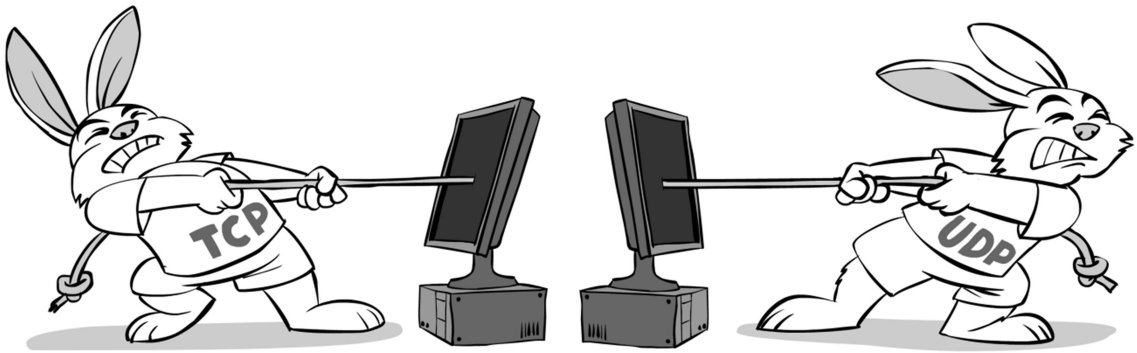
ΔΙΚΤΥΑ ΙΙ

ΑΝΑΦΟΡΑ

25/11/2020
ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ (ΗΜΜΥ)
ΧΡΥΣΟΥΛΑ ΤΣΙΜΠΕΡΗ ΑΕΜ: 9272
Tsimperi@ece.auth.gr



Εισαγωγή

Η παρούσα εργασία στοχεύει στην ανάπτυξη μίας εφαρμογής δικτυακού προγραμματισμού, με στόχο:

- την εξοικείωση με το πρωτόκολλο επικοινωνίας υπολογιστών UDP (user datagram protocol),
- την εξοικείωση με τους μηχανισμούς μετάδοσης ψηφιακού ήχου σε πραγματικό χρόνο (audio streaming)
- τη διεξαγωγή μετρήσεων ορισμένων παραμέτρων, και την παρατήρηση της επιρροής
- αυτών στην ποιότητα της επικοινωνίας.

Η εφαρμογή αναπτύσσεται με τη χρήση της γλώσσας προγραμματισμού Java, που χρησιμοποιείται κατά κόρον στις δικτυακές εφαρμογές λόγω

1. της ανεξαρτησίας της από τον τύπο πλατφόρμας του κάθε υπολογιστή και
2. της απλούστερης διαχείρισης των δικτυακών πόρων.

Η εφαρμογή αυτή συνδέεται μέσω του διαδικτύου με το ***server ithaki*** του εικονικού εργαστηρίου. Προκειμένου να είναι εφικτή αυτή η σύνδεση είναι απαραίτητη η διαθεσιμότητα μίας σύνδεσης στο internet, ενός modem, καθώς και η χρήση των θυρών (sockets) των δύο απομακρυσμένων υπολογιστών. Οι εφαρμογές ανταλλάσσουν δεδομένα με τη μορφή πακέτων, δηλαδή μικρών αυτόνομων τμημάτων πληροφορίας. Υπεύθυνες για την κατάτμηση της πληροφορίας σε πακέτα, και την μετέπειτα επανένωση των πακέτων είναι οι εφαρμογές που τρέχουν στους δύο απομακρυσμένους υπολογιστές. Η αποστολή και λήψη των πακέτων γίνεται μέσω μιας απλής εγγραφής ή ανάγνωσης από τα sockets του κάθε υπολογιστή, χωρίς να ενδιαφέρει, στο επίπεδο της εφαρμογής, η διαδικασία μεταφοράς των πακέτων. Μετά την εγγραφή στο socket και πριν την ανάγνωση αντίστοιχα, το modem αναλαμβάνει την κατάλληλη διαμόρφωση και αποδιαμόρφωσή των μεταφερόμενων δεδομένων. Με αυτό τον τρόπο είναι δυνατή η μετάδοση του κατάλληλου σήματος πληροφορίας μέσω του δικτύου. Η παραπάνω διαδικασία είναι σχετικά οικεία, λόγω της ενασχόλησης με το μάθημα Δίκτυα Υπολογιστών Ι, αλλά και μέσω της καθημερινής ενασχόλησης μας με το διαδίκτυο.Γεννιέται όμως η ανάγκη για εμβάθυνση στους μηχανισμούς των πακέτων που χρησιμοποιούνται, μιας και πρόκειται για ένα μηχανισμό επικοινωνίας που δε γίνεται άμεσα αντιληπτός - απτός από τον μέσο χρήστη. Ένας από τους υπάρχοντες “μηχανισμούς” πακέτων είναι και το πρωτόκολλο UDP, στη χρήση του οποίου βασίζεται η εφαρμογή της εργασίας. Η φύση του πρωτοκόλλου UDP, τα χαρακτηριστικά του, η δομή του, καθώς και η χρησιμότητά του, αναλύονται στην επόμενη παράγραφο.

