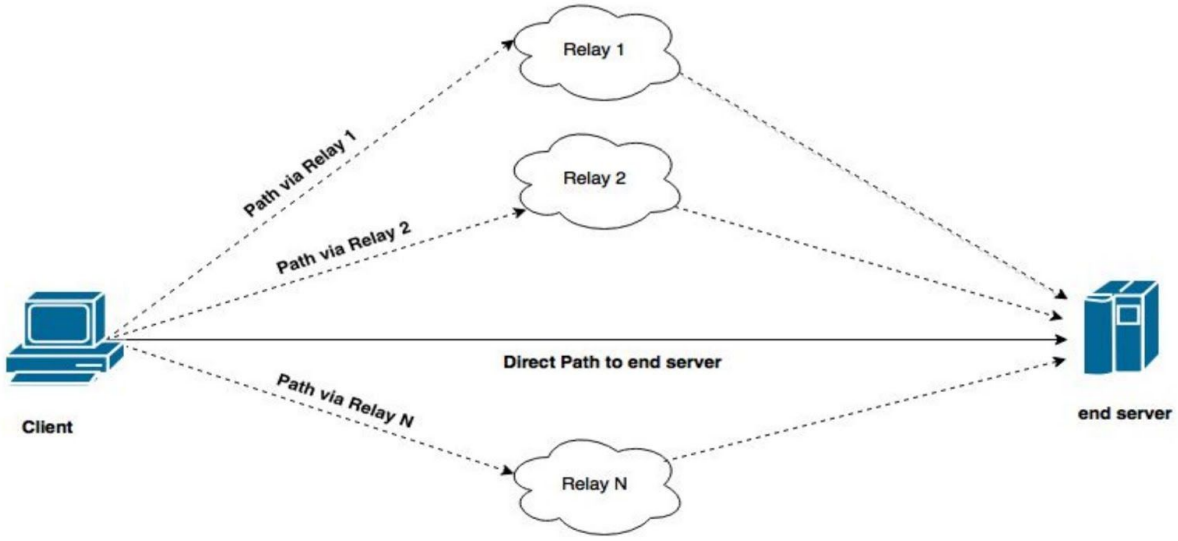
HY-335b
Δίκτυα υπολογιστών
Ευστράτιος Γελαγώτης 2784
Στέφανος Αυραμάκης 2707

Project Spring 2017
Building Short Anonymous Networks



Εισαγωγή

Η γλωσσα προγραμματισμού που χρησιμοποιήθηκε ήταν η Python 2.7.6.

Η υλοποίηση του Project είχε αρκετά στάδια που απαιτούσαν δημιουργία πολλών συναρτήσεων. Πολλές απο αυτές τις συναρτήσεις υλοποιούνται στο αρχείο myFunctionsLib.py επειδή υλοποιούν συναρτήσεις που καλούνται απο τους Relay Nodes και απο τον Client. Στην υλοποίηση μας ήταν απαραίτητο να χρησιμοποιηθούν Threads και Sockets. Όλα τα αρχεία θα αναλυθούν λεπτομερώς στη συνέχεια.

Αρχείο: myFunctionsLib

Class Relay_node_object

Δημιουργήσαμε την κλάση αυτή η οποία περιέχει μέσα το Allias, την IP διεύθυνση και το Port του Relay Node. Χρησιμοποιήται για να αποθηκεύμε οργανομένα πληροφορίες για κάθε ένα Relay Node που χρησιμοποιούμε.

get_average_ping(end_server,num_of_tests)

Η συνάρτηση αυτή εκτελεί όσα ping tests έχει ζητήσει ο χρήστης να εκτελέσει είτε ο client στον end_server είτε απο τον client στο relay node και μέσω του relay node προς τον end server.

Εδώ χρησιμοποίται το διαδικτυακό εργαλείο του λειτουργικού συστήματος Linux, ping.

end_server: Περιέχει το link προς τον server που θέλουμε να κάνουμε ping.

num_of_tests: Περιέχει τον αριθμό τον φορών που θέλουμε να κάνουμε ping στον end_server.

