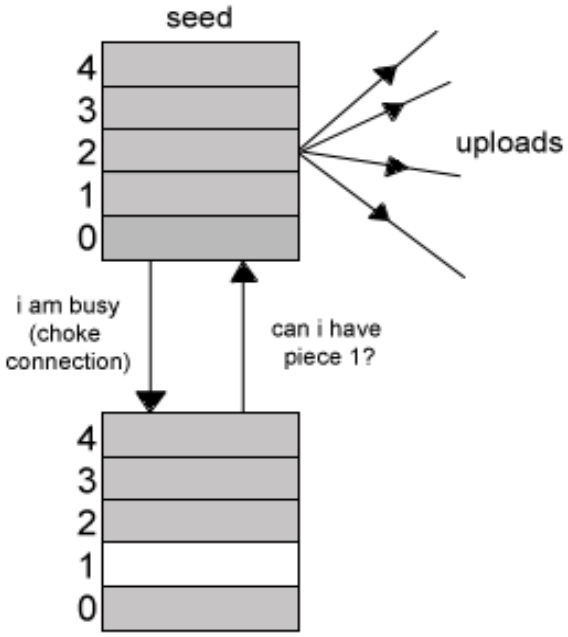
Η σταθερή ροή των μηνυμάτων επιτρέπει σε όλους τους ομότιμους στο swarm να στέλνουν δεδομένα, και να ελέγχουν τις αλληλεπιδράσεις με άλλους ομοτίμους. Ένας ομότιμος θα "ενδιαφερθεί" για δεδομένα εάν υπάρχει ένας ομότιμος που έχει τα απαραίτητα κομμάτια. Εάν ο ομότιμος που έχει αυτά τα δεδομένα δεν είναι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

2.7. Επιλογή ομοτίμου. - Διανομή δεδομένων μεταξύ ομοτίμων.

Για να διατηρήσει την **ακεραιότητα των δεδομένων** που έχουν γίνει download/upload, ένας ομότιμος δεν αναφέρει ότι έχει ένα κομμάτι του αρχείου έως ότου γίνει ένα **chechsum (SHA-1 algotithm)** με αυτό που περιλαμβάνεται στο αρχείο metainfo (για να είναι σίγουρο ότι έχει κατέβει το "σωστό" κομμάτι). Οι ομότιμοι θα συνεχίσουν να κατεβάζουν όσα δεδομένα μπορούν από όλους τους διαθέσιμους ομότιμους. Επίσης μπορούν να εμποδίσουν άλλους ομότιμους από το να κατεβάζουν, εάν είναι απαραίτητο. Αυτό είναι γνωστό ως choking[6].

2.7.1. Choking.

Όταν ένας ομότιμος λαμβάνει ένα αίτημα για ένα κομμάτι από έναν άλλο ομότιμο, μπορεί να επιλέξει να αρνηθεί να το μεταφέρει. Εάν αυτό συμβαίνει, ο ομότιμος λέγεται ότι είναι **choked** ("πνιγμένος"), όπως φαίνεται και στο παρακάτω σχήμα[5][6] :



Σχημ. 2.4. Choking *από* ένα ομότιμο.

Το choking γίνεται για διάφορους λόγους. Ο έλεγχος συμφόρησης TCP δε μπορεί να θεωρηθεί αξιόπιστος κατά τη διάρκεια πολλών ταυτόχρονων συνδέσεων. 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, **οι χρήστες τείνουν να προτιμούν και να συνεισφέρουν (upload) σε αυτούς που τους είχαν "βοηθήσει".**

Σε κάποια χρονική στιγμή, ο ομότιμος θα επιλέξει να κάνει upload στους 5 καλύτερους γείτονές του, οι οποίοι παρέχουν στον ομότιμο τα καλύτερα download rate. Το σύστημα κάνει αυτή τη λειτουργία κάθε 10 δεπτερόλεπτα.

Υπάρχουν διάφορα κριτήρια που ένας καλός choking αλγόριθμος πρέπει να ικανοποιεί[5][6]:

- Πρέπει να θέτει όρια στον αριθμό των ταυτόχρονων uploads, για καλύτερη TCP απόδοση (το TCP έχει καλύτερη συμπεριφορά όταν ο αριθμός των ταυτόχρονων uploads είναι περιορισμένος) δηλαδή να διατηρείται ο default αριθμός ταυτόχρονων uploads (max_uploads). Όλα τα περαιτέρω αιτήματα στον client θα χαρακτηριστούν όπως choked. Συνήθως η προεπιλογή για τα max_uploads είναι 4.
- Πρέπει να αποφεύγει τη γρήγορη εναλλαγή choking-unchoking, γνωστό ως **"fibrillation"** (το συχνό choking/unchoking δημιουργεί διακοπές στη μεταφορά δεδομένων που μειώνει την απόδοση του πρωτοκόλλου).
- Πρέπει να ανταποκρίνεται 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, αυτό είναι γνωστό και ως "optimistic unchoking".

Ο choking αλγόριθμος που επί του παρόντος εφαρμόζεται , αποφεύγει το fibrillation, απλά με το να αλλάζει τους choked ομότιμους, μία φορά κάθε δέκα δευτερόλεπτα. Η εναλλαγή αυτή καθώς και η τήρηση του αριθμού των max_uploads, ρυθμίζονται με το "ελευθερώνονται" (unchoking) οι τέσσερις ομότιμοι που έχουν το καλύτερο ποσοστό upload και που ενδιαφέρονται να συνεισφαίρουν. Αυτό μεγιστοποιεί το download rate του εκάστοτε ομότιμου. Οι ομότιμοι που έχουν ένα καλύτερο upload rate (σε σύγκριση με τους ήδη υπάρχοντες - downloaders) αλλά δεν ενδιαφέρονται, γίνονται unchoked. Εάν ενδιαφερθούν, τότε ο ομότιμος με το χειρότερο upload rate γίνεται choked. Εάν ένας client έχει ένα πλήρες αρχείο, χρησιμοποιεί το upload rate του παρά download rate του για να αποφασίσει ποιοι ομότιμοι θα μπλοκαριστούν.

Για να εξασφαλιστεί δικαιοσύνη μεταξύ των ομοτίμων, υπάρχει ένα σύστημα που εναλλάσει τους ομοτίμους που κατεβάζουν. Αυτό είναι γνωστό ως "**optimistic unchoking**".

2.7.2. Optimistic unchoking.

Το Bittorrent ενσωματώνει το μηχανισμό "optimistic unchoking" για την ανακάλυψη "καλύτερων ομοτίμων", δηλαδή αυτών που θα προσφέρουν (upload) περισσότερο. Οι ομότιμοι στο Bittorrent κάνουν upload μόνο σε ένα υποσύνολο των ομοτίμων που συνδέονται, οι οποίοι αποκαλούνται προτιμημένοι ομότιμοι *(preferred peers)* [5][6].

Στο optimistic unchoking, 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. Αυτό 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.** Έτσι ο νέος γείτονας έχει μια ευκαιρία για να συμμετάσχει[5][6].

2.7.3. Anti-snubbing.

Περιστασιακά συμβαίνει 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 έως ότου, μέσω του optimistic unchoking, βρεί καλύτερους ομότιμους. Για να μετριάσει αυτό το πρόβλημα, όταν δηλαδή περνάει η διάρκεια ενός λεπτού χωρίς να πάρει ένα ούτε ένα 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[7].

Αυτό οδηγεί συχνά σε περισσότερα από ένα ταυτόχρονα optimistic unchoking, (εξαίρεση στο κανόνα ότι μπορεί να γίνεται μόνο ένα optimistic unchoking τη φορά), το οποίο έχει σαν αποτέλεσμα **τα download rate να ανακτούνται πολύ πιο γρήγορα, όταν αυτά υποχωρούν**.

Όι μηχανισμοί chocking, optimistic unchocking και anti-snubbing, σε συνδιασμό με τη στρατηγική tit-for-tat, αποσκοπούν στο να υπάρχει **δικαιοσύνη** μεταξύ των ομοτίμων που βρίσκονται στο σύστημα, αλλά και στη **εξάλειψη του free-ridind**. Με τον όρο free-riding εννοούμε την εγωκεντρική συμπεριφορά ενός ομοτίμου, ο οποίος σταματάει να συνειφέρει όταν ολοκληρωθεί η λήψη του αρχείου του.

2.8. Επιλογή κομματιού.

Η σειρά με την οποία επιλέγονται τα κομμάτια από διάφορους ομότιμους είναι ζωτικής σημασίας για να υπάρξει καλή απόδοση στο σύστημα. Αν χρησιμοποιείται ένας "κακός" αλγόριθμος , θα μπορούσε να προκύψει μία κατάσταση στην οποία κάθε ομότιμος θα έχει όλα τα διαθέσιμα κομμάτια, εκείνης της χρονικής περιόδου, αλλά κανένας δε θα έχει τα υπολειπόμενα, συνήθως τα τελευταία κομμάτια του αρχείου. Αν ο ομότιμος (seed), που είχε ανεβάσει το αρχείο, σταματήσει πλέον να το μοιράζεται, τότε κανείς δε θα μπορεί να το αποκτήσει ολοκληρωμένο.

Οι ομότιμοι βάζουν συνεχώς σε μια "σειρά αναμονής" τα κομμάτια που πρόκειται να κατέβουν και τα οποία είχαν ζητήσει. Επομένως ο tracker απαντά συνεχώς στον χρήστη με μια λίστα ομοτίμων που έχουν τα ζητούμενα κομμάτια. Ποιο κομμάτι θα επιλεχθεί εξαρτάται από τον BitTorrent client. Υπάρχουν τέσσερα στάδια της επιλογής ενός κομματιού, τα οποία αλλάζουν ανάλογα σε ποιο σημείο ολοκλήρωσης για την απόκτηση του αρχείου είναι ένας ομότιμος [6]:

2.8.1. Strict Priority.

Κάθε ομότιμος πρέπει να ολοκληρώσει το κατέβασμα όλων των τμημάτων (blocks) (16kb) ενός κομματιού (256kb) για να ζητήσει το επόμενο κομμάτι. Αυτό έχει σαν αποτέλεσμα να μπορεί ομότιμος να κατεβάζει γρήγορα ολόκληρα κομμάτια[6].

2.8.2. Random First Piece.

Όταν ξεκινάει για πρώτη φορά η διαδικασία να κατέβει ένα αρχείο, δεδομένου ότι ο ομότιμος δεν έχει τίποτα να "μοιράσει", ένα κομμάτι επιλέγεται τυχαία για να αρχίσει το download. Έπειτα **επιλέγονται τυχαία κομμάτια** έως ότου ολοκληρωθεί το πρώτο κομμάτι και να ελεγθεί. Μόλις συμβεί αυτό αρχίζει η στρατηγική, 'rarest first'[6].

2.8.3. Rarest First Algorithm.

Όταν ένας ομότιμος επιλέγει ποιο θα είναι το επόμενο κομμάτι που θα κατεβάσει, τότε **επιλέγεται το σπανιότερο κομμάτι από το τρέχον swarm**, δηλ. το κομμάτι που διαθέτει ο μικρότερος αριθμός ομοτίμων. Αυτό σημαίνει ότι τα πιό κοινά κομμάτια αφήνονται τελευταία, και η εστίαση πηγαίνει στην αντιγραφή των σπανιότερων κομματιών. Να αναφέρουμε ότι swarm είναι μια τυχαία ομάδα ομοτίμων που ένας BitTorrent client είναι σε επικοινωνία . Κάθε swarm δημιουργείται όταν ένας χρήστης επιθυμεί να "κατεβάσει" ένα αρχείο, το οποίο σημαίνει ότι για κάθε διαφορετικό αρχείο, θα υπάρχει και ένα διαφορετικό swarm.

Στην αρχή ενός torrent, θα υπάρχει μόνο ένας σπόρος - seeder με το πλήρες αρχείο. Είναι πιθανό τότε, να υπάρξει bottleneck (μείωση της απόδοσης του συστήματος) εάν πολύ ομότιμοι προσπαθούν να έχουν πρόσβαση στο ίδιο κομμάτι. Ο αλγόριθμος rarest first αποφεύγει αυτή τη κατάσταση επειδή διαφορετικοί ομότιμοι

έχουν διαφορετικά κομμάτια. Δεδομένου ότι όσοι περισσότεροι ομότιμοι συνδεθούν, ο φόρτος στο tracker θα λιγοστεύει καθώς οι ομότιμοι θα αρχίσουν το download μεταξύ τους .

Εν τέλει, ο αρχικός seeder θα εξαφανιστεί από το torrent. Αυτό θα μπορούσε να είναι για λόγους κόστους , ή πιο συνήθως λόγω ζητημάτων εύρους ζώνης. Η απώλεια ενός seed (του αρχικού αρχείου) διατρέχει τον κίνδυνο της απώλειας κομματιών εάν κανένας τρέχον downloader δεν τα έχει. **Ο αλγόριθμος "rarest first" λειτουργεί για να αποτρέψει την απώλεια κυρίως των ποιο σπάνιων κομματιών. Αυτό το πετυχαίνει με το να αντιγράφει, όσο το δυνατόν γρηγορότερα, τα κομμάτια που κινδυνεύουν να χαθούν.** Εάν ο αρχικός seeder φύγει πριν τουλάχιστον ένας άλλος ομότιμος έχει αποκτήσει το πλήρες αρχείο, κανένας δε θα καταφέρει να αποκτήσει ολόκληρο το αρχείο, εκτός και αν ένας seeder επανασυνδεθεί στο swarm[6].

2.8.4. Endgame Mode.

Όταν η απόκτηση του αρχείου πλησιάζει στην ολοκλήρωση της, και περιμένοντας ένα κομμάτι από έναν ομότιμο με αργό transfer rate (ποσοστό μεταφοράς), το πιο πιθανό είναι να καθυστερήσει η ολοκλήρωση του αρχείου. Για να αποτραπεί αυτό, τα υπόλοιπα υπο-κομμάτια (blocks) ζητούνται από όλους τους υπόλοιπους ομότιμους στο τρέχον swarm[5].

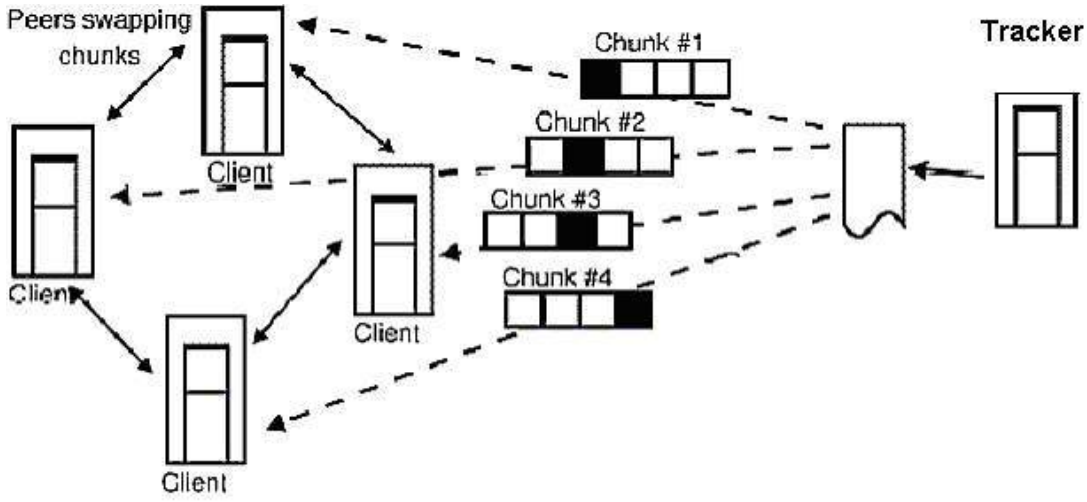
2.9. Δεδομένα - Προσέγγιση Bittorrent.

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

Το κάθε πρόγραμμα που κάνει χρήση του πρωτοκόλλου Bittorrent (Bittorrent client) εφαρμόζει μια τεχνική που ονομάζεται **swarming**. Η τεχνική αυτή χρησιμοποιείται από πολλά αναπτυγμένα συστήματα διανομής αρχείων για την αύξηση της απόδοσης του συστήματος καθώς επίσης απομακρύνει το πρόβλημα ότι ένας κόμβος μπορεί να μην έχει την χωρητικότητα να αποθηκεύσει ολόκληρο το αρχείο.

Στο BitTorrent, τα δεδομένα που μοιράζονται διαιρούνται σε πολλά ίσα μέρη που ονομάζονται **chunks** (κομμάτια) συνήθως των 128KB ή των 256KB. Κάθε κομμάτι υποδιαιρείται περαιτέρω σε υπο-κομμάτια που ονομάζονται **blocks**. Κατά συνέπεια, **οι peers μπορούν να κατεβάζουν παράλληλα διαφορετικά blocks σε τυχαία σειρά από διαφορετικούς peers και να αυξήσουν την διεκπεραιωτική ικανότητα** – ρυθμός αποστολή/παροχή (throughput) της υπηρεσίας. Κάθε ομότιμος ενεργεί ως server παρέχοντας υπηρεσίες σε πολλαπλούς δέκτες. Εντούτοις, οι περιορισμοί στο εύρος ζώνης (που αναγκάζονται συνήθως από τις τοπικές πολιτικές ή το δίκτυο πρόσβασης) δημιουργούν την ανάγκη για μια σειρά αναμονής σε κάθε ομότιμο όπου τα αιτήματα για τα blocks αποθηκεύονται και περιμένουν να εξυπηρετηθούν βασισμένα σε ένα εφαρμοσμένο χρονοπρογραμματισμό βασιζόμενος κυρίως στη προτεραιότητα.

Όλοι **οι ενδιαφερόμενοι clients στη διανομή ενός δεδομένου ομαδοποιούνται σε ένα swarm που ουσιαστικά πρόκειται για ένα τυχαίο overlay network**, και κάθε ένας από τους οποίους ρυθμίζεται από μια κεντρική οντότητα αποκαλούμενη ως tracker. Το Bittorrent δεν απαιτεί έναν χρήστη για να κατεβάσει ένα αρχείο από έναν κεντρικό υπολογιστή. Αντ' αυτού ένα αρχείο μπορεί να αποκτηθεί από πολλούς χρήστες που κατεβάζουν το ίδιο αρχείο. Μόλις ένας χρήστης έχει κάποιο σημαντικό αριθμό κομματιών ενός αρχείου μπορεί να αρχίσει να τα μοιράζεται με άλλους χρήστες . Όταν κάποιος έχει αποκτήσει το πλήρες αρχείο, ονομαζόμενος **seeder**, και μπορεί να βοηθήσει στη διαδικασία με τη μεταφορά των κομματιών του αρχείου στους άλλους χρήστες, αυτό εξαρτάται βέβαια από το κατά πόσο επιθυμεί ο ίδιος να προσφέρει, μειώνοντας το γενικό φορτίο στον κεντρικό υπολογιστή[6].



Σχημ. 2.5. BitTorrent File Transfer

Κάθε client στέλνει ανεξάρτητα ένα αρχείο που ονομάζεται torrent, το οποίο περιέχει τη θέση του tracker μαζί με τη τιμή hash κάθε κομματιού. Hash είναι μια κρυπτογραφική λειτουργία που δημιουργεί μια "εικόνα" για μια μεγάλη ποσότητα δεδομένων. Το BitTorrent χρησιμοποιεί τη SHA1 hash λειτουργία για να καθορίσει ποια μέρη του αρχείου είναι "καλά" και ποια δεν είναι. Όπως έχει αναφερθεί σε προηγούμενη παράγραφο, για να υπάρχει ακεραιότητα στα δεδομένα γίνεται ένα checksum με χρήση του SHA1 hash algorithm.

Μόλις λάβει ένας client όλα τα blocks για ένα δεδομένο κομμάτι (chunk), **μπορεί να ελέγξει τη τιμή hash εκείνου του κομματιού με το hash που υπαρχει στο torrent.** Κατά συνέπεια μόλις ένας client έχει κατεβάσει και ελέγξει όλα τα κομμάτια, μπορεί να είναι βέβαιος ότι έχει τα σωστά και πλήρη δεδομένα.

2.9.1. Μέγεθος κομματιού.

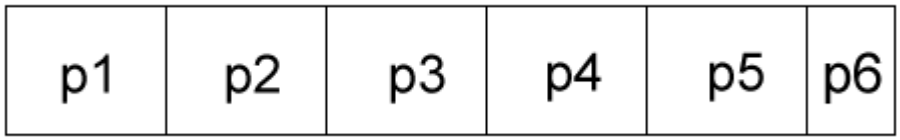
Τα δεδομένα, όπως έχει αναφερθεί χωρίζονται σε μικρότερα κομμάτια που στέλνουν μεταξύ τους οι ομότιμοι χρησιμοποιώντας το bittorrent πρωτόκολλο. Αυτά τα κομμάτια είναι ενός σταθερού μεγέθους, το οποίο επιτρέπει στον tracker να διατηρεί ετικέττες, στο ποιος έχει και ποια κομμάτια, των δεδομένων. Το bittorrent πρωτόκολλο επίσης, ορίζει ένα hash code σε κάθε κομμάτι, το οποίο μπορεί να

ελεγχθεί από το downloader για την ακεραιότητα των δεδομένων. Αυτά τα hashes αποθηκεύονται ως τμήμα του "αρχείου metainfo".

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 hash codes πρέπει να αποθηκευτούν στο metainfo αρχείο. Επομένως, εμπειρικά, τα κομμάτια πρέπει να επιλεχτούν έτσι ώστε το metainfo αρχείο 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 φαίνεται και στο παρακάτω σχήμα[6]:



Σχημ. 2.6. Κομμάτια ενός αρχείου.

2.10. BitTorrent Clients

Ένας BitTorrent client είναι ένα εκτελέσιμο πρόγραμμα που εφαρμόζει το πρωτόκολλο BitTorrent. Τρέχει μαζί με το λειτουργικό σύστημα στο μηχάνημα του χρήστη, και χειρίζεται τις αλληλεπιδράσεις μεταξύ του tracker και των ομοτίμων. Το πρόγραμμα εγκαθιστάται στο λειτουργικό σύστημα και είναι αρμόδιο για τον έλεγχο της ανάγνωσης/γραφής των αρχείων, άνοιγμα των sockets κ.λπ.

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

Υπάρχει μία ποικιλία από BitTorrent Clients 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 τρέχων διαδικασιών. Δεδομένου ότι όλες οι εφαρμογές υιοθετούν το ίδιο πρωτόκολλο, δεν υπάρχει κανένα ζήτημα ασυμβιβατότητας, εντούτοις λόγω των διάφορων tweaks και των βελτιώσεων μεταξύ αυτών, ένας ομότιμος μπορεί να έχει καλύτερη απόδοση από άλλους που τρέχουν την ίδια εφαρμογή.

Κατά κανόνα τα προγράμματα αυτά κάνουν 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[6].