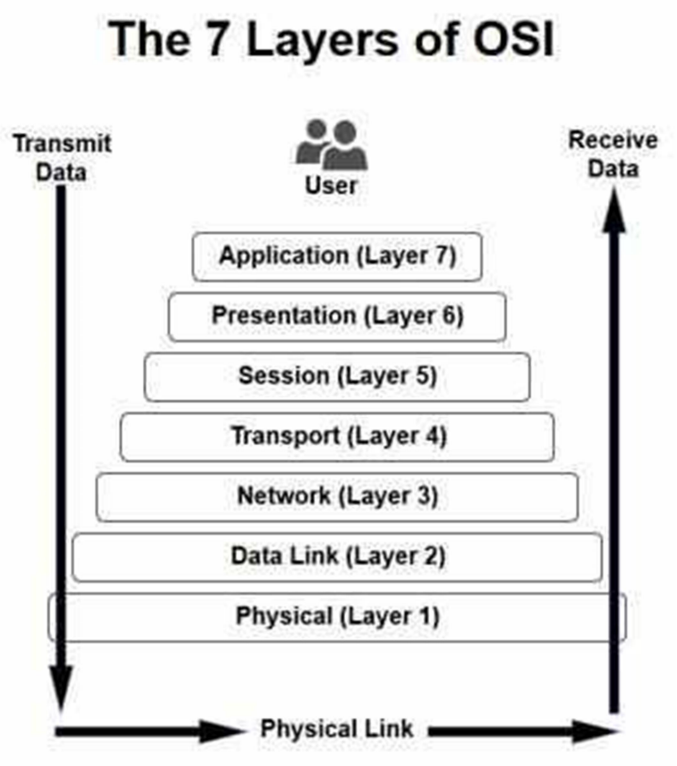
Αυτό το πρωτόκολλο είναι και το κύριο εργαλείο που χρησιμοποιείται στα πλαίσια της εργασίας ώστε να αναπτυχθεί ένας μηχανισμός audio streaming (όπως επίσης και ένας προαιρετικός μηχανισμός video streaming). Η ιδιαίτερη φύση των streaming εφαρμογών, τα χαρακτηριστικά και οι απαιτήσεις τους αναφέρονται κατά την ανάλυση της χρησιμότητας του UDP πρωτοκόλλου, μιας και αποτελούν σημαντικό τομέα εφαρμογής του. Η ανάπτυξη όμως των audio streaming εφαρμογών οδήγησαν και στη δημιουργία πολλών ακόμα σχετικών πρωτοκόλλων. Αναφορά σε μερικά

διεθνή πρωτόκολλα γίνεται σε ξεχωριστή παράγραφο, αμέσως μετά την τεχνική αναφορά στο πρωτόκολλο UDP.

Πρωτόκολλο UDP

Γενικά

Όπως είναι γνωστό, το internet, αλλά και τα περισσότερα είδη δικτύων, βασίζουν τη λειτουργία τους και την αποτελεσματικότητά τους σε τυποποιημένες διαδικασίες, που περιγράφονται λεπτομερώς από τη Σουίτα Πρωτοκόλλων Διαδικτύου (Internet protocol suite).



Τα πρωτόκολλα αυτής της σουίτας, σύμφωνα με το μοντέλο OSI κατατάσσονται σε 7 στρώματα (layers). Ένα από τα βασικότερα είναι το πρωτόκολλο UDP (User Datagram Protocol ή και Universal Datagram Protocol), το οποίο βρίσκεται στο επίπεδο (4) της Μεταφοράς (Transport) του μοντέλου OSI. Με τη χρήση αυτού είναι δυνατή η μεταφορά σύντομων μηνυμάτων (datagrams) ανάμεσα σε δύο υπολογιστές ενός δικτύου.

Προκειμένου να περιγράψουμε τη λειτουργία του UDP, θα χρησιμοποιήσουμε ως μέτρο σύγκρισης ένα πρωτόκολλο πιο οικείο στο μέσο χρήστη, το TCP, σύμφωνα με το οποίο επιτυγχάνεται η σύνδεση μεταξύ δύο υπολογιστών, καθώς και η αξιόπιστη μεταφορά πακέτων δεδομένων μεταξύ τους, με τη βοήθεια ειδικών μηχανισμών ελέγχου και αξιοπιστίας.

Το UDP επιτελεί την ίδια λειτουργία, δηλαδή την ανταλλαγή πακέτων, με το κυριότερο όμως χαρακτηριστικό του να είναι η παντελής έλλειψη μηχανισμών ελέγχου και αξιόπιστης ανταλλαγής πακέτων, όπως αυτά που συναντώνται στο TCP.

Έτσι, κατά τη διάρκεια της ανταλλαγής UDP πακέτων ανάμεσα σε δύο υπολογιστές, είναι δυνατό να έχουμε σφάλματα όπως τα εξής:

- Άφιξη των πακέτων στο τερματικό του παραλήπτη με λανθασμένη σειρά
- Πολλαπλή άφιξη του ίδιου πακέτου ή και
- Καταστροφή ενός πακέτου, το οποίο δε φτάνει ποτέ στον παραλήπτη, λόγω φόρτου του δικτύου.

Η έλλειψη αυτών των μηχανισμών μπορεί να καθιστά το πρωτόκολλο UDP αναξιόπιστο, του δίνει όμως το μεγάλο πλεονέκτημα του να είναι πολύ πιο ελαφρύ από το TCP. Έτσι, το UDP αποτελεί το ιδανικό πρωτόκολλο για εφαρμογές που δεν απαιτούν αξιοπιστία, αλλά η ταχύτητα αποτελεί γι' αυτές κρίσιμο παράγοντα. Χαρακτηριστικά παραδείγματα τέτοιων εφαρμογών που βασίζονται στο UDP αποτελούν οι εφαρμογές audio και video streaming, οι οποίες, όπως μαρτυρά και το όνομά τους, βασίζονται περισσότερο στη γρήγορη μεταφορά πακέτων, ούτως ώστε να έχουμε ταχεία και συνεχή ροή δεδομένων, και ακολούθως συνεχή αναπαραγωγή ήχου ή εικόνας. Κάποιο λανθασμένο ή κατεστραμμένο (χαμένο) πακέτο δεν επηρεάζει την ποιότητα (όπως αυτή γίνεται αντιληπτή από τα αισθητήρια όργανα του ανθρώπου, με βάση τους βιολογικούς περιορισμούς που εισάγουν) του ήχου ή της εικόνας σε τόσο μεγάλο βαθμό, όσο η αργή ταχύτητα μετάδοσης πακέτων, που συνεπάγεται διακοπές στη ροή του ήχου ή της εικόνας.