Η συνάρτηση καλεί το εργαλείο ping εσωτερικά. Το ping όταν καλείται μέσα στις πληροφορίες που δίνει για κάθε φορά που κάνει ping μας δίνει και το μέσο Round Trip Time. Επεξάργάζοντας το string που επιστρέφεται παίρνουμε το μέσο RTT και το επιστρέφουμε.

Αν για κάποιο λόγο κάποιος Server δεν ανταποκριθεί στο ping, τότε τυπώνεται διαφορετική έξοδος. Αυτή η περίπτωση ελέγχται και τυπώνεται ανάλογο μήνυμα λαθους επιστρέφοντας -1.

get_hops(end_server)

Στην συνάρτηση αυτή εντοπίζουμε πόσα Hops εκτελούνται είτε απο τον client προς τον end_server είτε απο τον client προς το relay node και μέσω του relay node προς τον end server.

Εδω χρησιμοποιήται το διαδικτυακό εργαλέιο Traceroute του λειτουργι κού συτήματος Linux.

end_server: Περιέχει το Link προς τον server που θέλουμε να υπολογίσουμε τα Hops

Η συνάρτησή μας καλεί το εργαλέιο traceroute εσωτερικά. Όταν το εργαλείο αυτό καλείται δινει πληροφορίες για κάθε server απο τον οποίο περνάει μέχρι να φτάσει στον τελικό προορισμό του.

Αρχικά εντοπίζει την IP address του end_server και την επιστρέφει. Έπειτα τυπώνει για κάθε server σημαντικές πληροφορίες όπως η IP address.

Αν το traceroute εχει επιτύχει τότε τελευταίος server που πηγε θα ήταν ο end_server. Συγκρίνοντας την IP address του τελικού προορισμού με την IP address που μας επιστράφηκε στην αρχή του traceroute βλέπουμε ότι IP διευθύνσεις είναι ίδιες. Στην περιπτωση που δεν είναι ίδιες τότε το traceroute απέτυχε.

Αν το traceroute τελειώσει με επιτυχεία τότε επιστεφεται ο αριθμός των Hops. Σε διαφορετική περίπτωση επιστρέφεται -1.

get_filename_from_link(filelink)

Ο σκοπός αυτής της συνάρτησης είναι να μας επιστρέφει το όνομα του αρχείου απο το link (filelink) του αρχείου.

Π.χ.

Από:
https://www.google.com/images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png

Μας επιστρέφει “googlelogo_white_background_color_272x92dp.png ”

Χρησιμοποιείται για την αποθήκευση των αρχείων στο σύστημα μας.

parse_files2download(filename)

parse_relay_nodes_file(filename)

parse_endservers(filename)

Οι 3 παραπάνω συναρτήσεις χρησιμοποιούνται για να διαβάσουν τα αρχεία files2download.txt, relay_nodes.txt και end_servers.txt αντίστοιχα. Η **parse_endservers** επιστρέφει ένα dictionary που περιέχει το allias του κάθε end_server ως Key και link για τον κάθε server ως value.

Η **parse_files2download** επιστρέφει ένα dictionary με το allias του κάθε end server ως Key και το link για το κάθε αρχείο ως Value.

Τέλος η **parse_relay_nodes_file** επιστρέφει ένα dictionary με το allias του κάθε relay_node ως Key ένα Relay_node_object ως Value.

httpResponseFileExists(filelink)

Κατα την υλοποίηση της εργασίας παρουσίασθηκε ένα πρόβλημα. Το πρόβλημα αυτό ήταν ότι κάποιοι σύνδεσμοι για αρχεία ήταν λανθασμένοι ή δεν ίσχυαν πια με αποτέλεσμα το πρόγραμμα μας να έχει διαφορετική συμπεριφορά ή να κάνει λήψη λανθασμένου αρχείου.