2.11. Συμπεράσματα.

Το BitTorrent έχει πρωτοπορήσει στην mesh-based διανομή αρχείων που χρησιμοποιεί αποτελεσματικά όλα τα uplinks των συμμετέχοντων κόμβων. Οι περισσότεροι που ασχολούνται με ποαρόμοιες έρευνες χρησιμοποιούν παρόμοιους κατανεμημένους και τυχαίους αλγορίθμους για την επιλογή ομοτίμου, καθώς και του κομματιού, αλλά με διαφορετική έμφαση.

Η εφαρμογή του BitTorrent σε αυτή την εποχή διαμοιρασμού των πληροφοριών, είναι σχεδόν ανεκτίμητη. Εντούτοις, ακόμα δεν είναι τελειοποιημένο δεδομένου ότι είναι ακόμα επιρρεπείς σε κακόβουλες επιθέσεις και πράξεις λόγω κακής χρήσης. Επιπλέον, **η διάρκεια ζωής κάθε torrent δεν είναι ακόμα ικανοποιητική**, το οποίο σημαίνει ότι το διάστημα της διανομής ενός αρχείου μπορεί μόνο να επιζήσει για μια περιορισμένη χρονική περίοδο. Κατά συνέπεια, η περαιτέρω ανάλυση και μια πιό λεπτομερής μελέτη στο πρωτόκολλο θα επιτρέψουν την ανακάλυψη περισσότερων τρόπων βελτίωσης[6].

ΚΕΦΑΛΑΙΟ 3^{ο}

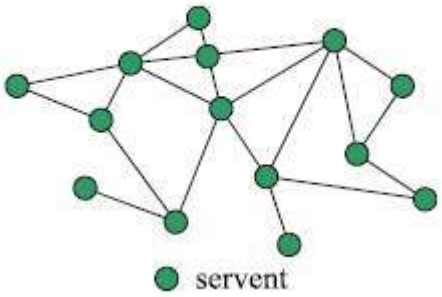
3.1. Εισαγωγή.

Αναπτύχθηκε από τον Justin Frankel και Tom Pepper της Nullsoft το 2000. Η λέξη Gnutella ήταν μια συγχώνευση των λέξεων Nutella (γνωστή σοκολάτα) και GNU General Public License (GPL). **Είχε σχεδιαστεί αρχικά να αποδείξει ότι η έννοια ενός pure P2P δικτύου θα μπορούσε να λειτουργήσει**. Η Nullsoft εμφάνησε το Gnutella αμέσως μόλις αποκτήθηκε από τη AOL (America On Line). Η AOL έβαλε γρήγορα ένα τέλος στο project, δεδομένου ότι ανησυχούσαν για τις νομικές επιπτώσεις της εφαρμογής ενός τέτοιου προγράμματος. Παρ' όλα αυτά, το Gnutella είχε ήδη κυκλοφορήσει στο ευρύ κοινό και σύντομα, πολλές άλλες εκδόσεις του αναπτύχθηκαν και ήταν διαθέσιμες. Το Gnutella έτσι υιοθετήθηκε ως πρωτόκολλο για διαμοιρασμό αρχείων στα P2P δίκτυα. Πρόκειται για **ανοιχτό πρωτόκολλο**, ελεύθερο σε οποιεσδήποτε τροποποιήσεις[11].

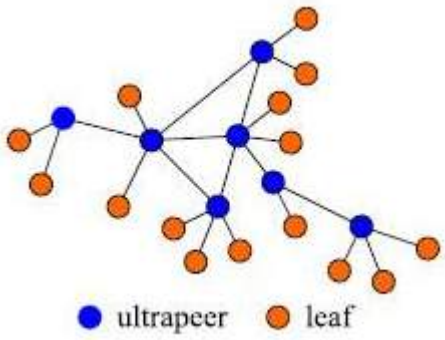
3.2. Το δίκτυο Gnutella.

Το δίκτυο Gnutella αποτελείται από ένα σύνολο συνδεδεμένων κόμβων, το οποίο δημιουργεί ένα **μη δομημένο overlay network** μέσω του οποίου πραγματοποιείται η κυκλοφορία στο δίκτυο, και που αποτελείται ουσιαστικά από queries, απαντήσεις στα queries, και επίσης άλλα μηνύματα ελέγχου για να διευκολύνει την ανακάλυψη άλλων κόμβων. Οι κόμβοι έχουν δυναμική συμπεριφορά, δηλ μπορούν να εισέρχονται και να εξέρχονται στο δίκτυο ανά πάσα στιγμή (**ad-hoc τοπολογία**).

Στην αρχική του μορφή (Gnutella v0.4), το δίκτυο ήταν **πλήρως αποκεντρικοποιημένο** (pure P2P), ενώ αργότερα (Gnutella v0.6), απέκτησε μια **υβριδική μορφή** με την παρουσία των ultrapeers.



Σχημ. 3.1. Τοπολογία του δικτύου Gnutella v0.4



Σχημ. 3.2. Τοπολογία του δικτύου Gnutella v0.6

Οι χρήστες αλληλεπιδρούν με τους κόμβους παρεχοντάς τους μία λίστα με τους πόρους που διαθέτουν και επιθυμούν να ανταλλάξουν στο δίκτυο. Μπορούν να εισαγάγουν 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 οι οποίοι θα μοιραστούν τα δεδομένα** .

Οι ανταλλαγές δεδομένων γίνονται άμεσα μεταξύ των κόμβων με τη χρήση του standard HTTP protocol.

3.2.1. Δίκτυο Gnutella v0.4.

Όπως έχει αναφερθεί, το αρχικό δίκτυο Gnutella v0.4 αποτελεί ένα πλήρως αποκεντρικοποιημένο και μη δομημένο P2P δίκτυο. Δεν υπάρχει η παρουσία κάποιας κεντρικής οντότητας 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.

3.3. Συμμετοχή ομοτίμου στο δίκτυο.

3.3.1. Gnutella client.

Κάθε χρήστης που θέλει που θέλει να συμμετέχει στο δίκτυο Gnutella, τρέχει στον υπολογιστή του ένα πρόγραμμα ονομαζόμενο ως Gnutella client, και το οποίο υλοποιεί το πρωτόκολλο Gnutella.

Στο δίκτυο Gnutella οι εφαρμογές αυτές ονομάζονται **servents** ή εναλλακτικά **gnodes**. Παρέχουν διεπαφές μέσω των οποίων οι χρήστες μπορούν να κάνουν ερωτήσεις και να δουν τα αποτελέσματα, να δεχτούν ερωτήσεις από άλλους servents, να ελέγξουν αν μπορούν να απαντήσουν, και αν ναι, να στείλουν τα αποτελέσματα. Οι ίδιοι κόμβοι είναι υπεύθυνοι για την διαχείριση της κίνησης η οποία υπάρχει και χρησιμοποιείται για την ακεραιότητα του δικτύου.

Δεν υπάρχει κάποιο τυποποιημένο λογισμικό Gnutella client. Αντ' αυτού κάποιος μπορεί να βρεί μια διαφορετική συλλογή από clients που όλοι όμως υποστηρίζουν το βασικό πρωτόκολλο Gnutella ή κάποιες εκδόσεις του. Οι clients μπορούν να επικοινωνούν μεταξύ τους, αλλά οι υπεύθυνοι - developers για την ανάπτυξη των προγραμμάτων είναι ελεύθεροι να εφαρμόσουν τη λειτουργικότητα και τυχόν επεκτάσεις στο πρόγραμμα όπως κρίνουν αυτοί. Οι προδιαγραφές του Gnutella όπως και οι περισσότεροι clients είναι open source (ως GnuGPL) και έχουν

αναπτυχθεί για κάθε είδους πλατφόρμας όπως Windows, Linux/Unix, Java, και Macintosh.

3.3.2. Σύνδεση ομοτίμου στο δίκτυο Gnutella.

3.3.2.1 Εύρεση αρχικού ενεργού κόμβου.

Για να μπορέσει ένας νέος κόμβος να συνδεθεί στο δίκτυο Gnutella πρέπει πρώτα **να συνδεθεί με έναν τουλάχιστον κόμβο ο οποίος είναι διαθέσιμος**, αυτό γίνεται μέσω του Gnutella client. Όποτε αρχίζει να εκτελείται ένας Gnutella client, δεν γνωρίζει ποιοι άλλοι κόμβοι είναι εκείνη τη στιγμή διαθέσιμοι στο δίκτυο. Επομένως το πρώτο βήμα είναι να βρεί τουλάχιστον ένα ενεργό κόμβο. Αυτή η διαδικασία καλείται **bootstrapping**[17]. Μέχρι να ολοκληρωθεί αυτή η διαδικασία, ο ομότιμος δεν μπορεί να συμμετέχει σε δραστηριότητες ανταλλαγής αρχείων. Στο δίκτυο Gnutella v0.4 δεν υπάρχει κάποιος κεντρικός server διαθέσιμος για να βοηθήσει στο σκοπό αυτό. Εάν ένας client εκτελούσε μία δοκιμαστική διαδικασία, παραδείγματος χάρη στέλνοντας μυνήματα 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:

Στο πρώτο αλγόριθμο, ένας client αποθηκεύει και διατηρεί μία λίστα (στο σκληρό δίσκο του υπολογιστή) από IPs με όλους τους γνωστούς κόμβους από τη τελευταία φορά που συνδέθηκε στο Gnutella και στην επόμενη σύνδεση στέλνει το κατάλληλο μύνημα σε όλες αυτές τις IPs. Δεδομένου ότι τα εργαλεία για την ανταλλαγή αρχείων, συχνά εκτελούνται στο background και οι σύγχρονοι υπολογιστές σπάνια είναι εκτός λειτουργίας, υπάρχει μια καλή πιθανότητα να βρεθεί ένας έγκυρος κόμβος. Επιπλέον, εάν τυχαία ο υπολογιστής συνδεθεί μέσω μιας IP που έχει αλλαξεί (αρκετά συνηθισμένο στις dial-up συνδέσεις) αλλά ο νέος υπολογιστής τρέχει επίσης το πρόγραμμα Gnutella, τότε το bootstrapping θα

λειτουργήσει κανονικά δεδομένου ότι δεν έχει σημασία ποιος Gnutella client βρέθηκε. Όλοι συμπεριφέρονται το ίδιο .

Ο δεύτερος αλγόριθμος είναι πιό αξιόπιστος και λειτουργεί με τον αποκαλούμενο τρόπο "web caches", σύστημα **GwebCache**. 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 αλλάζουν (update) καθημερινά επομένως υπάρχει επικοινωνία μεταξύ των σελίδων, καθώς επίσης παρέχουν και URLs από άλλες σελίδες. Ένας "καλός" Gnutella client 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 δεδομένου ότι απλά γνωρίζουν κάποιες IPs και URLs, καθώς και σχετικά εύκολο στη οργάνωση.

3.3.2.2 Σύνδεση ομοτίμου με τον ενεργό κόμβο.

Αφού ο αρχικός ομότιμος βρεί ένα ενεργό κόμβο, εγκαθιδρύεται μία σύνδεση μεταξύ αυτών, με σκοπό να ληφθεί η IP διεύθυνση του ενεργού κόμβου. Μόλις ληφθεί η διεύθυνση, δημιουργείται μια **σύνδεση TCP/IP** με βάση το πρωτόκολλο χειραψίας Gnutella, (το οποίο είναι διαφορετικό στις εκδόσεις v 0.4 και v 0.6) . Οποιαδήποτε άλλη απάντηση σημαίνει ότι ο servent δεν είναι πρόθυμος να δεχτεί τη σύνδεση. Οι συνδέσεις μπορούν να απορριφθούν, παραδείγματος χάριν, επειδή οι εκδόσεις του πρωτοκόλου δεν είναι συμβατές ή επειδή εκείνος ο συγκεκριμένος servent έχει ήδη πάρα πολλές συνδέσεις.

3.3.2.3 Σύνδεση ομοτίμου με υπόλοιπους κόμβους.

Αφού 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 τους κόμβους από τη λίστα κοκ (αλγόριθμος "**Friend of a Friend**")[17]. Κατά αυτό το τρόπο, 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 .

3.4. Το πρωτόκολλο Gnutella .

Έχοντας συνδεθεί στο δίκτυο μετά δηλαδή την εγκατάσταση σύνδεσης TCP (έχοντας μία ή περισσότερες ανοιχτές συνδέσεις με κόμβους ήδη συνδεδεμένους στο Gnutella), οι κόμβοι στέλνουν μηνύματα για να αλληλεπιδρούν μεταξύ τους.

Τα μηνύματα γίνονται **broadcast-limited flooded**, στέλνονται σε όλους τους κόμβους με τους οποίους ο αποστολέας έχει ανοιχτές συνδέσεις) και στη συνέχεια προωθούνται προς τα πίσω (**back-propagated**). Στέλνονται σε μία συγκεκριμένη σύνδεση στην αντίθετη κατεύθυνση από την διαδρομή που ακολούθησε ένα αρχικό broadcasted μήνυμα.

Διάφορα χαρακτηριστικά γνωρίσματα του πρωτοκόλλου διευκολύνουν τον μηχανισμό broadcast/back-propagation[12]*:*

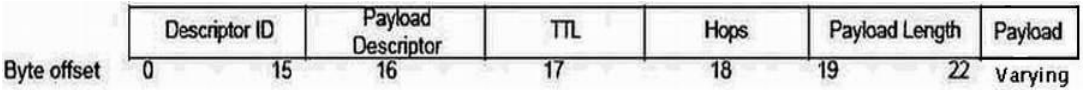
- Πρώτον, κάθε μήνυμα έχει ένα μοναδικό αριθμό σαν αναγνωριστικό **Globally Unique Identifier (GUID).**
- Δεύτερον, κάθε κόμβος κρατάει στην μνήμη του τα πιο πρόσφατα μηνύματα που δρομολόγησε και χρησιμοποιείται για να αποτρέψει το re-broadcasting και να κάνει εφικτή την υλοποίηση του back-propagation.
- Τρίτον, 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 από περιγραφείς-μυνήματα ,

που χρησιμοποιούνται για τη διαβίβαση των δεδομένων μεταξύ των servents, και ένα σύνολο κανόνων για την ανταλλαγή των descriptors μεταξύ των ομοτίμων. Οι περιγραφείς του πρωτοκόλλου, και ουσιαστικά οι τύποι των μηνυμάτων που επιτρέπονται στο δίκτυο είναι PING, PONG, QUERY, QUERYHIT, PUSH και GET.

3.4.1. Δομή πακέτου στο δίκτυο.

Αρχικά, πρίν από τους περιγραφείς, υπάρχει ένας Descriptor Header ο οποίος ορίζει τον εκάστοτε περιγραφέα πρωτοκόλλου. Κάθε πακέτο Gnutella και έχει τη δομή που απεικονίζεται παρακάτω[8]:



Σχημ. 3.3. Descriptor Header

- Descriptor ID: Ένα 16-byte string που προσδιορίζει, τον descriptor στο δίκτυο. Είναι μοναδικό για κάθε μύνημα. Κατά κανόνα, η λειτουργία της διεύθυνσης του αποστολέα κόμβου, χρησιμοποιείται στα μυνήματα Pong και QueryHit ως προσδιοριστικό του προορισμού (δημιουργός).

- Payload Descriptor: Προσδιορίζει το τύπο του μυνήματος. Ανάλογα με την τιμή, αντιστοιχεί και το ανάλογο μύνημα, έτσι έχουμε:
  0x00 = Ping, 0x01 = Pong, 0x40 = Push, 0x80 = Query,
  0x81 = QueryHit

- TTL: Time to live. Αριθμός που δείχνει πόσες φορές το μύνημα θα διαβιβαστεί από τους Gnutella servents προτού να απομακρυνθεί από το δίκτυο. Συνήθως η αρχική τιμή για το TTL ως 7. Κάθε servent ελλατώνει κατά ένα τη τιμή του TTL, πρίν το στείλει προς ένα άλλο servent. Όταν το TTL φθάσει το 0, το μύνημα δεν θα προωθηθεί περαιτέρω. **Το TTL είναι ο μόνος τρόπος να αφαιρεθούν (και κατα**

συνέπεια να περιοριστούν) τα μυνήματα 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 μύνημα μεταβιβάζεται από servent σε servent, τα πεδία TTL και Hops της κεφαλίδας πρέπει ικανοποιούν την συνθήκη:
  TTL (0) = TTL (i) + Hops (i), Όπου TTL (i) και Hops (i) είναι η τιμή των πεδίων TTL και Hops στη κεφαλίδα, στο i-στο άλμα του μυνήματος, για το i>= 0.

- Payload length: Είναι το μήκος του μυνήματος (όχι παραπάνω από 4kb), το οποίο ακολουθεί αμέσως μετά τη κεφαλίδα του μυνήματος. Ο επόμενος descriptor header είναι τοποθετημένος τόσα bytes όσο είναι το Payload_Length, από το τέλος αυτής της κεφαλίδας. Με πιο απλά λόγια, δεν υπάρχει κενό στο Gnutella's data stream. Το πεδίο Payload Length είναι ο μόνος τρόπος για έναν servent να βρεί την αρχή του επόμενου μυνήματος στην εισερχόμενη ροή των δεδομένων. Οι servents πρέπει αυστηρά να "εγκρίνουν" το πεδίο Payload Length για κάθε descriptor που λαμβάνουν (τουλάχιστον για τους καθορισμένου μήκους περιγραφείς).
  Το πεδίο αυτό πρέπει να ελέγχεται έτσι ώστε ο servent να παραμένει συγχρονισμένος με το input stream του. Αν ο servent είναι εκτός συγχρονισμού με το input stream του, τότε η σύνδεση πέφτει.
  Επί του παρόντος, ο μόνος τρόπος για την αξιόπιστη ανάγνωση της ροής δεδομένων του δικτύου, που δημιουργείται από τα μυνήματα, είναι εξετάζοντας τη κάθε κεφαλίδα του Payload Length για να βρεθεί η αρχή του επόμενου descriptor.

- Payload: Αμέσως μετά από τη κεφαλίδα του μυνήματος, ακολουθεί η ωφέλιμη πληροφορία (τα δεδομένα) του μυνήματος που αποτελείται

από από τους περιγραφείς του πρωτοκόλλου: PING, PONG, QUERY, QUERYHIT, PUSH και GET. Πιο αναλυτικά:

3.4.2. Ανακάλυψη γειτόνων - Συντήρηση δικτύου.

3.4.2.1. Μυνήμα PING.

Μετά την εγκατάσταση της σύνδεσης TCP, ο συνδεμένος στο δίκτυο, κόμβος στέλνει στους overlay γειτονές του ένα **μύνημα PING** (με μετρητή βημάτων), δηλώνοντας έτσι ουσιαστικά την παρουσία του στο δίκτυο. Οι γειτονικοί κόμβοι προωθούν το μύνημα (flooded - broadcasted) στους δικούς τους γείτονες κοκ μέχρι να εξαντληθεί ο μετρητής βημάτων.

Στο δυναμικό περιβάλλον όπου το Gnutella λειτουργεί, οι κόμβοι συχνά "έρχονται και φεύγουν" (ad-hoc τοπολογία) και οι συνδέσεις δικτύων είναι αναξιόπιστες. Για να αντιμετωπίσει αυτό το περιβάλλον, μετά από την ένταξη του στο δίκτυο, κάθε κόμβος στέλνει **περιοδικά PING μυνήματα** στους γείτονές του για να ανακαλύψει άλλους συμμετέχοντες κόμβους. Χρησιμοποιώντας αυτές τις πληροφορίες, ένας αποσυνδεμένος κόμβος μπορεί πάντα να επανασυνδέσει στο δίκτυο. Οι κόμβοι αποφασίζουν πού να συνδεθούν βασισμένοι μόνο στις τοπικές πληροφορίες, και διαμορφώνοντας έτσι ένα δυναμικό, self-organizing δίκτυο από ανεξάρτητες οντότητες. Αυτό το δίκτυο εικονικού, επιπέδου εφαρμογής έχει σαν κόμβους τους Gnutella servents στους και για συνδέσεις του έχει τις ανοικτές συνδέσεις TCP.

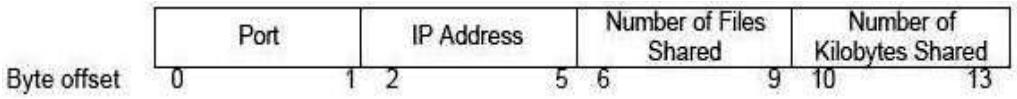
Τα μυνήματα Ping δε περιέχουν δεδομένα και άρα είναι μηδενικού μήκους. Αντιπροσωπεύονται απλά από τη κεφαλίδα του μηνύματος όπου το πεδίο Payload_Descriptor έχει τη τιμή 0x00 και ο τομέας Payload_Length είναι 0x00000000. Περιέχει επίσης ένα μετρητή βημάτων TTL, που καθορίζει πόσες φορές το αίτημα μπορεί να διαβιβαστεί σε άλλους υπολογιστές. Ο servent χρησιμοποιεί τα μυνήματα αυτά για να εξετάσει ενεργά το δίκτυο και να αναζητήσει

άλλους ομότιμους. Οι servent implementers πρέπει να κάνουν κάθε προσπάθεια να ελαχιστοποιούν το Ping traffic στο δίκτυο[8].

3.4.2.2. Μυνήμα PONG.

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

Ένας Pong descriptor έχει τη παρακάτω δομή[8]:



Σχημ. 3.4. Pong Descriptor

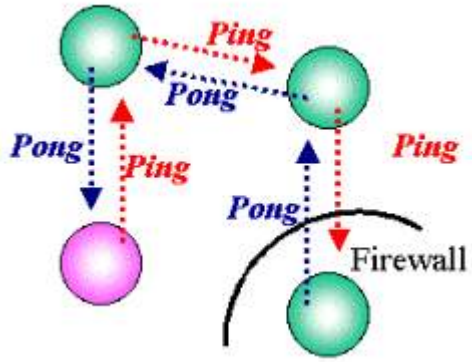
- Port: Ο αριθμός πόρτας στον οποίο ο host (που έστειλε το PONG μύνημα) μπορεί να δεχτεί εισερχόμενες συνδέσεις.
- IP Address: Η IP διεύθυνση (σε IPv4 format) του αποκρινόμενου host (που έστειλε το PONG μύνημα).
- Number of files shared: Ο αριθμός αρχείων που ο servent με τη δεδομένη IP διεύθυνση και 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 Pong descriptor στέλνεται μόνο, σε απάντηση σε έναν εισερχόμενο Ping descriptor. Όμως μπορεί και περισσότερα από ένα Pong μπορούν να σταλούν σε απάντηση σε ένα Ping, ο οποίος 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 μύνημα PONG μπορεί να σταλθεί μόνο κατά μήκος της ίδιας διαδρομής από την οποία είχε έρθει το αρχικό μύνημα PING** (back-propagation). Για αυτό το λόγο, κάθε servent διατηρεί μια μνήμη (ID cache) όλων των ping IDs (Κάθε Ping και Pong μύνημα περιέχει ένα παγκοσμίως μοναδικό αναγνωριστικό, Globally Unique Identifier (GUID)) που έχει δει σε ένα είδος routing table, με πληροφορίες σχετικά με το ποιά συνδέση το παρέδωσε. 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 περιγραφείς, που οφείλονται συνήθως στους βρόχους (loops) στην ad-hoc τοπολογία δικτύων.