Παρόλα αυτά, οι μηχανισμοί που λείπουν από το UDP μπορούν να υλοποιηθούν σε επίπεδο εφαρμογής (έχουμε δηλαδή κατά κάποιο τρόπο "software" υλοποίηση του TCP πάνω στο UDP), ελαχιστοποιώντας έτσι την απώλεια της ποιότητας λόγω της αναξιόπιστης μεταφοράς πακέτων. Σαν παράδειγμα, μερικές εφαρμογές audio και video streaming διαθέτουν μηχανισμούς διόρθωσης και παρεμβολής, που αντιμετωπίζουν την αλλοίωση και την απώλεια πακέτων, με αποτέλεσμα ακόμα και με την καταστροφή πακέτων να μη γίνεται από το χρήστη αντιληπτή διακοπή της ροής ήχου/ εικόνας. Κατά κύριο λόγο όμως, ειδικά από στις real time εφαρμογές (VoIP, media streaming, on-line games), αποφεύγεται η υλοποίηση τέτοιων μηχανισμών ή η υπερβολική χρήση τους, γιατί έτσι υποσκελίζεται το κύριο πλεονέκτημα του UDP, η ταχύτητα. Πέρα του video και του audio streaming, άλλες εφαρμογές που βασίζονται στο UDP είναι το DNS (Domain Name Server), το IPTV, το VoIP (Voice over IP), το TFTP (Trivial Transfer Protocol), και τα on-line games. Αναλυτικότερα, ακολουθούν τα κυριότερα χαρακτηριστικά του UDP, πάντα σε σύγκριση με το TCP, ώστε να γίνονται πιο καταληπτά.

Χαρακτηριστικά

Τρόπος ανταλλαγής πακέτων

Στο TCP η ανταλλαγή πακέτων γίνεται με μία διαδικασία εν είδη session, στο οποίο αποστέλλονται όλα τα επιθυμητά πακέτα σαν μία σειρά πακέτων.
Αρχικά πρέπει να επιτευχθεί ένα κανάλι επικοινωνίας μεταξύ των δύο τερματικών, με την εγκαθίδρυση μίας σύνδεσης.
Αυτό επιτυγχάνεται με την ανταλλαγή τουλάχιστον τριών πακέτων, πριν τη μετάδοση οποιουδήποτε “ωφέλιμου” πακέτου πληροφορίας. (έτσι το TCP γίνεται ακόμα πιο βαρύ)
Στο UDP το κάθε πακέτο αποστέλλεται σαν αυτόνομη οντότητα (με αποτέλεσμα να μπορούμε να πούμε ότι δεν υπάρχει η έννοια του session) με τον καλύτερο δυνατό τρόπο αξιοποίησης των πόρων του δικτύου (best effort transmission).
Έτσι δεν αντιστοιχεί στη διοχέτευση ενός μεγάλου όγκου πακέτων μέσα σε ένα κανάλι επικοινωνίας, αλλά στην αποστολή μεμονωμένων πακέτων μέσα από διαφορετικούς κάθε φορά δρόμους.

Σειρά πακέτων

Στο TCP έχει μεγάλη σημασία τα πακέτα να φτάνουν στον παραλήπτη με τη σειρά που αποστέλλονται, και γι’ αυτό το λόγο γίνεται αρίθμηση του κάθε πακέτου.
Έτσι, ακόμα κι αν φτάσουν σε άλλη σειρά, στον παραλήπτη γίνεται χρήση ενός buffer (προσωρινή μνήμη), όπου κατακρατούνται τα πακέτα μέχρι να γίνει παράληψη και σωστή αναδιανομή τους. Έτσι, η ροή των πακέτων από το buffer προς το χρήστη θα γίνεται με τη σωστή σειρά.
Στο UDP δε γίνεται αρίθμηση, με αποτέλεσμα τα πακέτα να δύνανται να φτάσουν σε τυχαία σειρά.

Αξιοπιστία

Στο TCP έχουμε μηχανισμούς διασφάλισης της ποιότητας επικοινωνίας, όπως:
- Επιβεβαίωση επιτυχούς παραλαβής πακέτου
- Επαναποστολή κατεστραμμένων πακέτων
- Καθορισμός ελάχιστου χρόνου για την παραλαβή κάθε πακέτου (timeout)

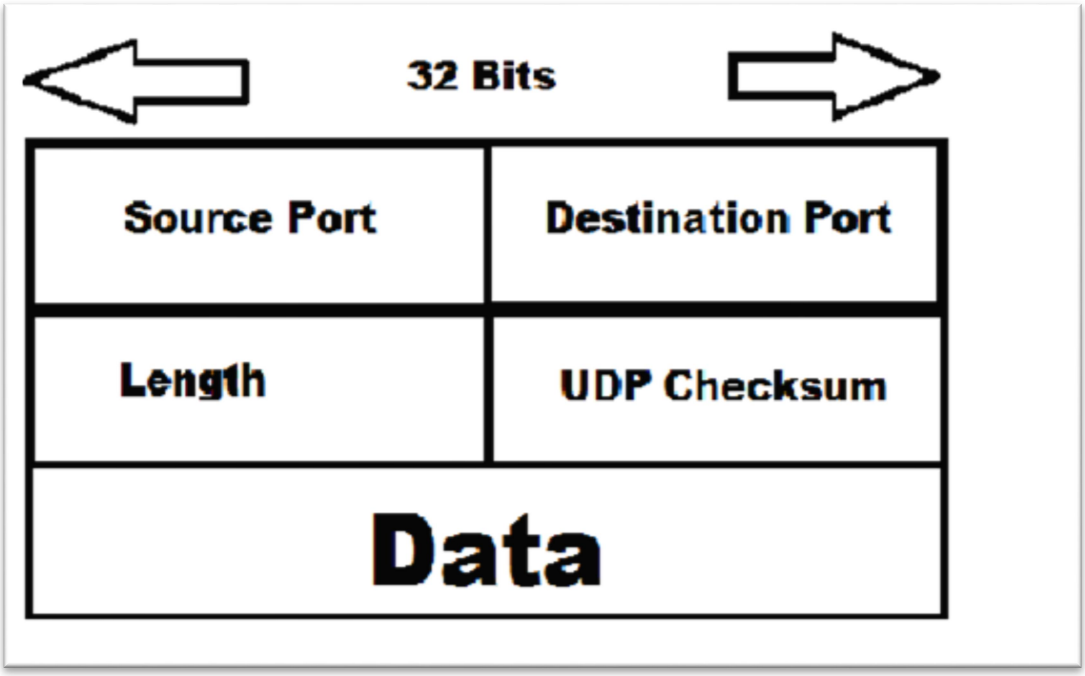
Αυτά επιτυγχάνονται κυρίως με τη χρήση “ερωταποκρίσεων” ACK/ NACK.
Έτσι, είναι σίγουρη η ασφαλής και επιτυχής μετάδοση κάθε πακέτου, και μάλιστα με τη σωστή σειρά.
Στο UDP η επικοινωνία γίνεται στα τυφλά, υπό την έννοια ότι μετά την αποστολή ενός πακέτου, ο αποστολέας δεν ξέρει εάν αυτό παραδόθηκε επιτυχώς ή καταστράφηκε ή αλλοιώθηκε κάπου κατά τη διαδρομή του στο δίκτυο.
Δηλαδή δεν υπάρχει διαδραστική επικοινωνία με χρήση μηχανισμών ACK/ NACK ή μηχανισμών επαναποστολής κατεστραμμένων πακέτων.

Βαρύτητα

Το TCP, με τους διάφορους μηχανισμούς διασφάλισης της αξιόπιστης επικοινωνίας, αλλά και με το μηχανισμό επίτευξης ενός καναλιού επικοινωνίας ανάμεσα σε δύο υπολογιστές, παρόλο που είναι αξιόπιστο, χαρακτηρίζεται ως βαρύ.
Το UDP, παρόλη την έλλειψη όλων των προαναφερθέντων, και παρά την αναξιοπιστία του, έχει το σημαντικό πλεονέκτημα της μεγάλης ταχύτητας, αφού είναι κατά πολύ ελαφρότερο.

Δομή UDP πακέτων

Προκειμένου να εμβαθύνουμε στη λειτουργία του UDP, ακολουθεί περιγραφή της δομής του UDP πακέτου



Κάθε πακέτο αποτελείται χονδρικά από δύο μέρη:

- τα δεδομένα (data), και
- την κεφαλίδα (header)

Η κεφαλίδα, περιέχει τέσσερα πεδία, που περιγράφουν τα χαρακτηριστικά του πακέτου. Παρόλο που τα τέσσερα πεδία θεωρούνται λίγα σε σύγκριση με άλλα πρωτόκολλα, τα δύο από αυτά (τα ροζ) είναι προαιρετικά.

Ειδικότερα:

Στο πεδίο Source Port δίνεται η πόρτα του αποστολέα, από την οποία προέρχεται το πακέτο. Έτσι, σε περίπτωση που έχουμε επιτυχή παράδοση στον παραλήπτη, και αυτός θέλει να στείλει κάποιο πακέτο – απάντηση, η αποστολή θα πρέπει να γίνει προς αυτή την πόρτα. Επειδή η χρήση του είναι προαιρετική, όταν δε χρησιμοποιείται έχει την τιμή μηδέν.
(Η προαιρετική φύση του πεδίου αυτού γίνεται εκμεταλλεύσιμη από κακόβουλους χρήστες που κάνουν επίθεση flood attack σε κάποιον server, με απώτερο σκοπό την επίτευξη άρνησης παροχής υπηρεσιών (DoS – Denial Of Service) από το server (αν και η πιο συνήθης τακτική έγκειται στη χρήση TCP πακέτων).
Έτσι, ο επιτιθέμενος μπορεί είτε να αφήσει κενό αυτό το πεδίο των πακέτων είτε να χρησιμοποιήσει μία τυχαία IP, εξασφαλίζοντας έτσι τόσο την ανωνυμία του όσο και την μη επιρροή του από τα πακέτα απόκρισης (ICMP Destination Unreachable) από το server.)
Στο πεδίο Destination Port δίνεται η πόρτα του παραλήπτη στην οποία θα γίνει η αποστολή του πακέτου.
Εδώ δίνεται το μέγεθος του πακέτου σε bytes. Θεωρητικά, μιας και το πεδίο αυτό έχει μέγεθος 16 bits, η μικρότερη δυνατή τιμή που περιέχει είναι 8 bytes (όσο και το μέγεθος της κεφαλίδας αυτής καθ' εαυτής) ενώ το μέγιστο 65527 bytes (αν και

πρακτικά είναι 65507 bytes λόγω περιορισμών που εισάγουν πρωτόκολλα επόμενων στρωμάτων στο OSI μοντέλο)
Τέλος το πεδίο Checksum χρησιμοποιείται για την εξακρίβωση της ορθής ανταλλαγής του πακέτου (σαν σύνολο, κεφαλίδα και δεδομένα), χωρίς να έχει υποστεί αλλοίωση.
Εδώ πρέπει να σημειωθεί ότι, ένα ακόμη πιο ελαφρύ πρωτόκολλο είναι το UDP Lite, σύμφωνα με το οποίο γίνεται παραλαβή πακέτων ακόμα και όταν έχουν μη έγκυρο checksum (βλέπε λίγο παρακάτω, στη δομή των UDP πακέτων), και χρησιμοποιείται κυρίως σε multimedia ή VoIP εφαρμογές, όπου είναι προτιμότερη η παραλαβή αλλοιωμένου πακέτου, σε σχέση με την απώλεια του.
Μετά την αποστολή του, το UDP πακέτο περνάει στο επίπεδο του δικτύου, όπου προστίθεται και μία επιπλέον κεφαλίδα, ώστε να γίνει η σωστή μετάδοσή του κατά μήκος του δικτύου.
Αυτή η επικεφαλίδα όμως δεν εξετάζεται εδώ, μιας και είναι αντικείμενο άλλων πρωτοκόλλων του επιπέδου του δικτύου (IPv4 ή IPv6).