Τα δεδομένα από τα μυνήματα 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 pongs σαν απάντηση, συμπεριλαμβανομένων και των pongs από άλλους κόμβους.

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, και το πρωτόκολλο Gnutella επομένως συστήνει έντονα την ελαχιστοποίηση του αριθμού και τη συχνότητα των μυνημάτων ping που στέλνονται από οποιοδήποτε client.



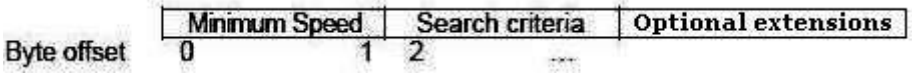
Σχημ. 3.5. Ανταλλαγή μυνημάτων PING και PONG.

3.4.3. Αναζήτηση αρχείου.

3.4.3.1. Μυνήμα QUERY.

Τα μυνήματα Query στέλνονται για την λειτουργία της αναζήτησης δεδομένων στο δίκτυο Gnutella. Με αυτό το μήνυμα ζητείται ένα συγκεκριμένο αρχείο. Εκτός από το όνομα του αρχείου, περιέχει επίσης μερικούς περιορισμούς στη ταχύτητας μεταφοράς. Παραδείγματος χάριν, μπορεί να ζητήσει τουλάχιστον 10 kbyte/sec. Το μέγεθος του μηνύματος είναι στις περισσότερες περιπτώσεις κάτω από 200 bytes.

Ένα μύνημα Query έχει τη παρακάτω δομή[8]:



Σχημ. 3.6. Query Descriptor

- Minimum Speed: Η ελάχιστη ταχύτητα (σε kb/second) των servents που πρέπει να αποκριθούν σε αυτό το μήνυμα. Ένας servent λαμβάνοντας έναν Query descriptor με έναν Minimum Speed του x kb/s πρέπει μόνο να ανταποκριθεί με ένα QueryHit εάν είναι σε θέση να επικοινωνήσει με ταχύτητα >= x kb/s

- Search Criteria: Η μηδενική τιμή (δηλ. 0x00) ολοκληρώνει το string, για το κριτίριο αναζήτησης. Το μέγιστο μήκος αυτού του string είναι προδιορισμένο από το πεδίο Payload_Length του Descriptor Header.

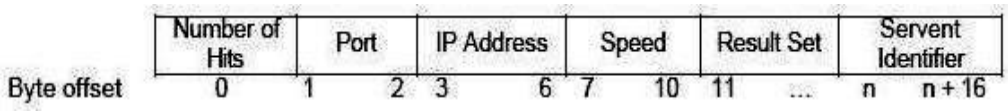
- Optional extensions: Αυτό το πεδίο είναι διαθέσιμο *μόνο στην έκδοση Gnutella v0.6*. Οι επιτρεπόμενοι τύποι επέκτασης ειναι:
  HUGE (Hash/URN Gnutella Extensions), XML ή GGEP.

3.4.3.2. Μυνήμα QUERYHIT.

Όταν ένας servent λαμβάνει έναν Query descriptor, ανταποκρίνεται στέλνοντας ένα QueryHit εάν υπάρχει αντιστοιχία στο σύνολο των τοπικών δεδομένων του, δηλαδή αν κάποιος servent έχει τα δεδομένα που ζητήθηκαν στο query.

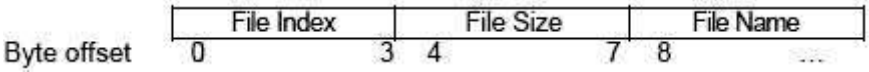
Αυτό το μήνυμα επιστρέφει μια ή περισσότερες πηγές για ένα αρχείο που ζητήθηκε από ένα query. Το σημαντικότερο είναι ότι φέρει τις IP διευθύνσεις όλων των κόμβων αυτών, τη ταχύτητα σύνδεσής τους και ένα προσδιοριστικό αρχείων. Αυτό το προσδιοριστικό χρησιμοποιείται όταν ζητήθει το αρχείο μέσω HTTP για να αποθηκευτεί..

Ένας Query descriptor έχει τη παρακάτω δομή[8]:



Σχημ. 3.7. QueryHit Descriptor

- Number of Hits: Ο αριθμός των query hits που βρίσκονται στο result set. (βλ παρακάτω)
- 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 Descriptor Header. Είναι ένα σύνολο απαντήσεων στην αντίστοιχη ερώτηση (Query). Αυτό το σύνολο περιλαμβάνει τα στοιχεία Number_of_Hits, κάθε ένα με την ακόλουθη δομή:



Σχημ. 3.8. Number of Hits

- File Index: Ένας αριθμός ο οποίος προσδιορίζει το κάθε file matching με την αντίστοιχη ερώτηση. 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.

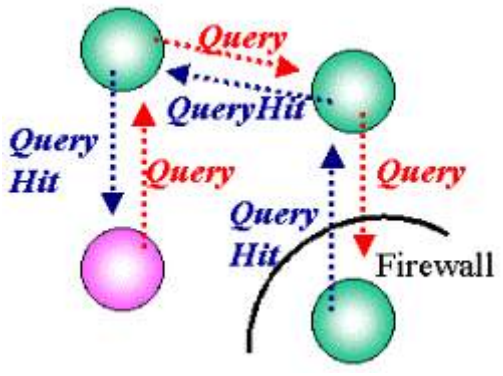
- File Name: Ο double-null (δηλ. 0x0000) με τον οποίο τελειώνει το όνομα του αρχείου του οποίου δείκτης είναι File_Index.

- Στην *έκδοση Gnutella v0.6* υπάρχουν τρία επιπλέον πεδία τα: 11ο byte: Extension blocks (HUGE, GGEP, plain), 12ο byte: Recommended EQHD block και 13ο byte: Private vendor specific data. Το EQHD block περιέχει κάποιες επιπρόσθετες πληροφορίες

- Servent Identifier: Ένα 16-byte string που προσδιορίζει μεμονωμένα (μοναδικά) τον αποκρινόμενο servent στο δίκτυο. Αυτό είναι συνήθως κάποια λειτουργία της διεύθυνσης δικτύου του servent. Ο Servent Identifier συμβάλλει στη λειτουργία του Push Descriptor (βλ. παρακάτω).

Τα μυνήματα QueryHit στέλνονται μόνο σε απάντηση σε έναν εισερχόμενο μύνημα Query. 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).

Το πεδίο Descriptor_Id που βρίσκεται στο Descriptor Header για το QueryHit πρέπει να περιέχει την ίδια τιμή με αυτόν του αντίστοιχου (συσχετισμένου) Query descriptor. Αυτό επιτρέπει σε έναν servent να προσδιορίσει τους περιγραφείς QueryHit που συνδέονται με τους περιγραφείς Query που ο servent δημιούργησε.



Σχημ. 3.9. Ανταλλαγή μυνημάτων QUERY και QUERYHIT.

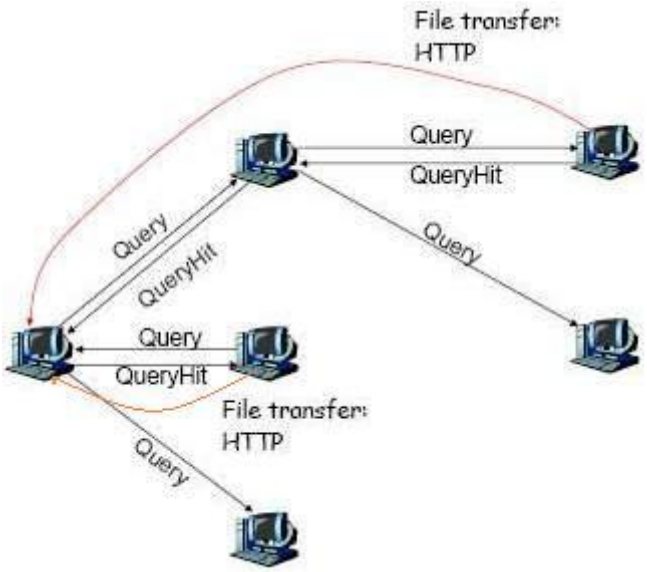
3.4.3.3. Περιγραφή μηχανισμού αναζήτησης ενός αρχείου.

Ο βασικός μηχανισμός αναζήτησης που υιοθετείται από το Gnutella είναι η περιορισμένη πλημμύρα (*limited flooding*). Στην πλημμύρα, ένας ομότιμος, που αναζητεί ένα αρχείο, εκδίδει ένα ερώτημα και το στέλνει σε όλους τους ομότιμους γείτονές του. Ο κόμβος που λαμβάνει το ερώτημα το διαβιβάζει σε όλους τους δικούς γείτονες εκτός από το κόμβο από τον οποίο προήλθε το μύνημα. Με αυτόν τον τρόπο, μια ερώτηση διαδίδεται (propagated) μέχρι ένα προκαθορισμένο αριθμό αλμάτων (TTL) από τον αρχικό ομότιμο. Το TTL που εφαρμόζεται από το Gnutella είναι γενικά για τις δημοφιλείς αναζητήσεις. Εντούτοις, TTL είχε εισαχθεί για τις σπάνιες αναζητήσεις.

3.4.4. Λήψη αρχείων.

Όταν στέλνεται ένα μύνημα Query, αυτό διανέμεται σε έναν τεράστιο αριθμό ομοτίμων. Κάθε ομότιμος ελέγχει, τα προς ανταλλαγή δεδομένα του, ώστε να διαπιστωθεί αν έχει αυτό που ζητείται. Στη συνέχεια, όσοι διαθέτουν αυτό το αρχείο, στέλνουν ένα πακέτο QueryHit μέσω της ίδιας διαδρομής από όπου το πακέτο Query είχε σταλθεί αρχικά.

Το πακέτο αυτό περιέχει τη διεύθυνση IP και το GUID του υπολογιστή που έχει τα δεδομένα όπως και πληροφορίες για το αρχείο που ταίριαξε με την ερώτηση. Όταν ληφθεί το πακέτο QueryHit τότε μέσω του Gnutella client δίνεται η επιλογή αποθήκευσης. **Για τη λήψη του αρχείου χρησιμοποιείται η GET μέθοδος του πρωτοκόλλου HTTP.** Στη συνέχεια ξεκινάει μια απευθείας σύνδεση HTTP μεταξύ των ομοτίμων που έστειλαν το QueryHit και αυτού που υπέβαλε την ερώτηση.



Σχημ. 3.10. Διαδικασία λήψης αρχείου.

Αναλυτικά:

Απο τη στιγμή που ένας servent λαμβάνει ένα μύνημα QueryHit, μπορεί να αρχίσει η άμεση λήψη (direct download) ενός από τα αρχεία που περιγράφονται από το Result Set. 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 GET request string στον target server (ο κάθε ομότιμος λειτουργεί και ως client και ως 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 περιγραφέα QueryHit, περιείχε τα:

File Index *2468*
File Size *4356789*
File Name *Song.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 κεφαλίδες HTTP 1.0 όπως:

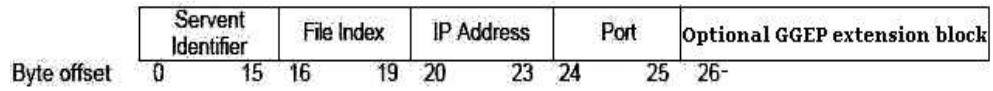
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type: application/binary\r\n
Content-length: 4356789\r\n
\r\n

Ακολουθούν τα δεδομένα των αρχείων, και 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.

3.4.4.1. Μύνημα PUSH.

Ένα μύνημα Push έχει τη παρακάτω δομή[8]:



Σχημ. 3.11. Push Descriptor

- Servent Identifier: Ένα 16-byte string που προσδιορίζει μοναδικά τον servent στο δίκτυο από τον οποίο ζητήθηκε να προωθήσει (push) 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 περιγραφέα QueryHit. Αυτό επιτρέπει στον παραλήπτη ενός αιτήματος ώθησης (push request) να καθορίσει ή όχι εάν 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 περιγραφέα QueryHit.

- IP Address: Η IP Address 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.

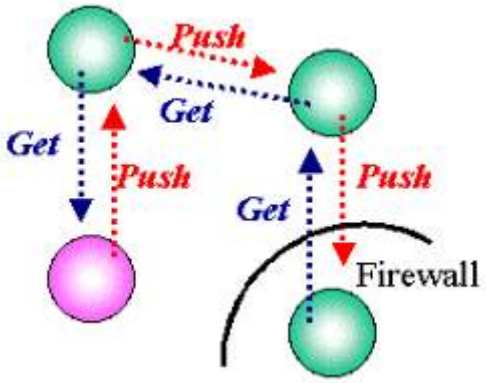
- Optional GGEP extension block: Υπάρχει *μόνο στο πρωτόκολλο Gnutella v0.6 .*

Όταν ένας servent λαμβάνει έναν Push descriptor, μπορεί να ενεργήσει κατόπιν του αιτήματος προώθησης εάν και μόνο εάν το πεδίο servent_Identifier περιέχει την τιμή του δικού του servent identifier. Το πεδίο Descriptor_Id στο Descriptor Header του Descriptor Header δεν πρέπει να περιέχει την ίδια τιμή με αυτόν του σχετικού περιγραφέα QueryHit, αλλά πρέπει να περιέχει μια νέα τιμή που παράγεται από τον αλγόριθμο παραγωγής Descriptor_Id του servent.

3.4.4.2. Firewalled Servents.

Η εγκαθίδρυση μιας άμεσης σύνδεσης σε ένα Gnutella servent, στη προσπάθεια να αρχίσει η λήψη ενός αρχείου, δεν είναι πάντοτε εφικτή. **Ο servent μπορεί, για παράδειγμα, να είναι πίσω από ένα firewall που δεν επιτρέπει τις εισερχόμενες συνδέσεις στο αντίστοιχο port 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[8].



Σχημ. 3.12. Διαδικασία λήψης αρχείου για firewalled servent.

Αναλυτικά:

Εάν η σύνδεση, από τον firewalled servent προς στο servent που άρχισε το Push request , είναι εφικτή, τότε ο firewalled servent πρέπει 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 Push request. 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, που είχε κάνει αίτηση για το Push, λαμβάνοντας την κεφαλίδα του GIV request, εξάγει τα <File Index> και <File Name> από την κεφαλίδα για να κατασκευάσει ένα HTTP GET αίτημα 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

Στη συνέχεια η διαδικασία λήψης του αρχείου συνεχίζεται κατά το γνωστό τρόπο, όπως έχει ήδη αναφερθεί.

3.4.4.3. Μύνημα BYE.

Το μήνυμα BYE εμφανίζεται μόνο στο Gnutella 0.6 και χρησιμοποιείται για να ενημερώσει το servent, ή servents, ότι ένας κόμβος που συνδέεται με αυτόν, ετοιμάζεται να αποσυνδεθεί από το δίκτυο, και θα κλείσει η σύνδεση. Από την στιγμή που η εφαρμογή αυτού του μηνύματος είναι προαιρετική, μια ειδική κεφαλίδα πρέπει να σταλεί 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 με τη τιμή 1, ώστε να αποφευχθεί μια κατά λάθος μετάδοση του μηνύματος.

Με την παραλαβή του μηνύματος ο servent κλείνει τη σύνδεση αμέσως. 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 σταλθεί μετά από το μήνυμα αυτό[9].

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: Description string

3.4.5. Δρομολόγηση μυνημάτων.

Η peer-to-peer φύση του δικτύου Gnutella απαιτεί από τους servents να δρομολογούν την κυκλοφορία δικτύων (queries, query replies, push requests, κ.λπ.) με κατάλληλο τρόπο. Ένας καλά συμπεριφερόμενος Gnutella servent θα δρομολογεί τους περιγραφείς πρωτοκόλλου σύμφωνα με τους ακόλουθους κανόνες[8]:

1. **Τα μηνύματα Pong μπορούν να σταλθούν μόνο κατά μήκος της ίδιας διαδρομής που μετέφερε τον εισερχόμενο περιγραφέα Ping.** Αυτό εξασφαλίζει ότι μόνο εκείνοι οι servents που δρομολόγησαν το μύνημα Ping θα λάβουν το μύνημα Pong σαν απάντηση. Ένας servent που λαμβάνει έναν περιγραφέα Pong με Descriptor ID = *n*, αλλά δεν έχει δει έναν μύνημα Ping με το ίδιο Descriptor ID, θα πρέπει να απομακρύνει το μύνημα Pong από το δίκτυο.

2. Τα **μηνύματα QueryHit μπορούν να σταλθούν μόνο κατά μήκος της ίδιας διαδρομής από την οποία ήρθε το εισερχόμενο Query descriptor.** 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 Query descriptor με το ίδιο Descriptor ID θα πρέπει να απομακρύνει τον περιγραφέα QueryHit από το δίκτυο.

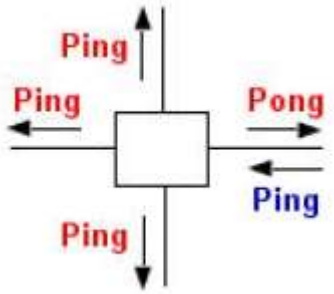
3. **Τα μηνύματα Push μπορούν μόνο να σταλθούν κατά μήκος της ίδιας διαδρομής από την οποία ήρθε το QueryHit.** Αυτό εξασφαλίζει ότι μόνο εκείνοι οι servents που δρομολόγησαν τον περιγραφέα QueryHit θα μπορούν να δουν τον περιγραφέα Push. Ένας servent που λαμβάνει μύνημα Push με Servent_Identifier = *n*, αλλά δεν έχει δει περιγραφέα QueryHit με το ίδιο προσδιοριστικό, πρέπει να απομακρύνει το πακέτο Push από το δίκτυο. Οι Push descriptors δρομολογούνται από τον Servent_Identifier, όχι από τον Descriptor_Id.

4. **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.

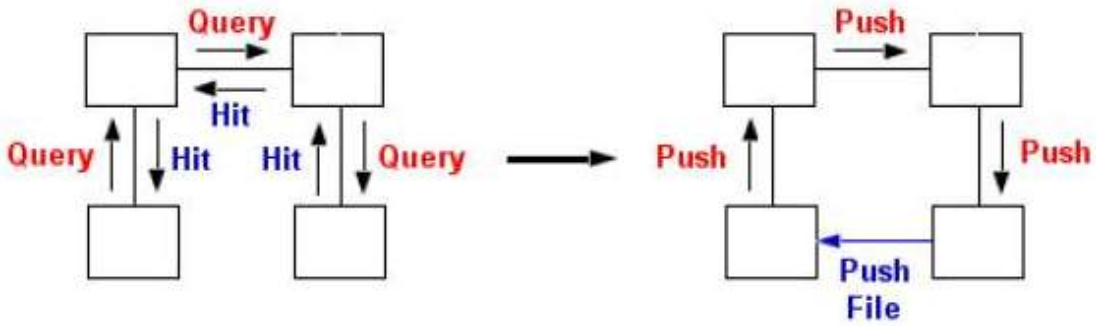
5. **Ένας servent θα μειώσει το πεδίο TTL του descriptor header, και θα αυξήσει το Hops πεδίο του, πρίν να διαβιβάσει το μύνημα σε οποιοδήποτε γειτονικό του servent.** Εάν, μετά από τη μείωση της κεφαλίδας του πεδίου TTL, ο τομέας TTL βρεθεί να είναι μηδέν, ο περιγραφέας δεν διαβιβάζεται κατά μήκος οποιασδήποτε σύνδεσης και απομακρύνεται από το δίκτυο.

6. Ένας servent που λαμβάνει έναν πακέτο με το ίδιο Payload Descriptor και Descriptor ID όπως αυτό που έχει λάβει πριν, πρέπει να προσπαθήσει να αποφύγει να διαβιβάσει τον περιγραφέα σε οποιοδήποτε συνδεδεμένο servent. Οι προοριζόμενοι παραλήπτες έχουν λάβει ήδη έναν τέτοιο περιγραφέα, και στέλνοντας τον πάλι, σπαταλά μόνο το εύρος ζώνης δικτύων.

Στα παρακάτω σχήματα φαίνεται πως γίνεται η δρομολόγηση των περιγραφέων του πρωτοκόλλου Gnutella.



Σχημ. 3.13. Ping/Pong δρομολόγηση.



Σχημ. 3.14. Query/QueryHit/Push δρομολόγηση.

3.5. Το πρωτόκολλο Gnutella v0.6.

3.5.1. Εισαγωγή – Αιτίες ανάπτυξης του νέου πρωτοκόλλου.

Τα πλήρως αποκεντρικοποιημένα και μη δομημένα P2P δίκτυα όπως το Gnutella, είναι ίσως τα δημοφιλέστερα overlay networks για τη διανομή αρχείων. Η απουσία μιας καθορισμένης δομής και ενός κεντρικού ελέγχου καθιστά τέτοια συστήματα πιό ανθεκτικά και ιδιαίτερα "αυτοθεραπευόμενα" έναντι των δομημένων συστημάτων[13][14].

Ωστόσο, το κύριο πρόβλημα σε αυτό το είδος δικτύων είναι η δυνατότητα κλιμάκωσης (scalability). 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** (συνήθως ελέγχονταν περίπου 10.000 κόμβοι). Συνεπώς δεδομένου ότι αυτά τα δίκτυα γίνονται ολοένα και δημοφιλέστερα, **η ποιότητα της υπηρεσίας υποβιβαθμίζεται** με ραγδαίο ρυθμό[26].

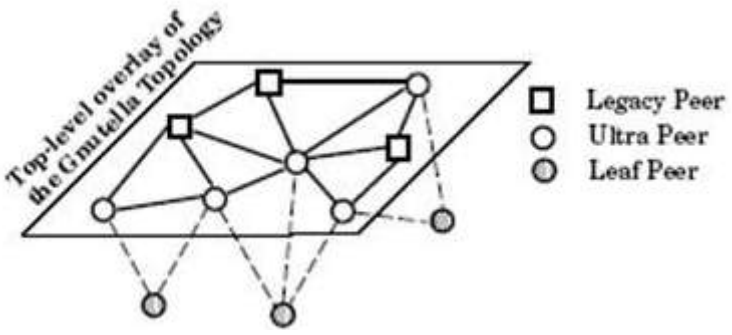
Για να γίνει έτσι το δίκτυο κλιμακώσιμο, το Gnutella αναβαθμίζει συνεχώς κάποια χαρακτηριστικά γνωρίσματά του και εισάγει νέες έννοιες. Έτσι οι κύριοι υπεύθυνοι για την ανάπτυξη των Gnutella clients, μαζί με υπόλοιπους συμμετέχοντες στο, ανοικτού κώδικα, πρωτόκολλο Gnutella, πρόσθεσαν μια σειρά από τροποποιήσεις στο αρχικό πρωτόκολλο, όπως και κάποιες πολύ ουσιαστικές αλλαγές που καλούνται, για κάποιο λόγο, "v0.6". (Gnutella v0.4 Μάρτιος 2000 - Gnutella v0.6 Οκτώβριος 2000).