Audio streaming Protocols

Οι audio streaming εφαρμογές, που ανήκουν στην κατηγορία των streaming media, είναι ένα είδος εφαρμογών που χαρακτηρίζονται από τη συνεχή ροή δεδομένων, καθώς και τη συνεχή αναπαραγωγή τους στον τελικό χρήστη με τη μορφή ήχου, σχεδόν αμέσως κατά την άφιξή τους στον τελικό προορισμό τους.
Αυτή είναι και η κυριότερη διαφορά των streaming media σε σχέση με πιο “παραδοσιακά” μέσα μετάδοσης πληροφοριών, όπως τα CDs και οι κασέτες.
Επίσης, ο όρος αποδίδεται κυρίως στη μεταφορά πληροφοριών μέσω τηλεπικοινωνιακών δικτύων, διαχωρίζοντας έτσι αυτήν την κατηγορία από την παραδοσιακή τηλεόραση και το ραδιόφωνο.
Προκειμένου να γίνει εφικτή η υλοποίηση των streaming media και ειδικότερα των audio streaming εφαρμογών, σχεδιάστηκαν αρκετά πρωτόκολλα που τυποποιούν όλες τις απαραίτητες διαδικασίες.
Τα βασικότερα ίσως πρωτόκολλα είναι ίσως τα datagram πρωτόκολλα, όπως το UDP (User Datagram Protocol), σύμφωνα με το οποίο η αποστολή των δεδομένων υλοποιείται με μία ροή μικρών πακέτων δεδομένων.
Παρόλο που αυτή η ιδέα είναι απλή και αποτελεσματική, πρέπει να λαμβάνεται υπόψη το γεγονός ότι τα πακέτα μπορούν εν δυνάμει να χαθούν ή να αλλοιωθούν κατά τη μεταφορά τους κατά μήκος του δικτύου.
Ανάλογα με το χρησιμοποιούμενο πρωτόκολλο και την τάξη των απωλειών των πακέτων, τα δεδομένων στην πλευρά του τελικού χρήστη μπορούν να ανακτηθούν με ανάπτυξη, σε επίπεδο εφαρμογής, ειδικών τεχνικών διόρθωσης σφαλμάτων για αλλοιωμένα πακέτα, τεχνικών παρεμβολής ώστε να μη γίνεται αισθητή η ολοκληρωτική καταστροφή πακέτων, ειδάλλως ο τελικός χρήστης αναγκάζεται να υποστεί μέχρι και στιγμιαία διακοπή της αναπαραγωγής του ήχου στις audio streaming εφαρμογές, ακριβώς επειδή διακόπτεται και η ροή δεδομένων προς αυτών με την απώλεια κάποιων πακέτων.
Τα πρωτόκολλα:

- RTSP (Real-time Streaming Protocol),
- RTP (Real-time Transport Protocol) και το
- RTCP (Real-time Transport Control Protocol)

σχεδιάστηκαν ειδικά για stream media πάνω σε δίκτυα.
Ειδικά τα δύο τελευταία, σχεδιάστηκαν να “λειτουργούν” “πάνω” στο UDP.
Το RTSP είναι ένα πρωτόκολλο που επιτρέπει έναν client να χειρίζεται από απόσταση έναν streaming media server, με τη χρήση εντολών σαν αυτές που συναντώνται στις οικιακές συσκευές ψυχαγωγίας, όπως “play” ή “pause”, και κάνει εφικτή την time-based πρόσβαση σε αρχεία σε έναν server.

Το RTP καθορίζει ένα συγκεκριμένο format πακέτων για τη μετάδοση ήχου και video στο internet.
Δεν λειτουργεί σε μία συγκεκριμένη TCP ή UDP πόρτα, αλλά ο μόνος περιορισμός που υπάρχει είναι ότι η UDP επικοινωνία λαμβάνει χώρα διαμέσου μίας άρτιας πόρτας, ενώ η αμέσως επόμενη περιττή πόρτα δεσμεύεται σύμφωνα με το RTCP.
Συμπεραίνουμε λοιπόν, ότι το RTP συναντάται σε συνδυασμό με το RTCP, και "λειτουργούν" πάνω στο UDP.
Συνήθως το RTP χειρίζεται real-time δεδομένα, όπως interactive ήχος και video.
Έτσι, συναντάται συχνά σε εφαρμογές videoconferencing ή Voice over IP.
Οι εφαρμογές που βασίζονται στο RTP είναι γενικά λιγότερο ευαίσθητες στην απώλεια πακέτων, αλλά σχετικά ευαίσθητες στην καθυστέρηση άφιξης των πακέτων, έτσι το UDP είναι πολύ καλύτερη επιλογή από το TCP για τέτοιου είδους εφαρμογές.
Το πρωτόκολλο δεν παρέχει μηχανισμούς εξασφάλισης της επιτυχούς παράδοσης των πακέτων, ενώ και η παράδοσή τους σε λανθασμένη σειρά είναι πιθανή.
Επίσης δεν παρέχονται εγγυήσεις Quality of service, αν και όλα αυτά μπορούν να εξασφαλιστούν με κάποιον άλλον τρόπο (πχ το πρωτόκολλο δίνει όλα τα εφόδια ώστε να αναπτυχθεί σε επίπεδο εφαρμογής μηχανισμός που να αναδιατάσσει σε σωστή σειρά τα αφιχθέντα πακέτα).
Το RTCP είναι το "αδελφό" πρωτόκολλο του RTP, καθώς όπως προαναφέρθηκε συναντώνται πάντα σε συνδυασμό, και παρέχει διάφορες πληροφορίες που κάνουν δυνατό τον έλεγχο της ροής δεδομένων που βασίζεται στο RTP.
Δηλαδή, παρόλο που "συμπληρώνει" το RTP κατά την κατάτμηση σε πακέτα και τη μεταφορά των multimedia δεδομένων, δε μεταφέρει το ίδιο δεδομένα.
Χρησιμοποιείται ώστε να μεταδοθούν περιοδικά πακέτα ελέγχου στους συμμετέχοντες σε μία streaming media σύνοδο.
Κύρια λειτουργία του είναι να παρέχει ένα είδος feedback σχετικά με το quality of service που παρέχεται από το RTP.
Έτσι, συλλέγει στατιστικά δεδομένα μίας media σύνδεσης, και πληροφορίες όπως τα απεσταλμένα bytes, τα απεσταλμένα πακέτα, τα χαμένα – κατεστραμμένα πακέτα, το jitter, ο χρόνος απόκρισης, κ.ά.
Μια εφαρμογή μπορεί να χρησιμοποιήσει αυτές τις πληροφορίες ώστε να βελτιώσει το quality of service που παρέχεται, με τον πιθανό περιορισμό της ροής δεδομένων, ή της χρήσης ενός codec χαμηλής αντί υψηλής συμπίεσης.
Σε αντίθεση με το UDP, υπάρχουν πιο αξιόπιστα πρωτόκολλα, όπως το TCP, το οποίο μπορεί να εγγυηθεί την επιτυχή μετάδοση και παράδοση κάθε bit της ροής των media δεδομένων.
Παρόλα αυτά, αυτό επιτυγχάνεται μέσω ενός συστήματος με timeouts και επανεκπομπές πακέτων, που το κάνει ιδιαίτερο πολύπλοκο στην υλοποίηση.
Επίσης, όταν έχουμε καταστροφή και απώλεια ενός πακέτου κατά τη διαδρομή του στο διαδίκτυο, οι μηχανισμοί αυτοί συνεπάγονται τη διακοπή της ροής των δεδομένων έτσι ώστε να γίνει αντιμετώπιση αυτής της απώλειας με επανεκπομπή του χαμένου πακέτου.
Η εφαρμογή του τελικού χρήστη μπορεί να καταπολεμήσει το ενδεχόμενο της διακοπτόμενης ροής δεδομένων με τη χρήση μνήμης buffer, ώστε να μη διακόπτεται η αναπαραγωγή του ήχου προς το χρήστη.

Παρατηρούνται επίσης οι εξής κατηγορίες πρωτοκόλλων
Σύμφωνα με τα unicast πρωτόκολλα ο server αποστέλλει προς τον κάθε client ένα ξεχωριστό αντίγραφο του media stream.
Σαν ιδέα μπορεί να είναι απλό, αλλά στην πράξη μπορεί να οδηγήσει στην μαζική κυκλοφορία αυτών των δεδομένων στο δίκτυο.
Αντίθετα, τα multicast πρωτόκολλα στέλνουν μόνο ένα αντίτυπο της media ροής μέσω μίας οποιασδήποτε σύνδεσης, για παράδειγμα στη διαδρομή άμεσα σε δύο routers του δικτύου.

Αυτός είναι ένας πιο αποτελεσματικός τρόπος εκμετάλλευσης της χωρητικότητας του δικτύου, αλλά πολύ πιο πολύπλοκος στην υλοποίηση.
Ένα από τα πιο σημαντικά multicast πρωτόκολλα, το IP πρωτόκολλο, πρέπει να υλοποιηθεί σε όλους τους κόμβους που μεσολαβούν ανάμεσα στο server και στον client, συμπεριλαμβανομένων και των router του δικτύου.
Σύμφωνα με αυτό, αναπτύχθηκε μία μέθοδος προώθησης IP datagrams σε ένα γκρουπ ενδιαφερομένων παραληπτών – clients.
Από το 2005 όμως, οι περισσότεροι routers στο internet δεν υποστηρίζουν το IP multicast, και πολλά firewalls το μπλοκάρουν.
Το IP multicast είναι πιο πρακτικό για οργανισμούς που αναπτύσσουν τα δικά τους δίκτυα, όπως πανεπιστήμια και εταιρίες, καθώς, απ' τη στιγμή που στήνουν ιδιόκτητα routers και συνδέσμους δικτύου, μπορούν να αποφασίσουν κατά πόσο το οικονομικό κόστος και η όλη προσπάθεια της υποστήριξης του IP multicast αξίζει σε σχέση με την τελική οικονομία στο bandwidth.
Τα P2P (Peer-to-peer) πρωτόκολλα ορίζουν ένα νέο τρόπο μετάδοσης των δεδομένων, σύμφωνα με τον οποίο αυτά μεταδίδονται από clients που τα έχουν ήδη, προς clients που δεν τα έχουν και τα επιθυμούν.
Αυτό αποτρέπει το server και τις δικτυακές του συνδέσεις από το φαινόμενο bottleneck.
Παρόλα αυτά, ανακύπτουν τεχνικά θέματα, θέματα απόδοσης, ποιότητας, ενώ πολύ σοβαρά είναι επίσης και τα νομικά θέματα, όπως έχει διαφανεί τα τελευταία χρόνια.
Παρακάτω έχουμε μία εισαγωγική αναφορά στον εξοπλισμό που χρησιμοποιήθηκε στην εργασία, που μπορεί να επηρεάζει τις μετρήσεις που διεξάγονται (px bandwidth σύνδεσης), ενώ έπειτα ακολουθεί το κύριο μέρος της εργασίας.