Όλες αυτές οι *βελτιώσεις* 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 network*, 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) (αναφορά στη §3.5.3), αλλά και ο μηχανισμός ***Pong-caching*** για καλύτερη διαχείρηση του εύρους ζώνης και μείωση της περιττής κίνησης στο δίκτυο. Μια από τη σημαντικότερη τροποποίηση στη τοπολογία του μη δομημένου δίκτυου έγινε με την παρουσίαση της έννοιας των ultrapeers και leafpeers δίνοντας μία ιεραρχική δομή στο δίκτυο, μαζί με μια two-tier τοπολογία δικτύων[14]. Να αναφερθεί ότι **σχεδόν το 99% πλέον των Gnutella clients χρησιμοποιούν το πρωτόκολλο Gnutella v0.6** .

3.5.2. Τοπολογία δικτύου.

Το Gnutella 0.6, ανήκει στη κατηγορία των **υβριδικών P2P συστημάτων**. Πρόκειται για **ένα two-tier overlay network** όπου στο ένα επίπεδο, ονομαζόμενο ως 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 οποιαδήποτε συνδέσεις με τους leaf-peers. (βλ. παρακάτω σχήμα) [13][14][15].



Σχημ. 3.15. Two-tier τοπολογία, στο Gnutella 0.6 .

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 (leaf nodes). Εντούτοις, οι ultrapeers συνδέονται μεταξύ τους χρησιμοποιώντας ένα παρόμοιο πρωτόκολλο με αυτό που χρησιμοποιείται στο Gnutella 0.4, επιτρέποντας στους ultrapeers να προωθούν query requests στους γειτονικούς τους ultrapeers, επεκτείνοντας με αυτόν τον τρόπο την επικοινωνία στο δικτύου. Να σημείωσουμε επίσης, ότι **το bootstrapping γίνεται πλέον μέσω των ultrapeers.**

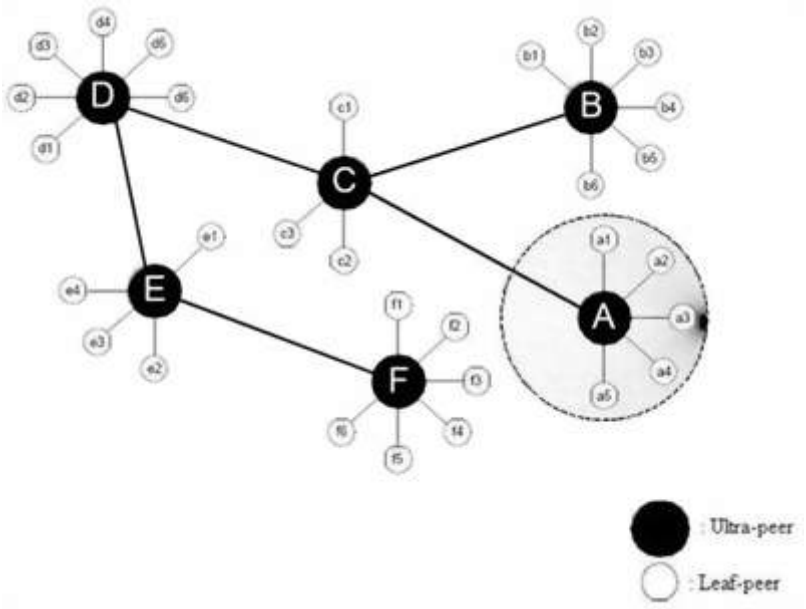
Σε ένα υβριδικό δίκτυο υπάρχουν κάποιοι κανόνες και περιορισμοί όσον αφορά τις συνδέσεις που λαμβάνουν χώρα στο δίκτυο[16]:

i. Οι leaf nodes δεν έχουν εξερχόμενες συνδέσεις με άλλους leaf nodes, δεν υπάρχει καμία άμεση σύνδεση μεταξύ δύο οποιωνδήποτε leaf-peers στο overlay network
ii. Οι leaf nodes δεν δέχονται εισερχόμενες συνδέσεις.
iii. Οι ultra-peers πραγματοποιούν εξερχόμενες συνδέσεις μόνο με άλλους super peers.

Για την αποφυγή των σημείων συμφόρησης όσον αφορά την ενημέρωση-ανανέωση και την διερεύνηση του καταλόγου, οι **ultrapeers δέχονται μόνο έναν περιορισμένο αριθμό συνδέσεων από τους leaf nodes**.

Τέλος, για την επιλογή τους, **οι ultrapeers απαιτούνται να έχουν και μεγάλο εύρος ζώνης και μεγάλη ικανότητα επεξεργασίας (μεγάλη υπολογιστική ισχύ) προκειμένου να αποφευχθούν τα προβλήματα κλιμάκωσης** στην επεξεργασία των ερωτήσεων από τους γειτονικούς ultrapeers. Επιλέγονται από το πρωτόκολλο (μέσω του 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 0.4[15] .



Σχημ. 3.16. Αρχιτεκτονική του Gnutella 0.6

3.5.3. PONG caching.

Τα μυνήματα PING και PONG διαμόρφωναν ένα σημαντικό ποσοστό της κυκλοφορίας στο Gnutella 0.4, έτσι για να μειωθεί αυτή η κατανάλωση του εύρους ζώνης, έχουν εισαχθεί μηχανισμοί caching για τα μηνύματα PONG, όπου περιλαμβάνουν την εφαρμογή του Gnutella PONG cache και το σύστημα PING reduction.

Στο δίκτυο Gnutella v0.6, **μόνο οι ultrapeers δέχονται συνδέσεις από άλλους υπολογιστές**. Τα μυνήματα PONGs κατά κάποιο τρόπο "διαφημίζουν" τους υπολογιστές που ένας ομότιμος μπορεί να συνδεθεί, έτσι **όλα τα PONGs που θα κυκλοφορούν στο δίκτυο θα πρέπει να αφορούν τους ultrapeers**.

Κατά αυτό το τρόπο, **μόνο οι ultrapeers διατηρούν πίνακα PONG μηνυμάτων** (ο πίνακας ανανεώνεται περιοδικά). Αντί ο ultrapeer να προωθεί τα PING μυνήματα από τους leafpeers του, στους υπόλοιπους ultrapeers, τους στέλνει σαν απάντηση τα pongs (τυπικά 10) που έχει αποθηκεύσει στην ανταλλαγή των PING-PONG μηνυμάτων με άλλους ultrapeers. Συνεπάγεται, πολύ λιγότερα PINGs στο δίκτυο.

Ένας ultrapeer του LimeWire διαβιβάζει τα PONGs στους leafpeers του, ακόμη και αν δεν έχουν στείλει PINGs, για να εξασφαλιστεί ότι θα ανακαλύψουν αρκετούς ultrapeers[18][19].

Η επίδραση της προσωρινής αποθήκευσης PONG (PONG caching) έχει δύο πτυχές: i) τη **μείωση του όγκου της κίνησης στο δίκτυο** και ii), καθώς η κάθε προσωρινή αποθήκευση μεταφέρει PONGs από τους ομότιμους κατά μήκος του δικτύου, **οι απαντήσεις τείνουν να είναι περισσότερο αντιπροσωπευτικές**[26]**.**

3.5.4. Τεχνικές αναζήτησης.

Το δίκτυο ακολουθεί την τεχνική: **limited flood based query search**. Ένα query ενός ultra-peer προωθείται στους leaf-peers του με TTL (0) και σε όλους τους ultra-peer γείτονές του με TTL μειωμένο κατά ένα, μόνο όταν (TTL > 0). Ένας leaf-peer δεν διαβιβάζει οποιαδήποτε λαμβανόμενο query. Από την άλλη πλευρά, **οι ultra-peers πραγματοποιούν το query searching εκ μέρους των leaf-peers τους**. Το query ενός leaf-peer στέλνεται αρχικά στους συνδεδεμένους με αυτόν ultra-peers, οι οποίοι (ultra-peers) στη συνέχεια το προωθούν (limited flood based query search) ταυτόχρονα την ερώτηση στους γειτονικούς τους ultra-peers μέχρι έναν περιορισμένο αριθμό αλμάτων (hops). Να αναφερθεί ότι **η μέθοδος της περιορισμένης πλυμμήρας εφαρμόζεται πλέον μόνο μεταξύ των ultra-peers**[14].

Το Gnutella 0.6 έχει υιοθετήσει έναν αριθμό άλλων τεχνικών με σκοπό την μείωση της κίνησης (από την άποψη των περιττών μυνημάτων) στο δίκτυο, και της

αποτελεσματικότερης αναζήτησης. Πιο αξιοσημείωτες είναι το QRP (Query Routing Protocol) και το DQ (Dynamic Querying) [10][14].

Dynamic Query (DQ): Το Gnutella 0.6 ενσωματώνει το dynamic querying με περιορισμένη πλημμύρα (limited flooding) ως τεχνική αναζήτησης ερώτησης. Στο dynamic querying, 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 εκείνη την ερώτηση. Ο ultra-peer μπορεί να σταματήσει την προώθηση της ερώτησης σε οποιοδήποτε βήμα εάν λάβει ένα ικανοποιητικό αριθμό από query hits, το οποίο γεγονός, **μειώνει σημαντικά το ποσό της κίνησης που προκαλείται από δημοφιλείς αναζητήσεις**. Συνεπώς το dynamic querying θα χρησιμοποιήσει το TTL (3) μόνο για τις σπάνιες αναζητήσεις.

Query Routing Protocol (QRP): Μια άλλη τεχνική του Gnutella 0.6, είναι το Query Routing Protocol. Ένας ultra-peer, προωθεί ένα ερώτημα μόνο στους συνδεδεμένους με αυτόν leaf-peers, οι οποίοι ενδέχεται να έχουν ένα αποτέλεσμα, έχοντας σαν αποτέλεσμα την **αποδοτικότερη αναζήτηση στα πιο σπάνια αρχεία**. Ένας leaf-peer δημιουργεί ένα hash table όλων των αρχείων που μοιράζεται και το στέλνει σε όλους τους ultra-peer/s 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 query hits, και που θα έχουν δηλαδή το αρχείο. Έτσι η αναζήτηση πραγματοποιείται μόνο στο ultra-peer layer, δεδομένου ότι οι ultra-peers περιέχουν τους δείκτες των "παιδιών" τους. **Ο ultrapeer αποφασίζει ποιο ερώτημα θα προωθήσει, χρησιμοποιώντας το Query Routing Protocol.**

3.5.5. Βασικό πρωτόκολλο χειραψίας στο Gnutella v 0.6 .

Πολλά προγράμματα (clients) χρησιμοποιούνται για τη πρόσβαση στο δίκτυο Gnutella v0.6 (π.χ. Limewire, Bearshare, gtk-Gnutella). Μέσω της χειραψίας, ένας ομότιμος εγκαθιστά τη σύνδεση-συνδέσεις με ένα ή περισσότερους ultra-peers. 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 δίσκο του 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 κεφαλίδων (groups of headers). Τα βήματα της χειραψίας διαμορφώνονται έπειτα ως εξής[14]:

1. Το πρόγραμμα (ομότιμος) που αρχικοποιεί τη σύνδεση, στέλνει την πρώτη ομάδα κεφαλίδων, η οποία αναφέρει στο απομακρυσμένο πρόγραμμα-ομότιμο για τα χαρακτηριστικά γνωρίσματά του, και τη κατάσταση, που υποδηλώνει τον τύπο γείτονα (leaf ή ultra) που θέλει να είναι.

2. Το πρόγραμμα που λαμβάνει τη σύνδεση αποκρίνεται με μια δεύτερη ομάδα κεφαλίδων που μεταβιβάζει ουσιαστικά το μήνυμα εάν συμφωνεί με την πρόταση του ομοτίμου που ξεκίνησε τη σύνδεση, ή όχι.

3. Τέλος, ο αρχικός ομότιμος στέλνει μια τρίτη ομάδα κεφαλίδων για να επιβεβαιώσει και να εγκαθιδρύσει τη σύνδεση.

3.5.5.1. Σύγκριση με το αντίστοιχο του Gnutella v 0.4 .

Το πρωτόκολλο χειραψίας βελτιώθηκε σε ένα 3-way πρωτόκολλο χειραψίας Gnutella v.0.6[16] . Το αρχικό πρωτόκολλο v.0.4 ήταν το ακόλουθο:

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. Και 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 πρωτόκολλο Gnutella v0.6 επιτρέπει στους Servents να ανταλλάζουν πρόσθετες πληροφορίες και είναι γενικά ασφαλέστερο. Η περιγραφή του δίνεται παρακάτω[16]:

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 κεφαλίδες ικανότητας (capability headers) 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 κεφαλίδες 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. Και ο client και ο server τέλνουν δυαδικά μηνύματα κατά βούληση, χρησιμοποιώντας τις πληροφορίες που λαμβάνονται στα βήματα 3 και 5.

Καμία χειραψία κεφαλίδων δεν μπορεί να χρησιμοποιηθεί στη χειραψία. v 0.4 .

Εδώ είναι ένα παράδειγμα της ροής πληροφορίας:

Client	Server
GNUTELLA CONNECT/0.6<cr><lf> Peer-Type: leaf<cr><lf> <cr><lf>	
	GNUTELLA/0.6 200 OK<cr><lf> Peer-type: super-peer<cr><lf> <cr><lf>
GNUTELLA/0.6 200 OK<cr><lf> <cr><lf>	

3.5.6. Gnutella crawler.

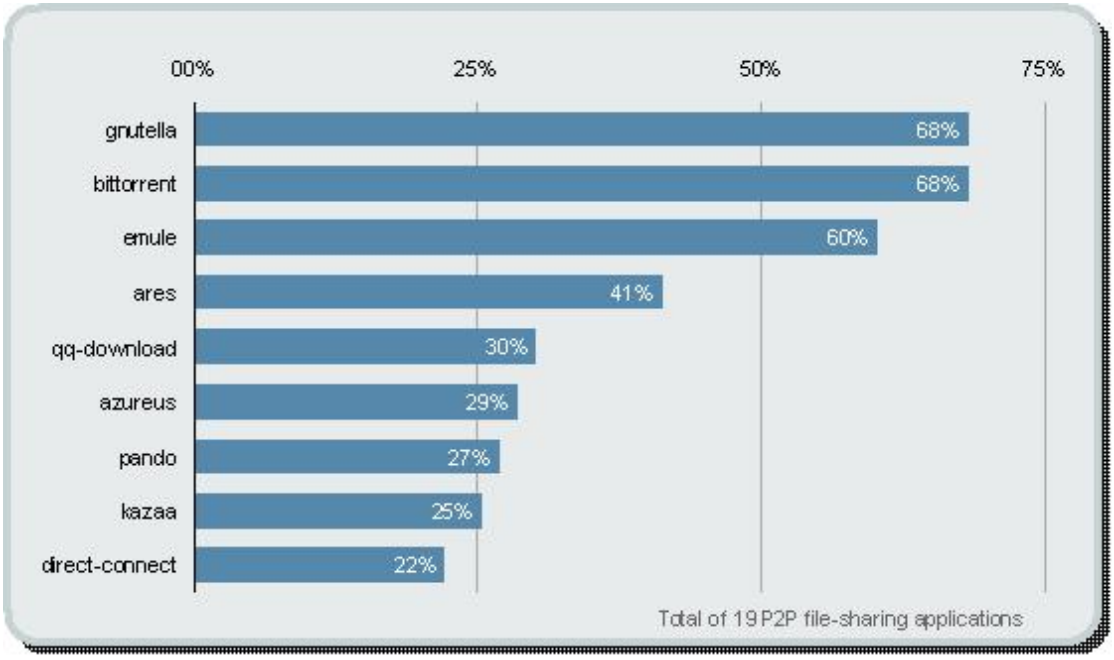
Ο **Gnutella crawler** είναι ένα πρόγραμμα λογισμικού που χρησιμοποιείται για τη **συλλογή στατιστικών πληροφοριών σχετικά με το δίκτυο** Gnutella, όπως τον αριθμό των χρηστών, και τη γεωγραφική κατανομή των ομοτίμων. Οι αρχικοί crawlers χρησιμοποιούσαν Ping/Pong μηνύματα για να ανακαλύψουν τους hosts που ήταν συνδεμένοι με το δίκτυο. Αν και αυτή η μέθοδος είναι ακόμα χρησιμοποιήσιμη, είναι πάρα πολύ αργή να συλλάβει αρκετά στοιχεία για μια επισκόπηση στη τοπολογία του δικτύου Gnutella καθώς απαιτεί την αρχικοποίηση των πλήρων συνδέσεων Gnutella.

Μια επέκταση έχει προστεθεί στο πρωτόκολλο Gnutella ώστε να επιτρέπει στους crawlers να έχουν γρήγορη πρόσβαση γρήγορα στους ultrapeers. Αυτή τη στιγμή, δεν υπάρχει κάποιος δημόσιος προσβάσιμος, online crawler στο δίκτυο Gnutella, δεδομένου ότι ο τελευταίος που φιλοξενούταν από τον Lime Wire LLC έχει αποσυρθεί[20].

ΚΕΦΑΛΑΙΟ 4ο

4.1. Εισαγωγή – Σκοπός του κεφαλαίου.

Τα διομότιμα δίκτυα Bittorrent και Gnutella είναι ίσως τα σημαντικότερα P2P δίκτυα που υπάρχουν αυτή τη στιγμή. Σε μία πρόσφατη έρευνα που πραγματοποιήθηκε[21] στην οποία συμμετείχαν 63 οργανισμοί, βρέθηκε ότι τα δίκτυα αυτά ήταν τα πιο ευρέως διαδεδομένα με ποσοστό 68%, όπως φαίνεται και στο παρακάτω σχήμα:



Σε αυτό το κεφάλαιο θα πραγματοποιηθεί μια συγκριτική μελέτη των διομότιμων δικτύων Bittorrent και Gnutella. Σκοπός αυτού του κεφαλαίου είναι να αναφερθούν οι ομοιότητες και διαφορές των συστημάτων αυτών σε θέματα όπως η τοπολογία των δικτύων, στους μηχανισμούς-πρωτόκολλα αναζήτησης και αποθήκευσης των αρχείων, αλλά και σε ζητήματα τα οποία είναι μέγιστης σημασίας στα διομότιμα δίκτυα όπως είναι τα θέματα της ασφάλειας (security), κλιμάκωσης (scalability), free-riding και ανθεκτικότητας (robustness).

4.2. Τοπολογία δικτύου.

4.2.1. Σύγκριση BitTorrent με Gnutella v0.4. .

Βαθμός αποκεντρικοποίησης.

Το δίκτυο **Bittorrent** έχει ως κύριο χαρακτηριστικό του, την παρουσία μιας κεντρικής οντότητας ονομαζόμενη ως tracker (παρ'όλη την αποκεντρικοποιημενη φύση του ίδιου του πρωτοκόλου). Λόγω της παρουσίας του, το δίκτυο κατατάσσεται στην κατηγορία των **ημικεντρικοποιημένων (semi-centralized) P2P δικτύων**. Παρόλο που πρόκειται για έναν software server, το δίκτυο δεν αποκαλείται σαν κεντρικοποιημένο (π.χ. Napster), για το λόγω ότι κάθε ομότιμος είμαι ικανός να θέσει σε λειτουργία έναν τέτοιο server.

Σε αντίθεση με το Bittorrent, το **Gnutella v0.4** αποτελεί ένα **πλήρως αποκεντρικοποιημένο δίκτυο (ή pure P2P).** Όλες οι απαραίτητες λειτουργίες του δικτύου καθορίζονται αποκλειστικά μεταξύ των ομοτίμων και του πρωτοκόλλου, και χωρίς την παρουσία κάποιας κεντρικής οντότητας που να εκτελεί χρέη "ρυθμιστή".

Δομή δικτύου.

Τόσο το Bittorrent όσο και το Gnutella v0.4, ανήκουν στη κατηγορία των μη δομημένων δικτύων (**unstructured networks**). Αυτό σημαίνει ότι δεν υπάρχει μία αυστηρά ορισμένη δομή σε αυτά τα δίκτυα. Η τοποθέτηση των δεδομένων είναι απολύτως ανεξάρτητη από το overlay network και οι ομότιμοι συνδέονται απευθείας ο ένας με τον άλλον χωρίς να έχουν καμία πληροφορία για τα δεδομένα των υπόλοιπων ομοτίμων.

Να αναφέρουμε πως και στα δύο δίκτυα **οι ομότιμοι είναι ακριβώς ισοδύναμοι μεταξύ τους**, και όπως συμβαίνει σε όλα τα διομότιμα συστήματα, μπορούν να συνδέονται και να αποσυνδέονται από το δίκτυο ανά πάσα στιγμή. Επίσης, και στα δύο συστήματα, οι συνδέσεις των ομοτίμων γίνονται μέσω του πρωτοκόλου TCP.

Τέλος, τα αποκεντρωμένα δίκτυα όπως το Gnutella v0.4, διαθέτουν μεγαλύτερο βαθμό αυτονομίας ελέγχου των δεδομένων και των πόρων τους, όμως, λόγω έλλειψης μιας σφαιρικής άποψης στο επίπεδο συστήματος, είναι δύσκολο να προβλεφθεί η συμπεριφορά του συστήματος αυτού.

4.2.2 Σύγκριση BitTorrent με Gnutella v0.6. .

Βαθμός αποκεντρικοποίησης.

Το Gnutella v0.6 ανήκει στην κατηγορία των υβριδικών P2P συστημάτων. Αυτό οφείλεται στην παρουσία των ultrapeers, δηλαδή ομότιμοι στους οποίους το σύστημα έχει αναθέσει κάποιες επιπλέον αρμοδιότητες, όπως π.χ. τον διαχειρισμό των queries από τους leafnodes με τους οποίους συνδέονται . Ουσιαστικά πρόκειται για δυναμικές κεντρικές οντότητες.

Με βάση το γεγονός αυτό, θα λέγαμε ότι παρουσιάζεται ομοιότητα με το Bittorrent, καθώς και τα δύο αυτά συστήματα δεν είναι πλήρως αποκεντρικοποιημένα. Παρ'όλα αυτά, ο ρόλος των ultrapeers και των trackers είναι τελείως διαφορετικός και αυτό οφείλεται στην φύση των πρωτοκόλλων τα οποία εξυπηρετούν.

Μία σημαντική διαφορά που παρατηρείται είναι ότι ενώ στο Bittorrent οι ομότιμοι είναι απολύτως ισοδύναμοι, κάτι τέτοιο δεν συμβαίνει στο Gnutella v0.6. Επιπλέον στο Bittorrent, κάθε ομότιμος μπορεί να συνδεθεί με έναν άλλον, με τη προυπόθεση φυσικά να ανήκουν στο ίδιο swarm, ενώ στο Gnutella v0.6 οι ομότιμοι δεν συνδέονται απευθείας μεταξύ τους, παρά μόνο με κάποιον ultrapeer.

Δομή δικτύου.

Τόσο το Bittorrent όσο και το Gnutella v0.6 ανήκουν στην κατηγορία των μη δομημένων, ή αδόμητων, δικτύων.

Σε αντίθεση με το Gnutella το οποίο σχηματίζει ένα, μεγάλης έκτασης δίκτυο επικάλυψης, το Bittorrent έχει ένα διαφορετικο overlay για κάθε αρχείο.

- Να αναφερθεί τέλος ότι **το Bittorrent είναι πιο απλό στην υλοποίηση**, και εξαιτίας της κεντρικής οντότητας υπάρχει σφαιρική άποψη για ολόκληρο το P2P δίκτυο, καθιστώντας το, **πιο ευέλικτο σε θέματα βελτιστοποίησης του overlay network**, οι τροποποιήσεις μπορούν να εφαρμοστούν ευκολότερα, χρησιμοποιώντας την σφαιρική γνώση του P2P δικτύου[22].

4.3. Συμμετοχή ομοτίμου στο δίκτυο.

Όπως έχει αναφερθεί στα προηγούμενα κεφάλαια, η διαδικασία με την οποία οι ομότιμοι εντάσσονται στο εκάστοτε δίκτυο, καλείται bootstrapping.

Στη σύγκριση των δύο διομότιμων συστημάτων πάνω στο θέμα αυτό, καταλήγουμε στα εξής συμπεράσματα:

- Τόσο το Bittorrent όσο και το Gnutella, 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 δύο κρίσιμα σημεία στον ακριβή τρόπο ένταξης των ομοτίμων στο δίκτυο, και που διαφοροποιεί τα δύο αυτά συστήματα.
    - Πρώτον, στο Gnutella, με την εφαρμογή του προγράμματος πραγματοποιείται **άμεση σύνδεση του ομοτίμου** με έναν τουλάχιστον ενεργό κόμβο (στο Gnutella v0.6 θα είναι ένας ultrapeer).
    - Στο Bittorrent ο ομότιμος θα πρέπει αναγκαστικά να **συνδεθεί με έναν tracker server**. Για πολλούς, αυτό αποτελεί και το αδύναμο σημείο του Bittorrent. 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 swarm.

- Δεύτερον, 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.
- Στο Bittorrent, μετά τη σύνδεση και την εγγραφή στο tracker, θα δωθεί στον ομότιμο μία **"έτοιμη" τυχαία λίστα με τους γείτονές** του, με τους οποίους θα πραγματοποιήσει τις συνδέσεις και την ανταλλαγή των δεδομένων. Στο Bittorrent, δεν υπάρχει η έννοια της "ανακάλυψης των γειτόνων".

4.4. Δεδομένα.

4.4.1. Δημοσίευση των δεδομένων.

Τα δίκτυα Bittorrent και Gnutella έχουν διαφορετικό τρόπο προσέγγισης πάνω στο θέμα της δημοσίευσης (**publishing**) των δεδομένων:

- BitTorrent: Η δημοσίευση των δεδομένων γίνεται με τη δημιουργία του .torrent αρχείου, και στη συνέχεια τη καταχώρηση του .torrent αρχείου σε ένα tracker server. Η δημοσίευση αρχίζει, απλά με την εφαρμογή του server. Το κάθε .torrent αρχείο είναι συγκεκριμένο για τα δεδομένα που δημοσιέυονται.
- Gnutella: Το Gnutella v0.4 δε χρειάζεται τη δημοσίευση των δεδομένων καθώς δεν υπάρχει κάποια μορφή κεντρικής οντότητας στο δίκτυο αυτό. Ο κάθε κόμβος διατηρεί τοπικά, τη λίστα με τα αρχεία που μοιράζεται. Στο Gnutella v0.6 οι leaf-peers παρέχουν μια λίστα με τα ονόματα των αρχείων που μοιράζονται, σε κάθε ultrapeer που έχουν συνδεθεί.

4.4.2. Μηχανισμός ανάζητησης δεδομένων.

Όσον αφορά το ζήτημα αναζήτησης δεδομένων, παρατηρούμε ότι τα πρωτόκολλα Bittorrent και Gnutella, δε παρουσιάζουν κάποια ομοιότητα. Εξάλλου, **το πρωτόκολλο Bittorrent, εστιάζει κυρίως στην αποτελεσματική προσκόμιση (fetching) και όχι στην αναζήτηση (searching) των δεδομένων.** Πρόκειται δηλαδή για ένα πρωτόκολλο ανταλλαγής αρχείων.

Όπως έχει αναφερθεί στο προηγούμενο κεφάλαιο, στο Gnutella v0.4 κάθε ομότιμος υποβάλει ερώτημα το οποίο προωθείται στους γείτονές του, οι οποίοι στη συνέχεια το προωθούν στους γειτονικούς τους κόμβους κοκ., εώς ότου μηδενιστεί η τιμή στο πεδίο TTL (περιορισμένη πλυμμήρα). Οι ομότιμοι που έχουν το αρχείο που ζητείται, στέλνουν ανάλογο μύνημα στον αποστολέα.

Στο Gnutella v0.6, η αναζήτηση των δεδομένων από τους ομοτίμους γίνεται μέσω των ultrapeers, με τους οποίους συνδέονται.

Στο Bittorrent, ο ομότιμος θα πρέπει μέσω κατάλληλων webserver να επισκεφτεί ιστοσελίδες με αποθηκευμένα .torrent αρχεία. Η αναζήτηση ουσιαστικά, γίνεται μέσω αυτών των σελίδων όπου ο χρήστης μπορεί να πληκτρολογήσει το όνομα του αρχείου που τον ενδιαφέρει. Μέσω της εφαρμογής ένος Bittorrent client, θα δωθεί μια τυχαία λίστα με τους ομότιμους που θα έχουν το αρχείο.

Γενικά πάνω στο θέμα αυτό, καταλήγουμε στα εξής συμπεράσματα που οφείλονται καθαρά στη φύση των δύο πρωτοκόλλων:

- Στο Gnutella, γίνεται προσπάθεια **μέσω των ομοτίμων και του δικτύου**, ώστε να βρεθεί το "κατάλληλο" αρχείο. Στο Bittorrent, **μέσω του ειδικού αρχείου .torrent** βρίσκονται και οι "κατάλληλοι" ομότιμοι. Ακριβώς για το λόγο ότι στο πρωτόκολλο Bittorrent δεν υπάρχει ενσωματωμένος μηχανισμός αναζήτησης.
- Στο **Bittorrent**, λόγω παρουσίας του tracker, η διαδικασία της αναζήτησης 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 για τη ποιότητα των υπηρεσιών, δεν υπάρχει εγγύηση ότι τα δεδομένα θα εντοπιστούν**[23].

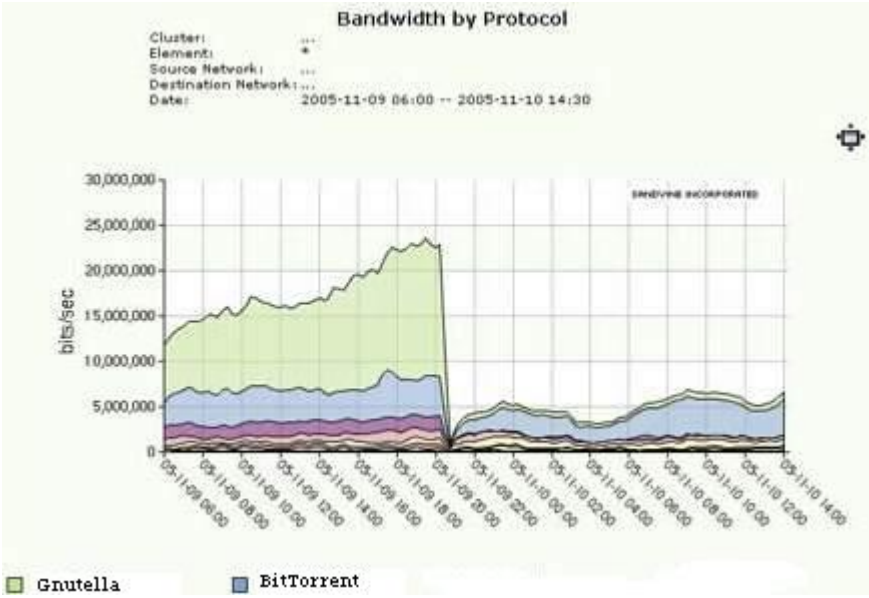
- Κάθε αναζήτηση στο Gnutella έχει σαν αποτέλεσμα τη δημιουργία μεγάλου όγκου μυνημάτων. Γενικά **στο δίκτυο Gnutella υπάρχει μεγαλύτερη καθυστέρηση στις αναζητήσεις**. Ωστόσο λειτουργεί καλύτερα σε μικρής κλίμακας δίκτυα.
- Στο **Gnutella**, για τη συντήρηση του δικτύου, υπάρχουν τα ζωτικής σημασίας μυνήματα PING-PONG, τα οποία δημιουργούν προβλήματα στη κλιμάκωση του δικτύου. Αυτό οδηγεί σε **μη αποδοτικές αναζητήσεις** καθώς, για "χάρη της κλιμάκωσης" και για να μην επιβαρυνθεί επιπλέον το σύστημα από μυνήματα, **δεν γίνεται αναζήτηση και έλεγχος σε όλους τους κόμβους**, καθώς υπάρχει και ο περιορισμός από τη τιμή του TTL (κάποιες φορές μπορεί και να χρειαστεί η εκ νέου υποβολή ενός ερωτήματος). Τέτοιο ζήτημα δεν υφίσταται στο Bittorrent.

4.4.3. Μηχανισμός αποθήκευσης-λήψης δεδομένων.

Κατά τη σύγκριση των δύο διομότιμων συστημάτων πάνω στη διαδικασία αποθήκευσης των δεδομένων, καταλήγουμε στα εξής συμπεράσματα:

- Οι ανταλλαγές δεδομένων γίνονται απευθείας μεταξύ των κόμβων, και στα δύο πρωτόκολλα.
- 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 (download) ενός αρχείου, θα πρέπει

υποχρεωτικά και να συνεισφέρει (upload) παράλληλα. Αυτό έχει σαν αποτέλεσμα την **καλύτερη εκμετάλευση του bandwidth των ομοτίμων**[24], το οποίο οδηγεί τις περισσότερες φορές σε **αύξηση του download rate**[24]. Κάτι τέτοιο δεν υπάρχει στο Gnutella άφου λόγω της διαδικασίας της πλυμμήρας των queries (αλλά και των υπολοίπων μυνημάτων), γίνεται σπατάλη του μεγαλύτερου ποσοστού του εύρους ζώνης (κυρίως σε παλαιότερου τύπου συνδέσεις). Στο παρακάτω σχήμα φαίνεται ότι **το Gnutella απαιτεί πολύ μεγαλύτερο bandwidth από το Bittorrent** (www.sandvine.com / *Νοέμβριος '05*).



- Εξάλλου 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.
- Ωστόσο η σημαντικότερη διαφορά και που ουσιαστικά, μαζί με την ύπαρξη του tracker, αποτελεί την ειδοποιό διαφορά ανάμεσα στα δύο πρωτόκολλα, είναι ότι **στο Bittorrent τα δεδομένα χωρίζονται σε κομμάτια** (chunk-based) και γίνεται ανταλλαγή αυτών, μεταξύ των ομοτίμων που βρίσκονται στο swarm. Αντίθετα **στο Gnutella, γίνεται**

λήψη ολόκληρου του αρχείου (whole file) από τους άλλους ομοτίμους.

- Με βάση το προηγούμενο, συμπεραίνουμε ότι το **Bittorrent ενδείκνυται 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.
- Στο Bittorrent, σε περίπτωση όπου δεν υπάρχει seeder σε ένα συγκεκριμένο swarm, "παγώνει" η διαδικασία λήψης ολόκληρου του αρχείου. Σε 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.

4.5. Σύγκριση σε γενικά ζητήματα των διομότιμων συστημάτων.

4.5.1. Ασφάλεια.

Η επίτευξη της ασφάλειας, γενικά στα P2P συστήματα, δεν είναι εύκολο έργο. 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: διαθεσιμότητα (**availability**) των δεδομένων και των ομοτίμων, έλεγχος της ακεραιότητας (**integrity**) των δεδομένων που παρέχονται από τους ομότιμους, ανωνυμία (**anonymity**) των ομοτίμων, έλεγχος της αυθεντικότητας (**authenticity**)

των ομοτίμων, αλλά και μηχανισμούς κατά των κακόβουλων λογισμικών (malware) όπως ιών, ad-wares κ.τ.λ.

Συγκεκριμένα για τα διομότιμα δίκτυα Bittorrent και Gnutella, πάνω στο θέμα αυτό, μπόρουμε να αναφέρουμε τα εξής:

Διαθεσιμότητα.

- **Στο Bittorrent όπως και στο δίκτυο Gnutella** [23]**η διαθεσιμότητα ενός περιεχομένου δεν μπορεί να προβλεφθεί και να εγγυηθεί**.
  Π.χ. Στο δίκτυο Bittorrent όταν και ο τελευταίος ομότιμος-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 επιρεάζει άμεσα την απόδοση του συστήματος. Στο δίκτυο 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**[32], που υπήρχαν στα δομημένα P2P δίκτυα

Ακεραιότητα.

- **Στο Gnutella δεν υπάρχει εγγύηση για την ακεραιότητα των δεδομένων**, καθώς η επαλίθευση 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.

Ανωνυμία.

- Τόσο **το Bittorrent όσο και το Gnutella δε προσφέρουν ανωνυμία στους χρήστες τους**, καθώς και στις δύο περιπτώσεις δεν αποκρύπτονται οι IP διευθύνσεις των ομοτίμων. Για το δίκτυο Bittorrent, οι ομότιμοι που συμμετέχουν σε ένα swarm 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.

Αυθεντικότητα.

- Τόσο το BitTorrent όσο και το Gnutella εμφανίζονται εκτεθημένα στο θέμα της αυθεντικότητας καθώς **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.

Μηχανισμοί κατά των malware.

- 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[24]. Επιπλέον ο αλγόριθμος της πλημμύρας που χρησιμοποιείται, βοηθάει στην περαιτέρω όξυνση και εξάπλωση του προβλήματος.

Τέλος να αναφέρουμε ότι γενικά **το πρωτόκολλο Gnutella, σε αντίθεση με το Bittorrent, δεν έχει δώσει ιδιαίτερη προσοχή στο θέμα της ασφάλειας**[25][26] και αυτό για χάρη της απλότητάς του. Με βάση μία μελέτη που είχε γίνει[25], υποστηρίζεται 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.

4.5.2. Κλιμάκωση.

Τα αποκεντρωμένα συστήματα όπως το Gnutella, παράγουν μεγάλο όγκο μυνημάτων κατά τις αναζητήσεις (αυξάνεται εκθετικά σε σχέση με τους συνδεδεμένους ομότιμους) 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 ένα πρόβλημα.

Αντίθετα **το Bittorrent** δε περιέχει κάποιο ενσωματωμένο μηχανισμό αναζήτησης και **θεωρείται αποδοτικότερο στο θέμα της κλιμάκωσης**, όπου αισθητή είναι η διαφορά κυρίως στα μεγάλα μεγέθους 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 το ποιά πρωτόκολλα (TCP/HTTP/HTTPS/UDP) χρησιμοποιεί [22].

Το θέμα της κλιμάκωσης αλλά και της ανθεκτικότητας στο Bittorrent, συνήθως επιλύεται με τους **multiple trackers**[32]**.** 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[31].

4.5.3. Free-riding.

Το θέμα του free-riding (ή anti-leeching) είναι ένα πολύ σημαντικό ζήτημα στα P2P δίκτυα. Αν σε ένα διομότιμο σύστημα παρατηρηθεί ότι οι ομότιμοι τείνουν να υιοθετούν μια "εγωιστική" συμπεριφορά, αυτό θα επιρεάσει κατά ένα πολύ μεγάλο ποσοστό την συνολική απόδοση του συστήματος. Ωστόσο η εξάλειψη του free-riding δεν είναι καθόλου εύκολη υπόθεση και αυτό οφείλεται στην αλτρουιστική φύση των διομότιμων δικτύων. Δεν είναι υποχρεωτική η παραμονή των ομοτίμων στο δίκτυο.

Θα λέγαμε ότι **το Bittorrent υπερτερεί κατά πολύ έναντι του Gnutella στο θέμα free-riding**, καθώς το Bittorrent έχει ενσωματώσει στο πρωτόκολλο του, μηχανισμούς (αναφορά στο 2^{ο} κεφάλαιο) οι οποίοι 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 το Bittorrent δεν έχει καταφέρει ακόμα να δώσει κίνητρα στους ομότιμους έτσι ώστε να μένουν στο δίκτυο και μετά την ολοκλήρωσει της λήψης του αρχείου.

4.5.3. Ανθεκτικότητα.

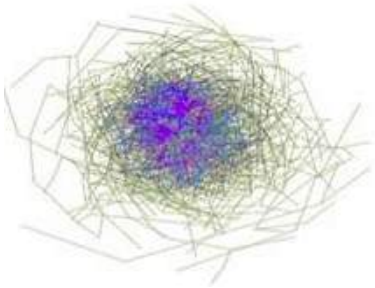
Στο Gnutella δεν υπάρχει κάποια μορφή κεντρικής οντότητας και άρα δεν υπάρχει single point of failure όπως στο Bittorrent λόγω του tracker. Ο tracker server αποτελεί ένα single point of failure για το δίκτυο[22]. Αν τεθεί εκτός λειτουργίας, θα είναι αδύνατο για τους νέους ομότιμος να ενταχθούν στο δίκτυο ή για τους ήδη υπάρχοντες να επικοινωνήσουν μεταξύ τους. Ωστόσο μια καλή λύση αποτελούν οι **multiple trackers**[32].

Το Gnutella είναι εκ φύσεως ανθεκτικότερο στην ανοχή σφαλμάτων (fault-tolerant), καθώς η απώλεια ενός ή ακόμα και πολλών ομοτίμων μπορεί εύκολα να αντισταθμιστεί, αλλά και στο Gnutella v0.6 η απώλεια ενός ή πολλών ultrapeer

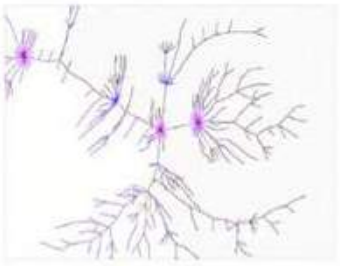
μπορεί εύκολα να αντισταθμιστεί, με τους "καλύτερους" υπάρχοντες κόμβους να γίνονται αυτόματα supernodes.

Παρ' όλα αυτά, **το Gnutella εμφανίζεται πιο ευάλωτο σε επιθέσεις** όπως π.χ. στην επιλογή και απομάκρυνση μερικών κόμβων που παρέχουν τη μεγαλύτερη συνδετικότητα (connectivity) στο δίκτυο.

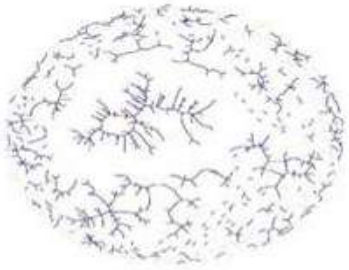
Στο παρακάτω σχήμα, μετά από μελέτη που έγινε[29], παρατηρήθηκε ότι αν από το αρχικό δίκτυο (Σχήμα 1), απομακρυνθεί με τυχαίο τρόπο το 30% των ομοτίμων (Σχήμα 2), το δίκτυο θα συνεχίσει να λειτουργεί χωρίς ιδιαίτερο πρόβλημα, ενώ αν απομακρυνθεί μόλις το 4% των "καλύτερων" ομότιμων (Σχήμα 3), το δίκτυο θα εμφανίζεται διχοτομημένο.



Σχημ.1 Partial Topology



Σχημ.2 Random 30% die



Σχημ.3 Targeted 4% die

Στο Bittorrent σε έρευνα που έγινε για την ανθεκτικότητά του σε επιθέσεις, παρατηρήθηκαν τα εξής[30]:

- Attack 1: Μετά από request για τη λίστα των ομοτίμων, εντοπισμός των seeders, και λήψη μόνο από αυτούς. Καθόλου upload.
  Αποτέλεσμα:
  - Παραβίαση του fairness model, βλαβερό για τους "δίκαιους" ομότιμους.
  - Private tracker: Ελαφρά καλύτερο rate κατά 22%. "Δίκαιοι" ομότιμοι, αντίκτυπο <15%.

- Public tracker: *Καλύτερο rate κατά 155%,* αντίκτυπο στους "δίκαιους" ομότιμους μέχρι και 32%. Το σύστημα μπορεί να εξαπατηθεί όσον αφορά τους public trackers.

- Attack 2: Μέσω του optimistic unchoking, εντοπισμός ομοτίμων με καλύτερο download rate και λήψη μόνο απόαυτούς. Αγνόηση υπολοίπων.
  Αποτέλεσμα:
  - Public tracker: *Μείωση* από 1-30%.

- Attack 3: Ανάρτηση ψεύτικων κομματιών.
  Αποτέλεσμα: Λειτουργεί στους private trackers, μερικό κέρδος.
  Δεν λειτουργεί στους public trackers, εντοπίστηκαν από τις εφαρμογές.

Με μόνη την εξαίρεση στην πρώτη επίθεση για τους public trackers, θα λέγαμε γενικά ότι **το bittorrent εμφανίζεται πιο ανθεκτικό στις επιθέσεις** από το Gnutella. Συμπεριλαμβανομένου και του μικρού αριθμού των free-riders, λόγω των μηχανισμών που υπάρχουν στο σύστημα, παρέχει ικανοποιητικές υπηρεσίες στους "συμμορφωμένους" ομοτίμους, κάτι το οποίο αφ'ενός δεν έχει προβλεφθεί από το gnutella (για το freeriding), και αφ'ετέρου γενικότερα οι υπηρεσίες που παρέχει δεν είναι τόσο αξιόπιστες.

ΠΑΡΑΡΤΗΜΑ Ά

Πειραματική μελέτη.