Ανάλυση της εφαρμογής

Προκειμένου να ανταποκριθούμε στις απαιτήσεις τις εργασίας, έπρεπε να αναπτύξουμε μία εφαρμογή σε γλώσσα java, η οποία σε γενικές γραμμές
•Συνδέεται μέσω του διαδικτύου με το server
•Επικοινωνεί με αυτόν μέσω συνδέσεων datagram
•Δέχεται ένα μεγάλο αριθμό UDP πακέτων echo
•Δέχεται συνεχή ροή UDP πακέτων εικόνας από τη webcam του εργαστηρίου, στα πλαίσια ad-hoc υλοποίησης ενός πρωτοκόλλου video streaming
•Δέχεται συνεχή ροή UDP πακέτων ήχου από το server του εργαστηρίου, στα πλαίσια ad-hoc υλοποίησης ενός πρωτοκόλλου audio streaming/

(Ειδικότερα για το τμήμα που αφορά το audio streaming, η ροή των πακέτων δεν είναι στην πραγματικότητα συνεχής, μιας και η ανάπτυξη των threads διαχήρησης της συνεχούς ροής πακέτων και της εισαγωγής στους στη buffer memory της εφαρμογής, όπως και το thread εξαγωγής τους από το buffer με σταθερό ρυθμό προς την παραγωγή του ήχου είναι προαιρετικά. Έτσι η αναπαραγωγή του ήχου γίνεται σε δύο ξεχωριστά στάδια, λήψης πακέτων και εισαγωγής στο buffer, και έπειτα εξαγωγής με σταθερό ρυθμό και αναπαραγωγή του ήχου. Λεπτομέρειες αναφέρονται στην αντίστοιχη μέθοδο του κώδικα java)
Εκμεταλλευόμενοι το γεγονός ότι η java ακολουθεί τη φιλοσοφία του δομημένου προγραμματισμού, κάθε αυτόνομος "υποτομέας" του προγράμματος αναπτύχθηκε σε μία μέθοδο, για ευκολότερη συγγραφή αλλά και αποσφαλμάτωση.
Όλες οι μέθοδοι βρίσκονται σε μία κλάση μόνο, η οποία περιέχει μερικές global μεταβλητές για τη διευκόλυνση της επικοινωνίας των μεθόδων.

Προκειμένου να διευκολυνθεί η ανάπτυξη της εφαρμογής, χρησιμοποιούνται μερικές built-in βιβλιοθήκες της java, οι οποίες θα πρέπει να εισαχθούν στο project της εφαρμογής, στην αρχή του κώδικα (αμέσως μετά τη δήλωση του package), με εντολές import.

```
import java.awt.image.BufferedImage;
import java.io.*;
import java.net.*;

import javax.imageio.ImageIO;
import javax.sound.sampled.*;
import javax.swing.*;
```

Οι κυριότερες βιβλιοθήκες είναι οι εξής:

- ✓ javax.net

που διευκολύνει τη διαχείριση των δικτυακών πόρων (network resources)

- ✓ java.io

που διευκολύνει τη διαχείριση υπολογιστικών πόρων

- ✓ javax.sound.sampled

που διευκολύνει την εγγραφή ψηφιακού ήχου στον υπολογιστή

Οι java.awt, και javax.swing έχουν σχέση με το γραφικό περιβάλλον της εφαρμογής, που είναι ένα απλό παράθυρο που εμφανίζει κάποια μηνύματα - εικόνες (είτε ενημερωτικά είτε μηνύματα σφάλματος) κατά τη διάρκεια του audio και video streaming, ενώ ειδικά κατά το video streaming απεικονίζει και το τρέχον frame της webcam του εργαστηρίου.
Γι' αυτήν την απεικόνιση θα πρέπει να διαβάζονται οι εικόνες που μόλις έχουν αποθηκευτεί στο σκληρό δίσκο, κάτι το οποίο γίνεται με τη βοήθεια του javax.imageio.

Η εφαρμογή περιλαμβάνει 2 κλάσεις, την *userApplication* και την *Network.*

<u>Η κλάση Network</u>
Είναι υπεύθυνη για την σύνδεση μεταξύ των 2 τερματικών, την δημιουργία των απαραίτητων Datagram Sockets και Datagram Packages τα οποία υλοποιούν την μεταφορά πακέτων στις επιμέρους μεθόδους.
Ακόμη περιλαμβάνει τις μεθόδους που αναφέραμε προηγουμένως, οι οποίες υλοποιούν τις συγκεκριμένες διεπαφές και μεταφορά πακέτων μεταξύ των 2 τερματικών.
Είναι υπεύθυνη για τον ορισμό των παραμέτρων που λαμβάνουμε από το εικονικό εργαστήριο για την επικοινωνία μας με τον Server Ιθάκη.

<u>Η μέθοδος ECHO (int min)</u>
Η μέθοδος αυτή είναι αρμόδια για την επικοινωνία request/response, και λήψης των Echo_Request πακέτων όπως αυτά ορίζονται από το εικονικό εργαστήριο.
Το όρισμα min που δίνεται αφορά την χρονική διάρκεια επικοινωνίας με την Ιθάκη για λήψη των echo_Request πακέτων. Ο χρόνος αυτός ορίζεται από τον χρήστη ο οποίος τρέχει την εφαρμογή μέσα από την main().
Να αναφέρουμε ότι στην μέθοδο echo(int min) υπολογίζονται και αποθηκεύονται σε αντίστοιχα αρχεία οι μετρήσεις για τους χρόνους μετάδοσης πακέτων, χρόνους αποστολής και λήψης πακέτων, χρόνους καθυστέρησης και ρυθμαπόδοση. Τα αντίστοιχα αρχεία φέρουν τις πιο κάτω ονομασίες:

- ✓ echofile.txt
- ✓ thoughput.txt
- ✓ transmission.txt
- ✓ time.txt

Η μέθοδος ECHO_noDelay (int min)