Σε αυτό το σημείο θα ακολουθήσει μια επίδειξη πάνω στη χρήση του πρωτοκόλλου Bittorrent. Συγκεκριμένα θα δούμε στη πράξη, το τρόπο εφαρμογής και λειτουργίας του πρωτοκόλλου και θα αναφερθούν με λεπτομέρεια τα βήματα και οι ρυθμίσεις που απαιτούνται από πλευράς χρήστη, έτσι ώστε να μπορεί να χρησιμοποιήσει το πρωτόκολλο Bittorrent για τη λήψη αρχείων.

Όπως είχε αναφερθεί στο 2^{ο} κεφάλαιο, για να μπορεί κάποιος να συμμετέχει στο διαμοιρασμό αρχείων μέσω του Bittorrent, θα πρέπει να εγκαταστήσει στον υπολογιστή του ένα πρόγραμμα, γνωστό ως Bittorrent client, και το οποίο εγκαθίσταται στο εκάστοτε λειτουργικό σύστημα του χρήστη. Αυτή τη στιγμή υπάρχουν γύρω στους 50 Bittorrent clients[33], όπου ο καθένας εξ αυτών υιοθετεί κάποια επιπλέον χαρακτηριστικά, με σκοπό την βελτίωση της ποιότητας των υπηρεσιών προς τους χρήστες[33].

Μερικά από τα σημαντικότερα extra χαρακτηριστικά είναι τα:

- Super seeding
- Tracker
- UPnP Port Mapping
- NAT Port Mapping Protocol
- NAT Traversal
- DHT
- Peer Exchange
- Encryption
- UDP Tracker
- Cache
- Web Seeding
- Broadcatching (RSS)
- Priotarization

- Selective downloading
- SOCKS
- Remote control via Web
- Search Engine
- Auto updates

Για τη παρουσίαση του πειραματικού μέρους της διπλωματικής εργασίας, θα επιλέξουμε το πρόγραμμα **uTorrent**, καθώς πρόκειται για έναν από τους δημοφιλέστερους[35] Bittorrent clients (βλ. παρακάτω σχήμα), ο οποίος υποστηρίζει όλα τα χαρακτηριστικά[33][34] που αναφέρθηκαν παραπάνω, και απαιτεί λιγότερους πόρους από το σύστημα σε σχέση με τα υπόλοιπα προγράμματα (λιγότερο από 6MB μνήμης)[36]. Πρόκειται για έναν client που αναβαθμίζεται συνεχώς (τελευταία έκδοση 28-09-2009) και είναι freeware. Ίσως το μόνο μειονέκτημά του είναι ότι έχει δημιουργηθεί μόνο για πλατφόρμα Windows, αλλά ωστόσο ύπαρχει τρόπος εγκατάστασης του και σε Linux (Ubuntu)[37].

	Sep-06	Jan-07	May-07	Sep-07	Network
Limewire	34.10%	36.80%	35.50%	36.40%	Gnutella
µTorrent	3.00%	4.60%	7.30%	11.30%	BitTorrent
BitTorrent	5.20%	5.30%	4.90%	4.60%	BitTorrent
Ares	3.50%	3.80%	4.30%	4.60%	Ares
Azureus	6.00%	6.30%	5.90%	4.30%	BitTorrent
eMule	3.80%	3.80%	4.30%	4.00%	eDonkey
BitComet	4.90%	4.40%	4.30%	3.90%	BitTorrent
BearShare	3.40%	3.40%	3.20%	2.90%	Gnutella
BitLord	2.40%	2.70%	2.80%	2.60%	BitTorrent
KaZaa	4.50%	3.10%	2.10%	1.50%	FastTrack
BitTornado	2.10%	1.80%	1.60%	1.50%	BitTorrent
Frostwire	0.00%	0.40%	1.00%	1.20%	Gnutella
Pando	0.00%	0.00%	0.80%	0.90%	Pando

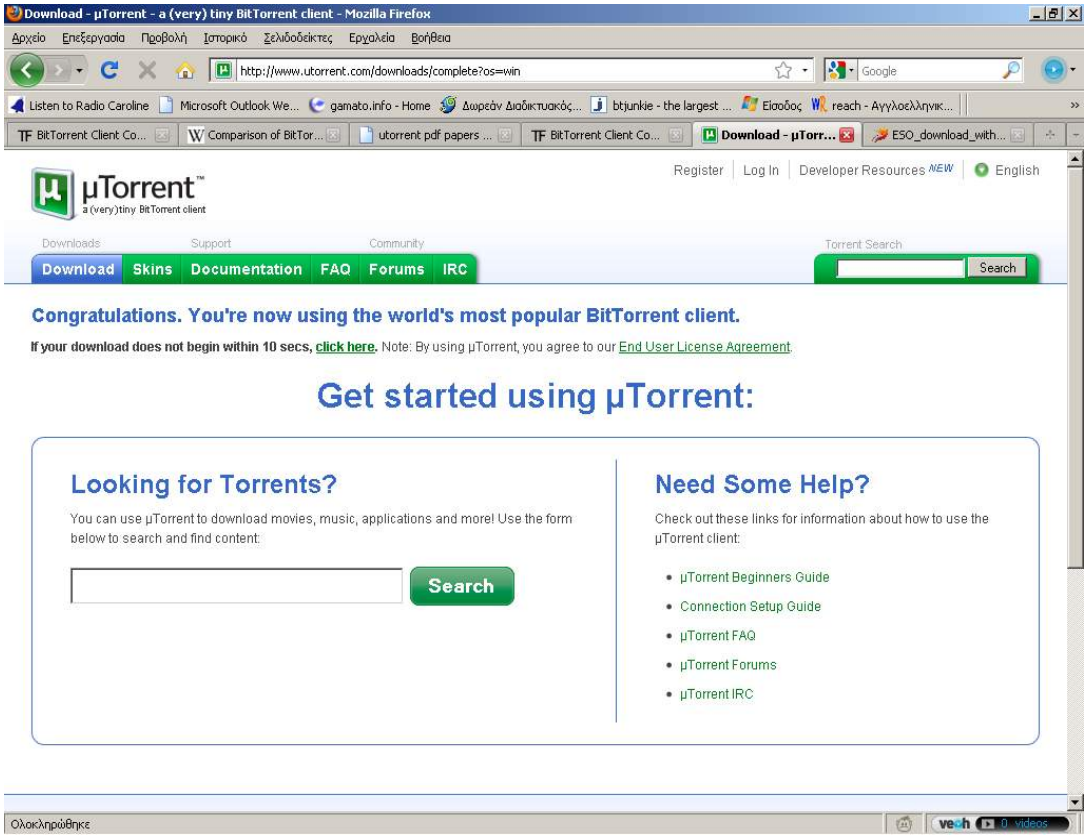
Digital Music News Research Group - P2P Market Share to September 2007 (1,661,688 PCs polled)

Σχ.1 Το uTorrent (ή µTorrent) αποτελεί τον δημοφιλέστερο Bittorrent client.

ΒΗΜΑΤΑ ΓΙΑ ΤΗ ΧΡΗΣΗ ΤΟΥ UTORRENT

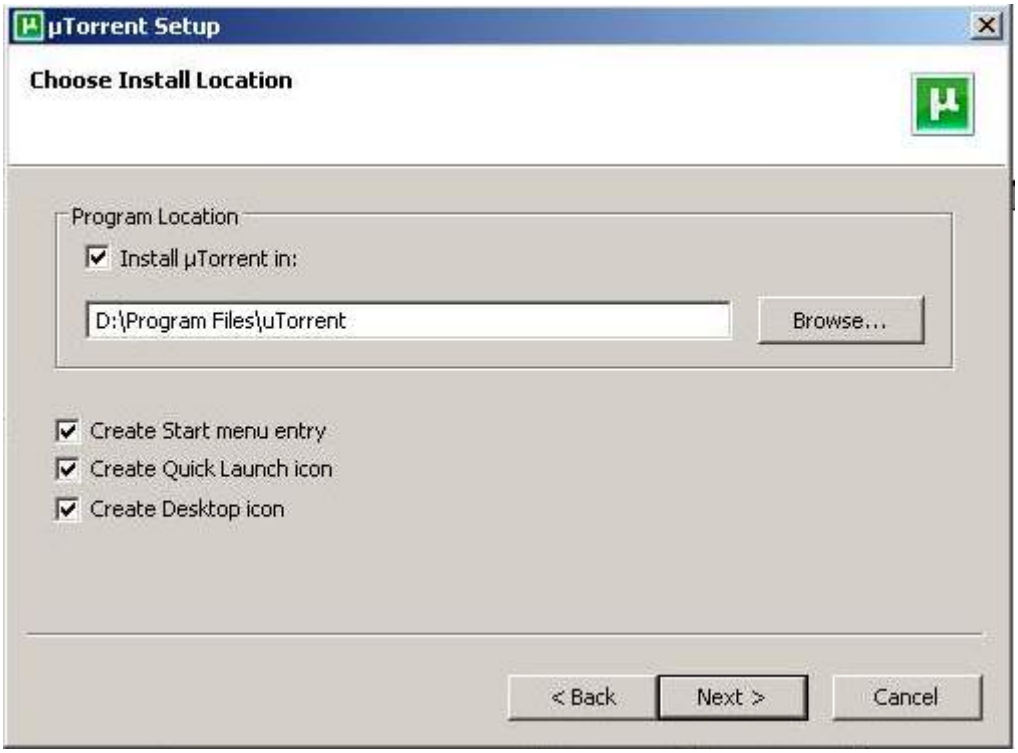
Βήμα 1^{ο}. Λήψη και εγκατάσταση του uTorrent.

Μέσω ενός web browser πηγαίνουμε στη διεύθυνση www.utorrent.com (επιλογή download)

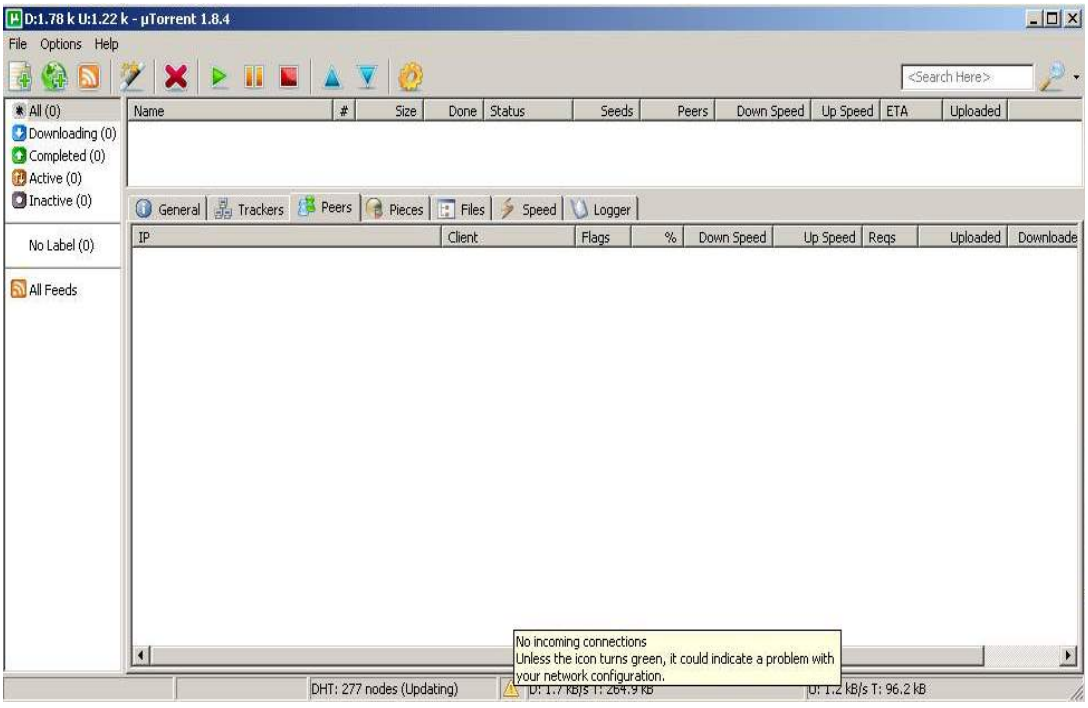


και κάνουμε λήψη του setup.

και εγκατάσταση του αρχείου

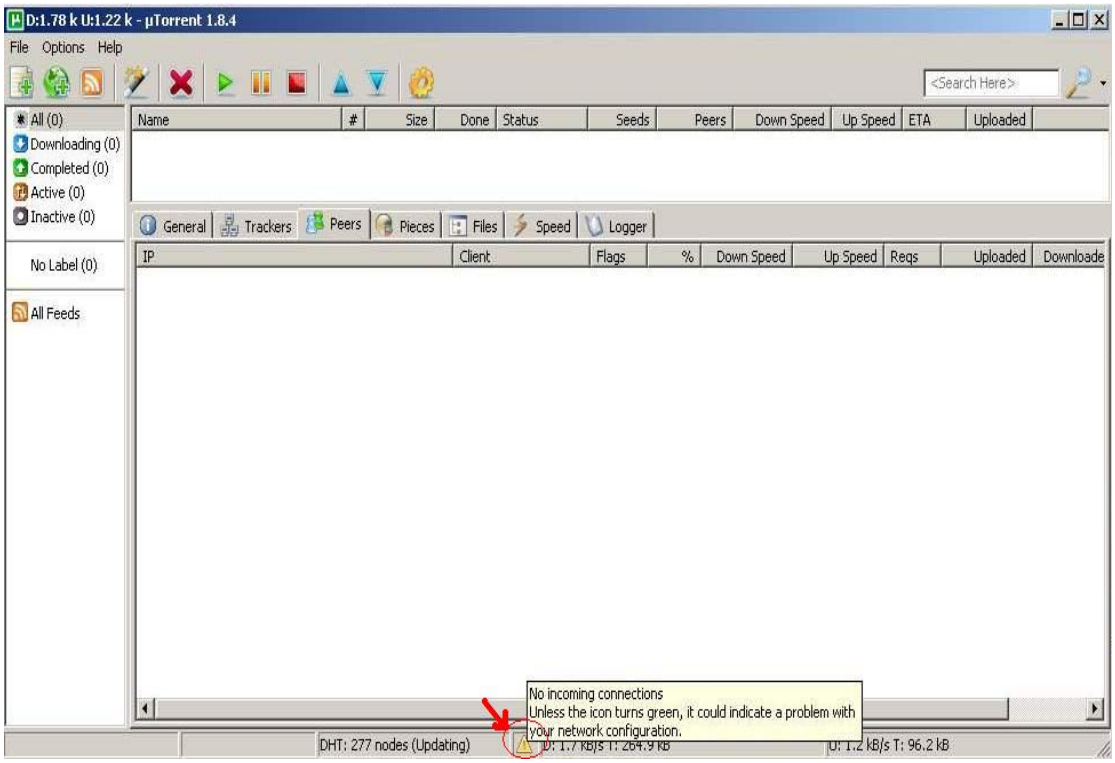


Μετά την εγκατάσταση του προγράμματος εμφανίζεται στον χρήστη το παρακάτω interface:



Βήμα 2ο. Ρυθμίσεις για τη σωστή λειτουργία του προγράμματος.

Η εγκατάσταση του προγράμματος δεν είναι αρκετή για να μπορεί το πρόγραμμα να λειτουργήσει σωστά.



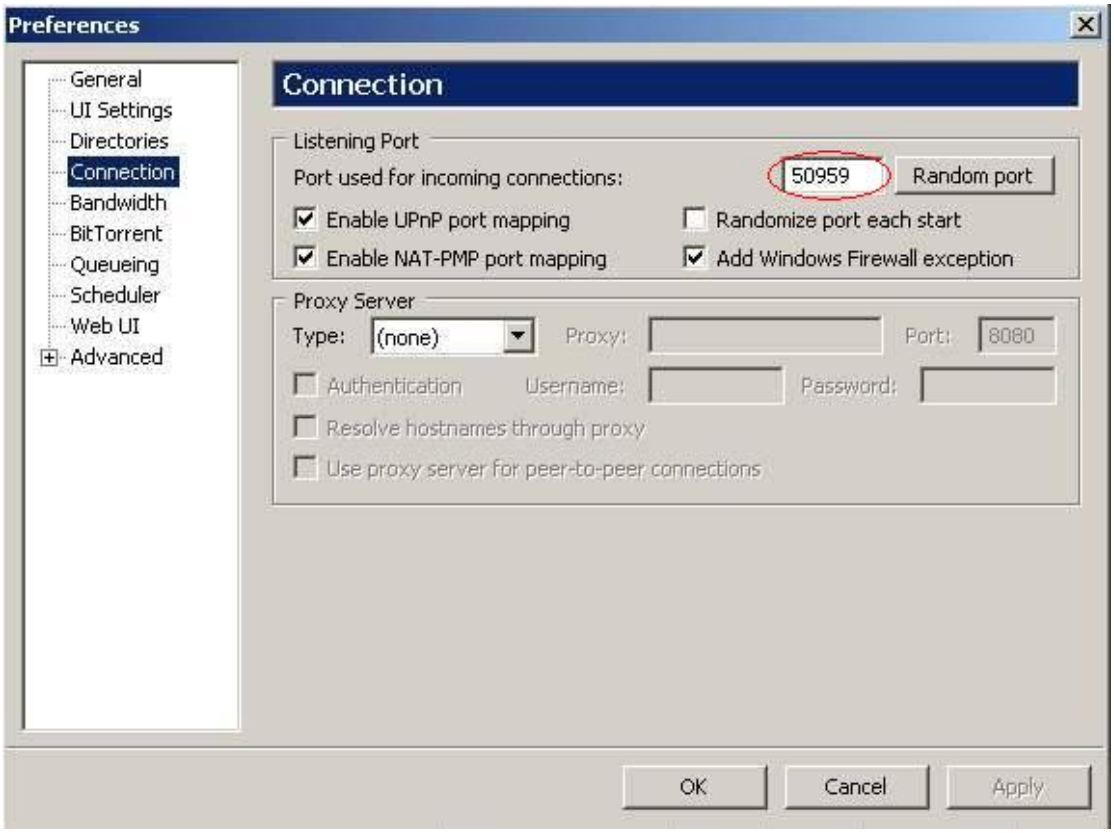
Σχ2. Το πρόγραμμα δε μπορεί να δεχτεί εισερχόμενες συνδέσεις.

Για να μπορεί να γίνει χρήση του προγράμματος ώστε να είναι δυνατή η ανταλλαγή αρχείων μεταξύ των ομοτίμων, απαιτείται μια διαδικασία γνωστή ως **port forwarding**.

Με αυτή τη τεχνική επιτρέπεται σε ένα απομακρυσμένο χρήστη να "εντοπίζει" ένα port σε μία ιδιωτική IP address μέσω ενός router[38]. Καθιστώντας έτσι δυνατή την επικοινωνία των υπολογιστών καθώς και το άνοιγμα συνδέσεων TCP και UDP.

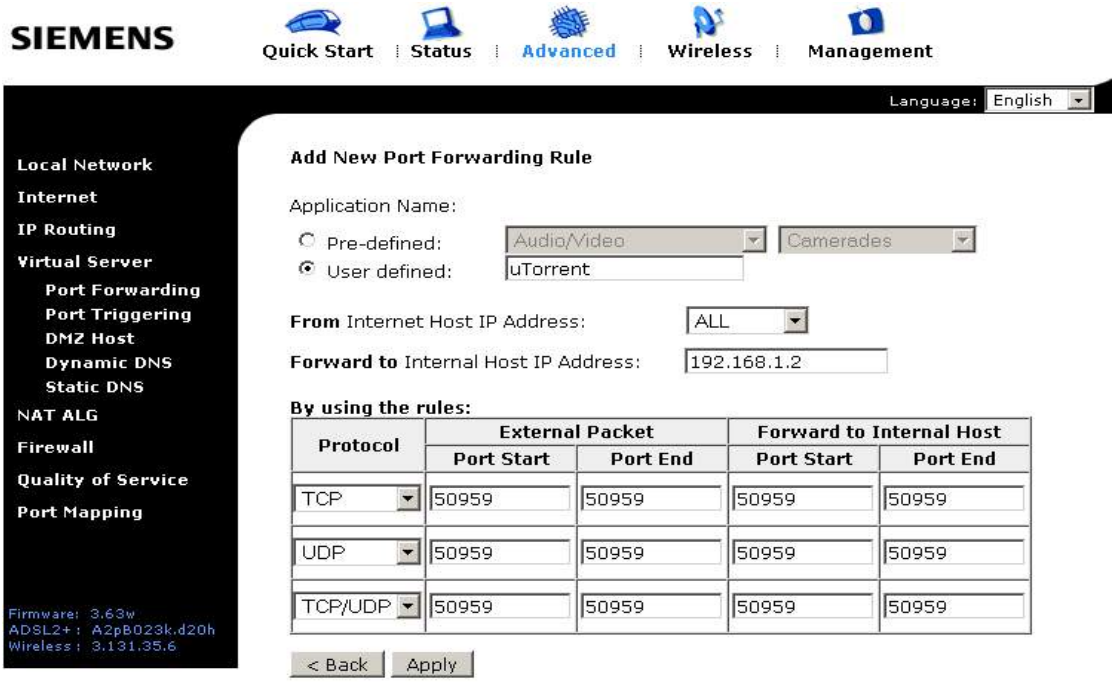
Port forwarding.

Μέσα από το μενού του προγράμματος (options->preferences->connection) επιλέγουμε το port, έστω 50959.

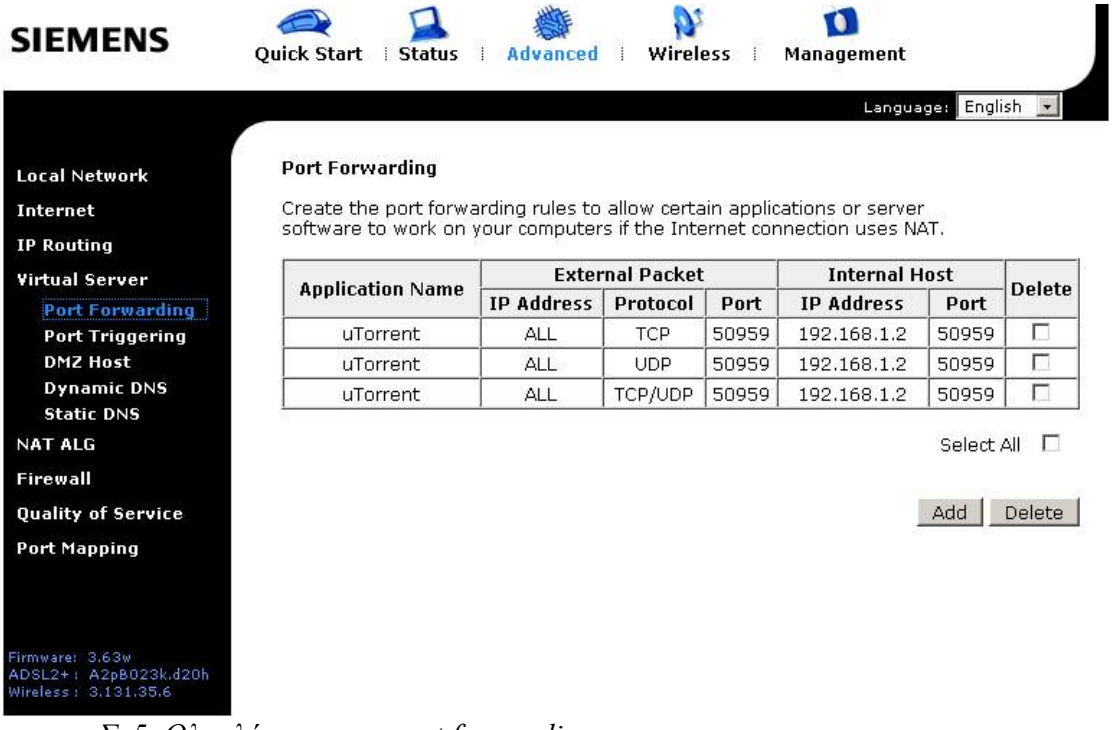


Σχ3. Ρύθμιση του επιθυμητού port για το uTorrent.

Απαιτείται επιπλέον και η ρύθμιση στο router:

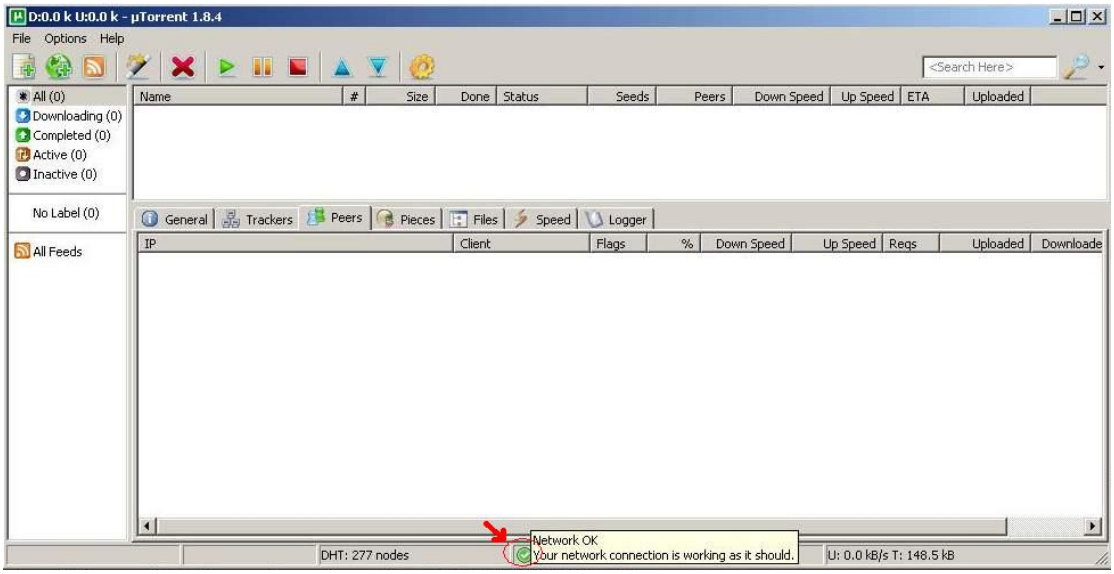


Σχ4. Ρύθμιση παραμέτρων στο router, για "άνοιγμα" συνδέσεων TCP και UDP στο αντίστοιχο port που έχει οριστεί στο uTorrent. (Απαιτείται και η εσωτερική IP του υπολογιστή (192.168.1.2))



Σχ5. Ολοκλήρωση του port forwarding.

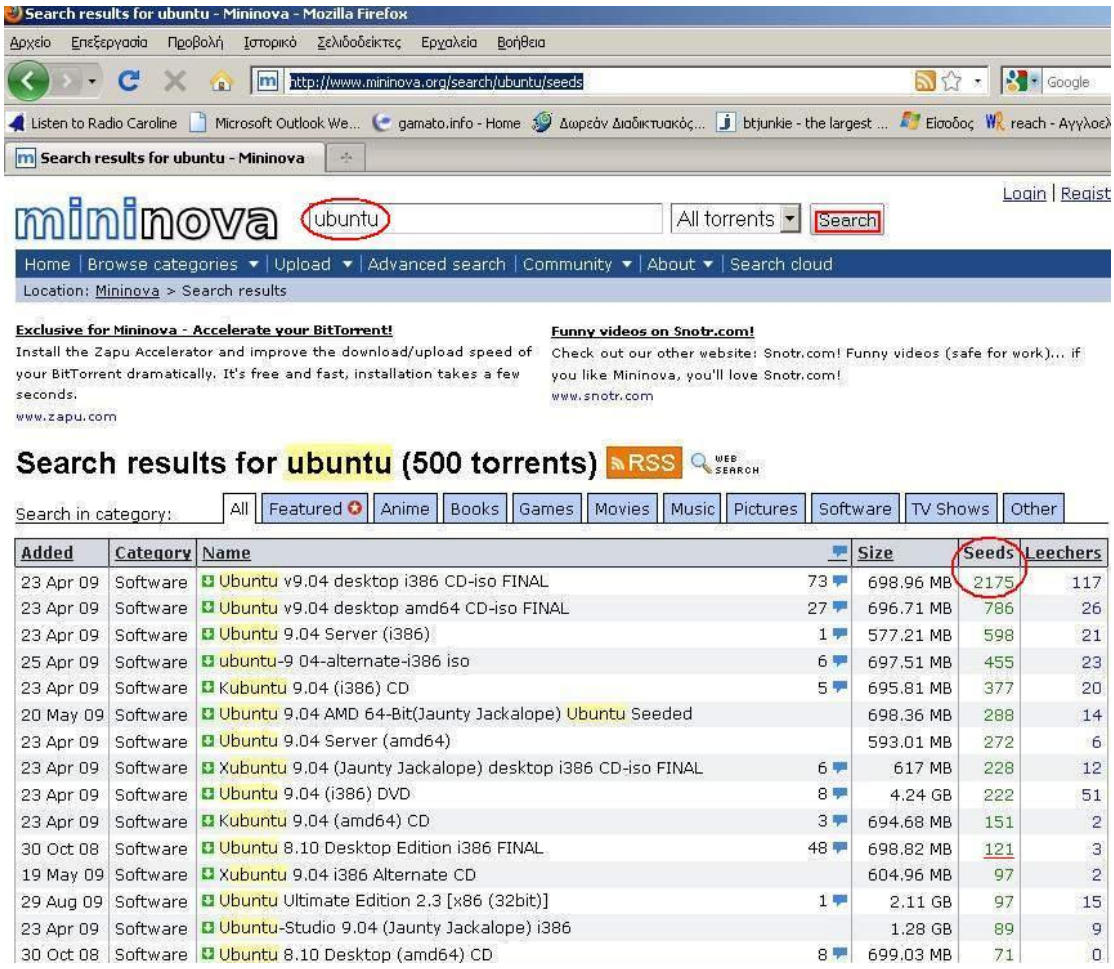
Μετά τη διαδικασία που αναφέρθηκε, εμφανίζεται στο χρήστη το παρακάτω interface:



Σχ6. Μετά την ολοκλήρωση του portforwarding, το πρόγραμμα είναι έτοιμο να δεχθεί εισερχόμενες συνδέσεις.

Βήμα 3ο. Λήψη του .torrent αρχείου.

Όπως έχει αναφερθεί στο 2ο κεφάλαιο, στο πρωτόκολλο Bittorrent δεν υπάρχει ενσωματωμένη η διαδικασία αναζήτησης. Αντ'αυτού, μέσω κατάλληλων ιστοσελίδων (torrent sites), ο χρήστης πληκτρολογεί το αρχείο που επιθυμεί, για τη λήψη του αντίστοιχου αρχείου .torrent. Εδώ έχει επιλεχθεί το site: http://www.mininova.org, και έστω ότι θέλουμε να κατεβάσουμε τη τελευταία έκδοση των ubuntu.

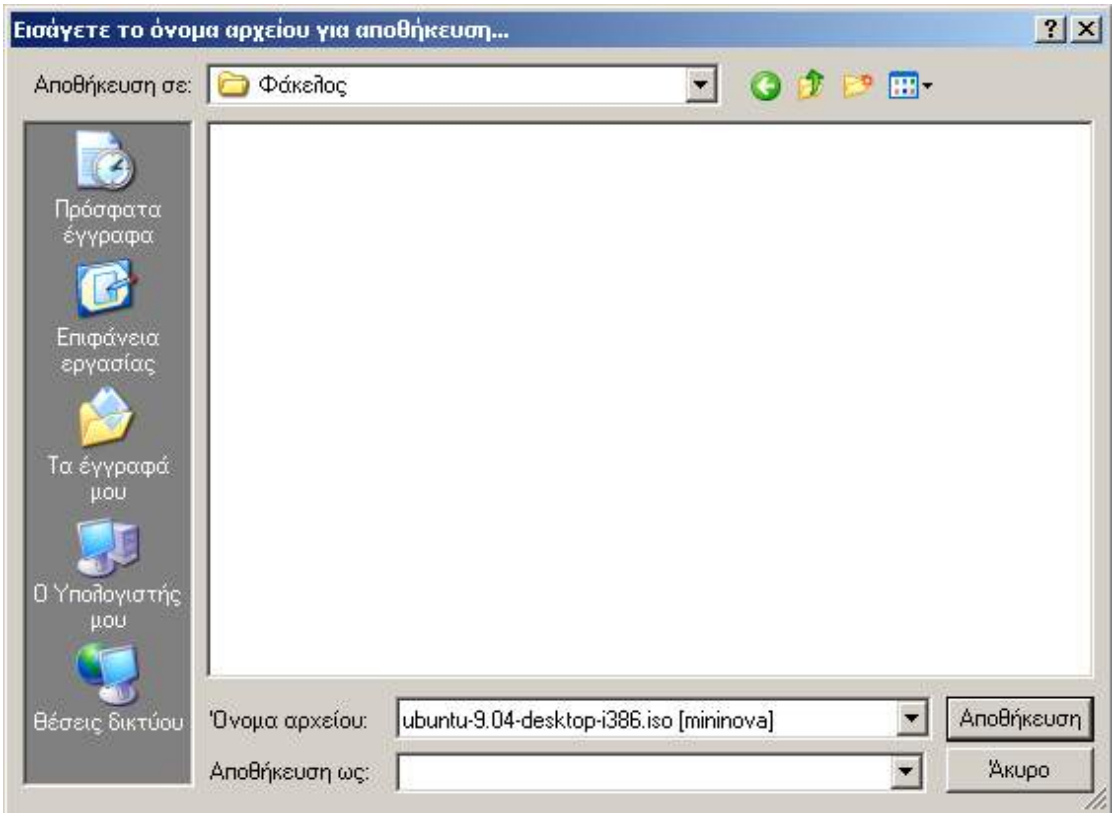


Added	Category	Name		Size	Seeds	Leechers
23 Apr 09	Software	Ubuntu v9.04 desktop i386 CD-iso FINAL	73	698.96 MB	2175	117
23 Apr 09	Software	Ubuntu v9.04 desktop amd64 CD-iso FINAL	27	696.71 MB	786	26
23 Apr 09	Software	Ubuntu 9.04 Server (i386)	1	577.21 MB	598	21
25 Apr 09	Software	ubuntu-9 04-alternate-i386 iso	6	697.51 MB	455	23
23 Apr 09	Software	Kubuntu 9.04 (i386) CD	5	695.81 MB	377	20
20 May 09	Software	Ubuntu 9.04 AMD 64-Bit(Jaunty Jackalope) Ubuntu Seeded		698.36 MB	288	14
23 Apr 09	Software	Ubuntu 9.04 Server (amd64)		593.01 MB	272	6
23 Apr 09	Software	Xubuntu 9.04 (Jaunty Jackalope) desktop i386 CD-iso FINAL	6	617 MB	228	12
23 Apr 09	Software	Ubuntu 9.04 (i386) DVD	8	4.24 GB	222	51
23 Apr 09	Software	Kubuntu 9.04 (amd64) CD	3	694.68 MB	151	2
30 Oct 08	Software	Ubuntu 8.10 Desktop Edition i386 FINAL	48	698.82 MB	121	3
19 May 09	Software	Xubuntu 9.04 i386 Alternate CD		604.96 MB	97	2
29 Aug 09	Software	Ubuntu Ultimate Edition 2.3 [x86 (32bit)]	1	2.11 GB	97	15
23 Apr 09	Software	Ubuntu-Studio 9.04 (Jaunty Jackalope) i386		1.28 GB	89	9
30 Oct 08	Software	Ubuntu 8.10 Desktop (amd64) CD	8	699.03 MB	71	0

Σχ7. Έυρεση αρχείου μέσω torrent site.

Όπως έχει αναφερθεί στο προηγούμενο κεφάλαιο, η διαθεσιμότητα στο Bittorrent, πολλές φορές εξαρτάται και από τη δημοτικότητα του αρχείου. Έτσι παρατηρούμε ότι για τη τελευταία έκδοση των ubuntu υπάρχουν 2175 seeders, ενώ για την έκδοση ubuntu 8.1 υπάρχουν μόνο 121.

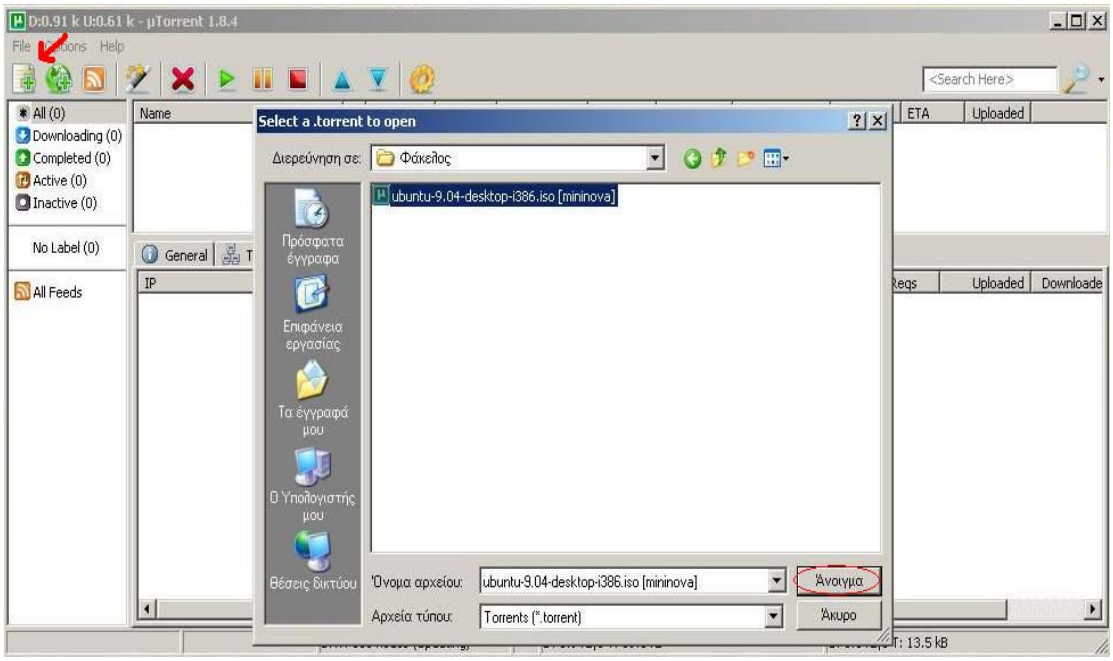
Το αρχείο αποθηκεύεται στον υπολογιστή του χρήστη.



Σχ8. Αποθήκευση του αρχείου .torrent.

Βήμα 4^{ο}. Άνοιγμα του .torrent αρχείου μέσω του Bittorrent client.

Στη συνέχεια ανοίγουμε το αρχείο .torrent, επιλέγοντας το κουμπί "add torrent" από τον client και κάνουμε browse το αποθηκευμένο αρχείο .torrent, ή απλά κάνοντας double click στο αρχείο.

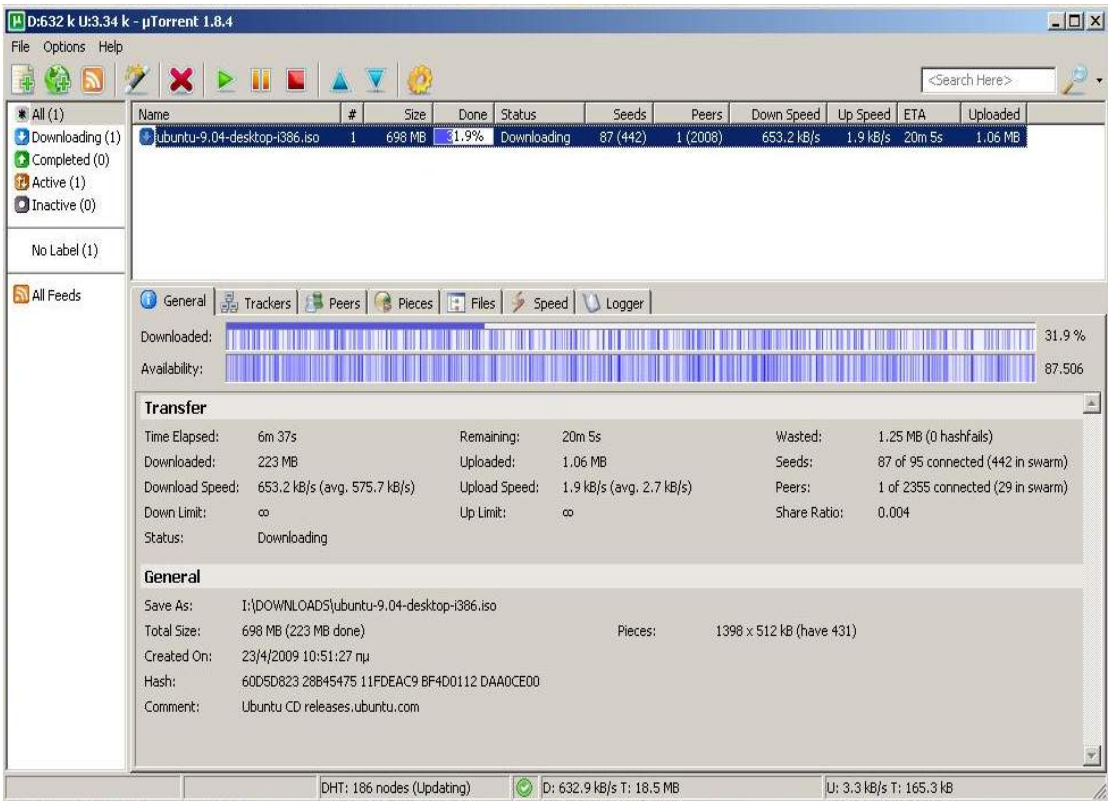


Σχ9. Άνοιγμα αρχείου .torrent.

Μετά από αυτό το βήμα, μπορεί να αρχίσει η λήψη του αρχείου.

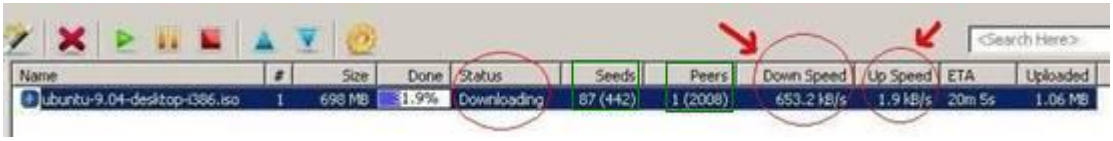
Βήμα 5ο. Λήψη αρχείου.

Στη συνέχεια στο χρήστη εμφανίζεται το παρακάτω interface, όπου στη πρώτη καρτέλα, παρέχονται γενικές πληροφορίες:



Σχ10. Interface του Bittorrent client, utorrent.

Όπως έχει αναφερθεί στο 2ο κεφάλαιο, το πρωτόκολλο bittorrent, υποχρεώνει τους χρήστες που κάνουν λήψη ενός αρχείου, να κάνουν ταυτόχρονα και upload, με αποτέλεσμα τη καλύτερη εκμετάλευση του bandwidth των ομοτίμων, οδηγώντας συχνά σε αύξηση του download rate. Όπως φαίνεται και παρακάτω:



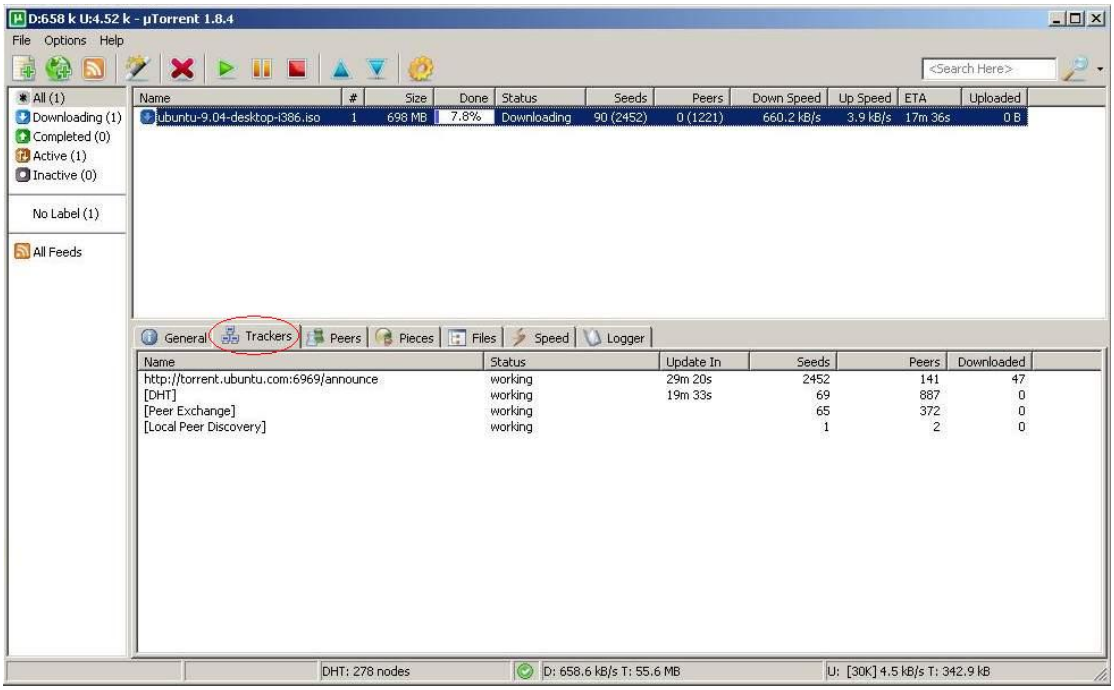
Σχ11. Κατάσταση downloading, παράλληλο download και upload.