Η μέθοδος αυτή είναι αρμόδια για την επικοινωνία request/response, και λήψης των Echo_Request πακέτων, με απενεργοποιημένη την τυχαία καθυστέρηση που ορίζει ο server, όπως αυτά ορίζονται από το εικονικό εργαστήριο.
Το όρισμα min που δίνεται αφορά την χρονική διάρκεια επικοινωνίας με την Ιθάκη για λήψη των echo_Request πακέτων. Ο χρόνος αυτός ορίζεται από τον χρήστη ο οποίος τρέχει την εφαρμογή μέσα από την main().
Να αναφέρουμε ότι και στην μέθοδο echo(int min) υπολογίζονται και αποθηκεύονται σε αντίστοιχα αρχεία οι μετρήσεις για τους χρόνους μετάδοσης πακέτων, χρόνους αποστολής και λήψης πακέτων, χρόνους καθυστέρησης και ρυθμαπόδοση. Τα αντίστοιχα αρχεία φέρουν τις πιο κάτω ονομασίες:

- ✓ Echofile2.txt
- ✓ Thoughput2.txt
- ✓ Transmission2.txt
- ✓ Time2.txt

Στο σημείο αυτό πρέπει να αναφέρω ότι οι αιτήσεις στον server για αποστολή κάθε επόμενου πακέτου γίνονται με κάθε 500ms καθώς αν δεν παρεμβάλουμε την καθυστέρηση αυτή στην αίτηση νέων πακέτων λαμβάνουμε timeout στα 10sec. Δηλαδή σε τυχαίο χρόνο και χωρίς να έχει οριστεί από εμάς, η λήψη νέων πακέτων σταματά.
Στην εφαρμογή έχω προνοήσει να διακόπτεται η σύνδεση αν για 10sec δεν λάβουμε πακέτο μετά από αίτηση.
Αυτό γίνεται με την εξής εντολή:
receive.setSoTimeout(10000);
Η παρεμβολή καθυστέρησης σε κάθε νέα αίτηση στον server γίνεται με τον εξής τρόπο:
Thread.sleep(500);

Η μέθοδος image (int camera)

Η μέθοδος αυτή αφορά την λήψη εικόνων από την κάμερα που βρίσκεται στον διακομιστή. Το όρισμα camera έχει σκοπό τον ορισμό της κάμερας από την οποία θέλουμε να λάβουμε την αντίστοιχη εικόνα και μπορεί να πάρει 2 τιμές, «0» και «1». Η παράμετρος αυτή ορίζεται από τον χρήστη της εφαρμογής μέσα από την main(). Τα byteStreams που λαμβάνουμε από την Ιθάκη αποθηκεύονται σε ένα αρχείο .jpeg με την ονομασία *"image3.jpeg*".

Η μέθοδος getTemp ()

Η μέθοδος αυτή είναι υπεύθυνη για την λήψη πακέτων που περιλαμβάνουν τις θερμοκρασίες, τις οποίες καλούμαστε να λάβουμε στα πλαίσια της εργασίας.
Οι αντίστοιχες μετρήσεις των θερμοκρασιών αποθηκεύονται σε ένα αρχείο με την εξής ονομασία: *temperature.txt*

Η μέθοδος Audio (int numofpack, int song)

Η συγκεκριμένη μέθοδος είναι υπεύθυνη για την αναπαραγωγή της μουσικής που λαμβάνουμε από τον server, στα ηχεία του υπολογιστή μας.
Τα δύο ορίσματα που φέρει η μέθοδος έχουν σκοπό τον ορισμό από τον χρήστη της εφαρμογής, του αριθμού των πακέτων που ζητά να λάβει και τον αριθμό του τραγουδιού αντίστοιχα.
Ο αριθμός των bits του κβαντιστή ορίστηκε στα 16 για να λειτουργεί και στις 2 περιπτώσεις:

- ✓ DPCM
- ✓ AQDPCM

Η μέθοδος getdpcm (int numofpack, int song)

Η μέθοδος αυτή είναι υπεύθυνη για την αποκωδικοποίηση των πακέτων ήχου με

κωδικοποίηση DPCM που λαμβάνουμε
Οι τιμές των δειγμάτων που λαμβάνουμε από τον server αποθηκεύονται σε ένα αρχείο με ονομασία «DPCMmusicXX», όπου XX είναι ο αριθμός του τραγουδιού που όρισε ο χρήστης της εφαρμογής και τον οποίο περνάμε σαν όρισμα στην μέθοδο Audio().

Η μέθοδος getaqdpcm (int numofpack, int song)

Η μέθοδος αυτή είναι υπεύθυνη για την αποκωδικοποίηση των πακέτων ήχου με κωδικοποίηση AQDPCM που λαμβάνουμε.
Οι τιμές των δειγμάτων που λαμβάνουμε από τον server αποθηκεύονται, μετά από την σχετική επεξεργασία, σε ένα αρχείο με ονομασία «AQDPCMmusicXX», όπου XX είναι ο αριθμός του τραγουδιού που όρισε ο χρήστης της εφαρμογής και τον οποίο περνάμε σαν όρισμα στην μέθοδο *Audio().*

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. https://www.tutorialspoint.com/difference-between-tcp-and-udp
2. https://www.tutorialspoint.com/java/java_networking.htm
3. https://www.tutorialspoint.com/internet_technologies/internet_protocols.htm
4. https://www.tutorialspoint.com/java_nio/java_nio_datagram_channel.htm
5. https://en.wikipedia.org/wiki/OSI_model
6. https://study.com/academy/lesson/the-udp-protocol-characteristics-structure.html
7. http://zetcode.com/java/socket/
8.https://web.mit.edu/java_v1.5.0_22/distrib/share/docs/api/java/net/DatagramSocket.html
9. http://web.mit.edu/2.993/www/lectures/lecture5.html