Σε πράσινο πλαίσιο του παραπάνω screenshot, εμφανίζονται οι seeders (87) και οι peers (1), με τους οποίους ο χρήστης είναι συνδεμένος, καθώς και ο συνολικός αριθμός των seeders/peers που έχουν συνδεθεί στο συγκεκριμένο torrent, 442 και 2008 αντίστοιχα.

Στο παρακάτω snapshot, εμφανίζονται φαίνονται οι πληροφορίες που περιέχονται στο .torrent αρχείο.

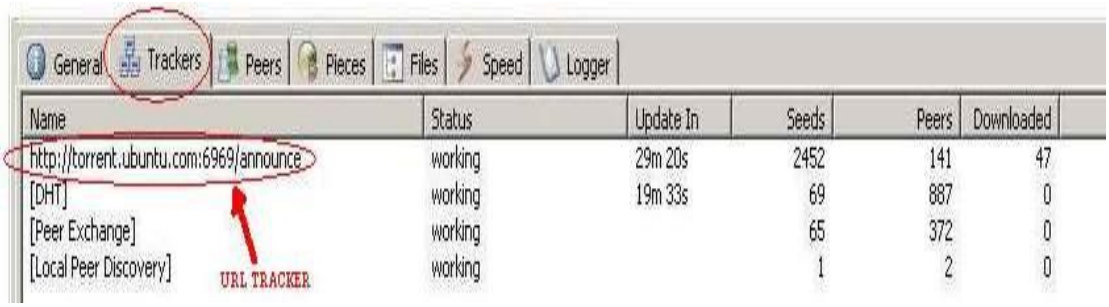
Σχ12. Πληροφορίες για το .torrent αρχείο.

Στη δεύτερη καρτέλα του προγράμματος, υπάρχουν πληροφορίες σχετικά με τον tracker (ή τους trackers).



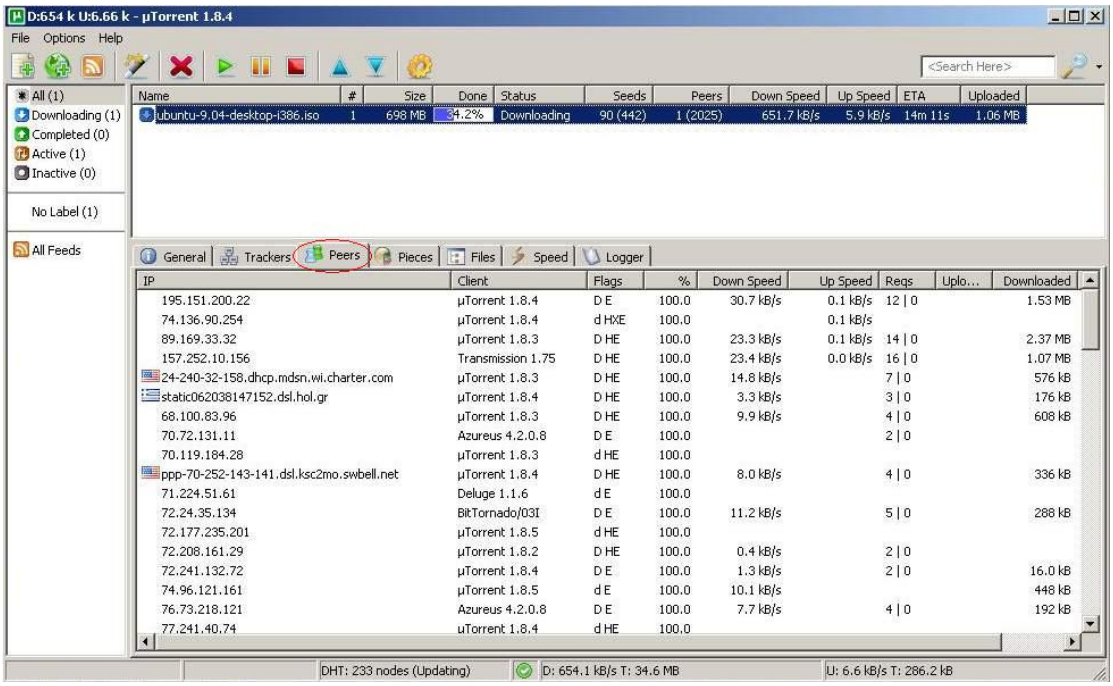
Σχ12. Καρτέλα trackers.

Παρακάτω εμφανίζεται το url του tracker, καθώς και ο αριθμός των seeds/peers που είναι εγγεγραμένοι σε αυτόν .



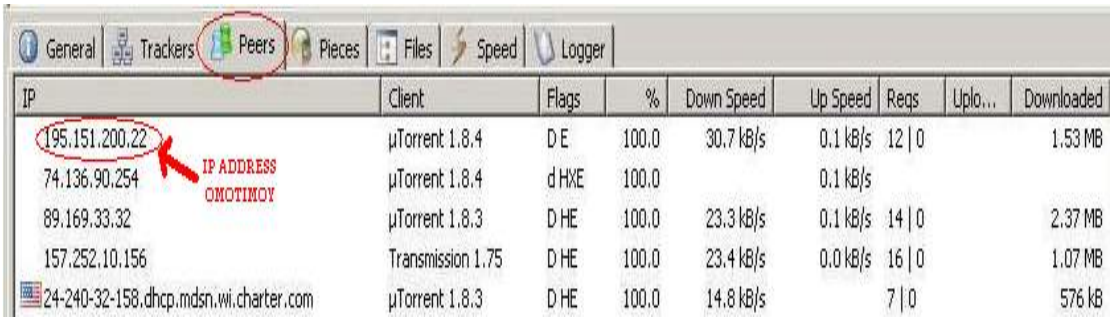
Σχ13. Πληροφορίες για τον tracker.

Στη καρτέλα, ”Peers”, εμφανίζονται πληροφορίες σχετικά με τους ομοτίμους οι οποίοι είναι συνδεδεμένοι στο swarm, εκείνη τη στιγμή.

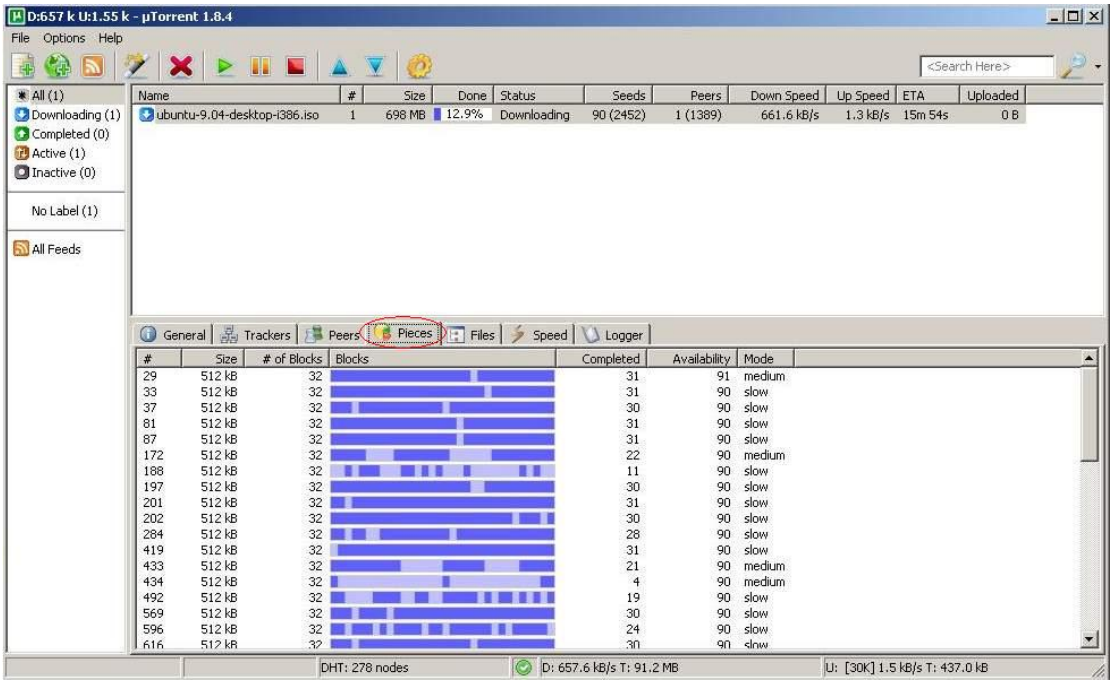


Σχ12. Καρτέλα ομοτίμων.

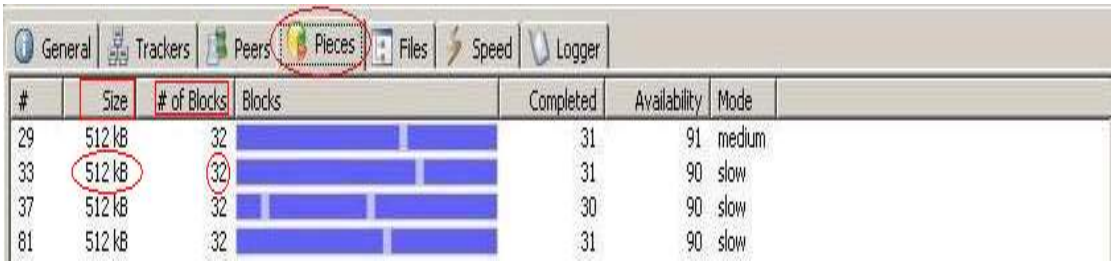
Όπως είχε αναφερθεί στο 4^{ο} κεφάλαιο, στο πρωτόκολλο bittorrent δεν υπάρχει ανωνυμία, όπως φαίνεται και στο παρακάτω snapshot, όπου είναι εμφανείς οι IP διευθύνσεις των ομοτίμων.



Στην καρτέλα “Pieces” του προγράμματος φαίνονται τα κομμάτια του αρχείου, αλλά και ο αριθμός των blocks από τα οποία αποτελείται το κάθε κομμάτι.

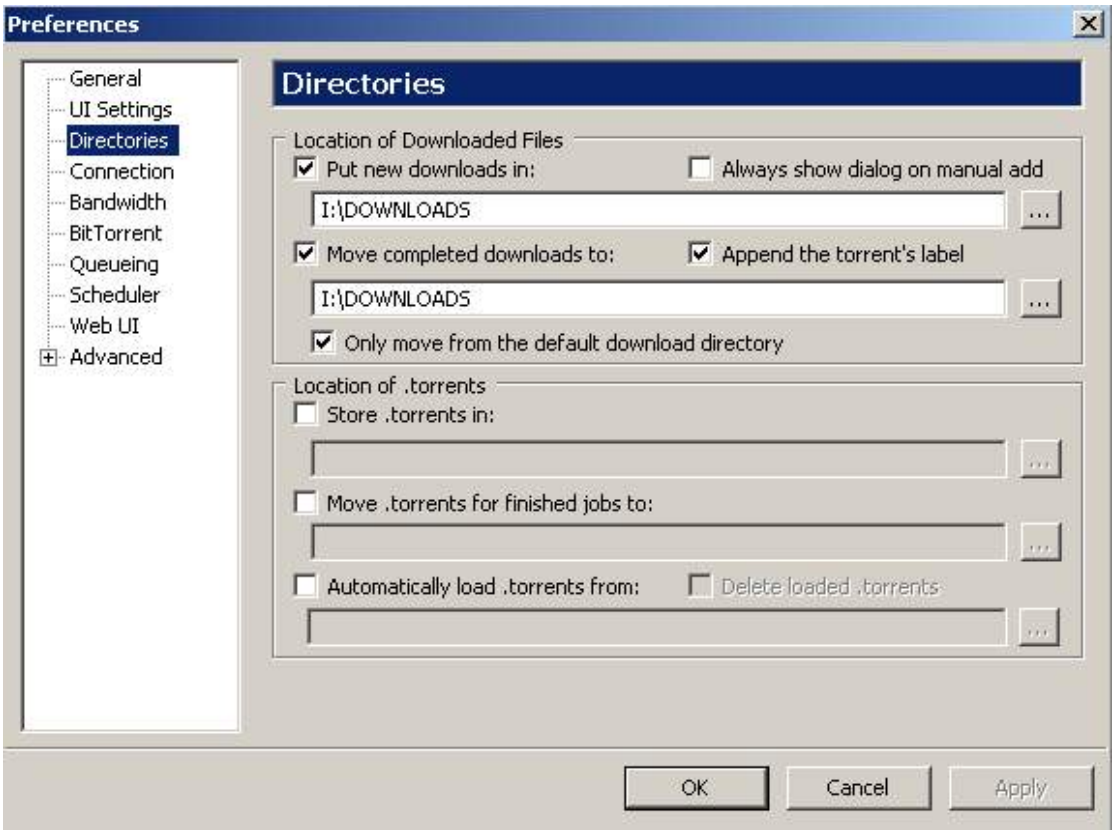


Στο συγκεκριμένο αρχείο, τα κομμάτια έχουν μέγεθος 512 MB, και το κάθε κομμάτι αποτελείται από 32 blocks, όπως φαίνεται και παρακάτω:

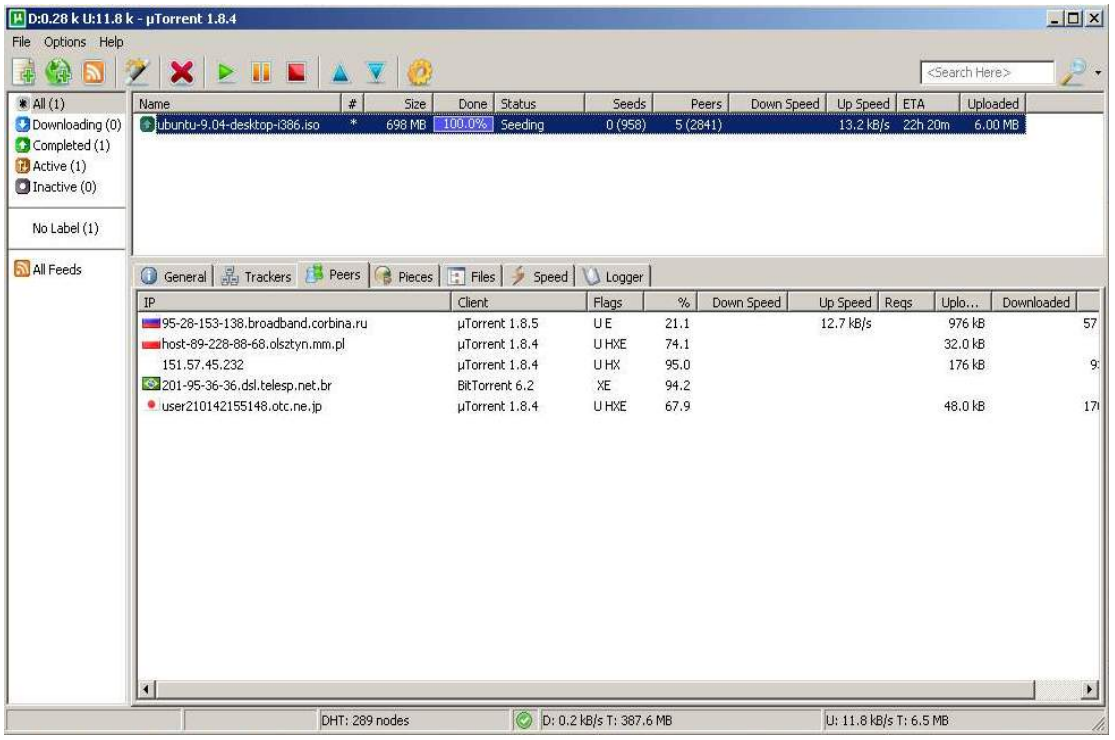


Βήμα 6ο. Ολοκλήρωση λήψης αρχείου.

Μετά την ολοκλήρωση λήψης του αρχείου, τα δεδομένα αποθηκεύονται σε ένα προκαθορισμένο χώρο στο σκληρό δίσκο του χρήστη.



Μετά την ολοκλήρωση λήψης του αρχείου, ο ομότιμος γίνεται seeder, και μπορεί να προσφέρει (upload) στους υπόλοιπους ομότιμους στο swarm, χωρίς ωστόσο να έχει πλέον ο ίδιος κάποιο κέρδος.



Σχ12. Κατάσταση seeding.

ΠΑΡΑΡΤΗΜΑ ΄Β

1. ΕΥΧΑΡΙΣΤΙΕΣ

Για την διεκπεραίωση της διπλωματικής μου εργασίας θα πρέπει να ευχαριστήσω την επιβλέπουσα καθηγήτρια κ.Π.Φραγκοπούλου για τη βοήθεια που μου προσέφερε κατά τη διάρκεια της εργασίας αυτής, καθώς και για την επιλογή του θέματος.

2.ΒΙΒΛΙΟΓΡΑΦΙΑ

ΠΗΓΕΣ ΑΠΟ ΔΙΑΔΙΚΤΥΟ

[1]http://www.cs.ucy.ac.cy/~dzeina/talks/p2p-ucy.16.11.2006.pdf "Peer-to-Peer Systems: Introduction and Challenges" Δημήτρης Ζειναλιπούρ Λέκτορας, Ανοιχτό Παν/μιο Κύπρου.

[2] http://www.cs.uoi.gr/~pitoura/courses/ds03_gr/assign/assgn5/GP.pdf
http://www.comp.brad.ac.uk/het-net/HET-NETs04/CameraPapers/T4.pdf
"Structured P2P Networks in Mobile and Fixed Environments".

Jorg Eberspacher, Rudiger Schollmeier, Stefan Zols, Gerald Kunzmann:

[3]http://ce.et.tudelft.nl/publicationfiles/1075_526_prorisc05.pdf
"A Survey of Peer-toPeer Networks". B.Pourebrahimi, K. Bertels, S Vassiliadis. Computer Engineering Laboratory, ITS,TU Delft, The Netherlands.

[4]http://www.run.montefiore.ulg.ac.be/~soldani/P2P_Behaviour_Detection.pdf, "Peer-to-Peer Behaviours Detection by TCP Flows Analysis.", University of Liege, Faculty of Applied Sciences, Cyril Soldani.

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

[6]www.slideshare.net/SridharBR/bit-torrent-protocol-report
Dept of CS&E,WCE

[7]http://www.bittorrent.org/bittorrentecon.pdf
"Incentives Build Robustness in BitTorrent", Bram Cohen, May 22, 2003

[8]http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf

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

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

[11]http://www.smith.umd.edu/seminars/pdfs/masanell.pdf

"Peer-to-Peer File Sharing and the Market for the Digital Information Goods" Harvard Business School, Ramon Casadesus - Masanell, Andres Hervas, Jordan Mitchell, (March 7, 2006).

[12]www.cs.rice.edu/Conferences/IPTPS02/128.pdf

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

[13]http://www.eecs.berkeley.edu/~daw/teaching/cs261-f02/reports/defig.pdf

"Analysis of Peer-to-Peer Network Security using Gnutella", Dimitri Defigueiredo, Antonio Garcia, Bill Kramer. Department of Computer Science, Univercity of California, Berkeley.

[14]http://www.facweb.iitkgp.ernet.in/~pabitra/facad/06CS6019t.pdf

"Topology Formation and Replication Strategies for Gnutella Network", Santosh Kumar Shaw, Department of Computer Science and Engineering, Master of Technology, Indian Institute of Technology, Kharagpur, May 2008.

[15]http://www.ftc.gov/be/seminardocs/050523peertopeersmith.htm.pdf

"Interest-Based Self-Organizing Peer-to-Peer Networks: A Club Economics Approach", Atip Asvanund, Ramayya Krishnan, Michael D. Smith, and Rhul Telang. School of Public Policy and Management, Carnegie Mellon University, Pittsburgh, April 2005.

[16]http://www.faculty.jacobs-university.de/jschoenwae/nds-2007/varsandan-report.pdf

"A Peer to Peer Network for Distributed Case Based Reasoning", Project Report, Ioana Varsandan, Jacobs University Bremen, Spring 2007.

[17]http://www.stephan-brumme.com/download/agents/Intelligent %20Agents.pdf

"Monitoring The Gnutella Network", Stephan Brumme, University of Technology, Sydney Australia.

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

[19]http://www.comp.lancs.ac.uk/~geoff/Publications/DSO05.pdf

"Free Riding on Gnutella Revisited: the Bell Tolls?"

Daniel Hughes, Geoff Coulson, James Walkerdine

Computing Department, Lancaster University, Lancaster, UK

[20]http://en.wikipedia.org/wiki/Gnutella_crawler

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

[22]http://www.land.ufrj.br/~classes/coppe-redes-2008/biblio/P2P-content-delivery.pdf

"On peer-to-peer (P2P) content delivery", Jin Li, 29 November 2007

[23]http://www.cl.cam.ac.uk/teaching/2005/AdvSysTop/survey.pdf

"A Survey and Comparison of Peer-to-Peer Overlay Network Schemes", Eng Keong Lua, Jon Crowcroft, Marcelo Pias, Ravi Sharma and Steven Lim. 2004.

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

[25]http://www.cs.ucr.edu/~csyiazti/courses/cs260-2/project/gnutella.pdf

"Exploiting the Security Weaknesses of the Gnutella Protocol",

Demetrios Zeinalipour-Yazti,

Department of Computer Science University of California.\

[26]http://cseweb.ucsd.edu/~rmotta/my_papers/Gnutella_Integrating_Performance_And_Security_In_Fully_Decentralized_P2P_Models.pdf

"Gnutella: Integrating Performance And Security In Fully Decentralized P2P Models", Rossana Motta, Wickus Nienaber, Jon Jenkins.

Florida State University, Computer Science.

[29]http://www.cs.washington.edu/homes/gribble/papers/mmcn.pdf

"A Measurement Study of Peer-to-Peer File Sharing Systems"

Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble

Dept. of Computer Science and Engineering, Univ. of Washington, Seattle

[30]www.cs.ucsb.edu/~ravenben/classes/176C/lectures/structuredp2p1.ppt

CS176C, Spring 2006 "Structured Peer-to-Peer Systems"

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

[32]http://www.cs.umass.edu/~arun/papers/BT_availability.pdf

"Availability in BitTorrent Systems", Giovanni Neglia, Giuseppe Reina,

Honggang Zhang,Don Towsley, Arun Venkataramani, John Danaher.

University of Palermo and Massachusetts.

[33]http://en.wikipedia.org/wiki/Comparison_of_BitTorrent_software

[34]http://www.computerpoweruser.com/articles/archive/c0608/40c08/40c08chart.pdf?guid=

[35]http://news.softpedia.com/newsPDF/uTorrent-Is-The-Most-Popular-BitTorrent-Client-83696.pdf
[36]http://www.utorrent.com/
[37]http://news.softpedia.com/newsPDF/uTorrent-under-Ubuntu-in-3-Easy-Steps-49037.pdf
[38]http://en.wikipedia.org/wiki/Port_forwarding