Το πρόβλημα αυτό μας λύνει η συνάρτηση αυτή. Αν κάποιο link είναι έγκυρο τότε ο server επιστρέφει HTTP status 200. Στη περίπτωση που δεν είναι έγκυρο μπορεί π.χ να επιστραφεί HTTP status 302. Σε αυτή τη περίπτωση ο Server κάνει ανακατεθυνση σε αλλη ιστοσελίδα. Αυτό έχει ως αποτέλεσμα να γίνει επιτυχώς λήψη κάποιου άλλου ανεπιθύμητου αρχείου. Στην περίπτωση αυτή έκανε λήψη ολόκληρης της HTML σελίδας.

Αν ο server επιστρέψει HTTP status 200 τότε η συνάρτηση επιστρέφει True. Διαφορετικά επιστρέφει False.

direct_download(file_link,allias,fname)

Η συνάρτηση αυτή χρησιμοποιείται για λήψη του αρχείου απο τον end server στον client απευθείας χωρίς τη χρήση ενδιάμεσων κόμβων.

relay_download(file_link,allias)

Η συνάρτηση αυτή χρησιμοποιείται για λήψη του αρχείου απο τον end server στον relay node.

Οι δύο παραπάνω συναρτήσεις λειτουργούν πανομοιότυπα με τη διαφορά ότι κατεβάζουν τα αρχεία σε διαφορετικούς προορισμούς με διαφορετικά ονομάτα.

Σε κάθε συναρτηση ελέγχεται με την **httpResponseFileExists** αν το αρχείο που ζητούμε να ληφθεί υπάρχει. Αν υπάρχει συνεχίζεται η λήψη. Αν το αρχείο ληφθεί επιτυχώς τότε επιστρέφεται True αλλιώς αν δε ληφθεί είτε λόγω προβλήματος κατα τη λήψη είτε επειδή η **httpResponseFileExists** δεν μας το επέτρεψε, επιστρέφεται False.

Αρχείο: relay_node.py

Το αρχείο relay_node.py είναι το αυτό που υλοποιεί τον ενδιάμεσο κόμβο (Relay Node)

Στην αρχή του κώδικα θα δούμε το server address που αποτελείται απο tuple που περιέχει την IP διεύθυνση και το port του ενδιάμεσου κόμβου. Έπειτα δημιουργείται το Socket που θα διαχειρηστεί τη σύνδεση προς τον client.

Στη συνέχεια ο κόμβος αναμένει για κάποια σύνδεση απο την οποία θα πάρει κατάλληλη εντολή να εκτελέσει μια απο τις τρεις λειτουργίες του.

Οι λειτουργίες του κόμβου είναι να κάνει τις απαραίτητες μετρικές προς τον end server που του έχει ζητηθεί, να λάβει το αρχείο που του έχει ζητηθεί και τέλος να τερματίσει τη λειτουργία του.

Στη πρώτη περίπτωση ο κόμβος έχει λάβει μέσω του socket, εκτός απο την εντολή για την λειτουργία που θα εκτελέσει, και ένα μήνυμα που περιέχει τη διεύθυνση του end server και πόσα ping πρέπει να εκτελέσει προς τον end server. Αφού υπολογίσει το μέσο RTT και τον αριθμό των hops, με τη χρήση των **get_average_ping** και **get_hops** αντίστοιχα, τότε αυτά επιστρέφονται στον client μέσω της σύνδεσης που έχει ήδη εγκαθιδρυθεί

Στη δεύτερη όπου λαμβάνει εντολή να κατεβάσει το αρχείο στέλνονται μαζί με την εντολή ο σύνδεσμος για το αρχείο μαζί με το allias του end server. Με τη χρήση της συνάρτησης **relay_download** λαμβάνει το αρχείο απο τον end server. Αν έχει γίνει επιτυχώς η λήψη τότε συνεχίζει στην αποστολή του αρχείου στο client. Αν δεν ήταν επιτυχής η λήψη τότε στέλνεται κατάλληλο μήνυμα στο client και τυπώνεται μήνυμα λάθους απο τη πλευρά του relay_node.

Κατά την αποστολή του αρχείου υπολογίζεται το μέγεθος του αρχείου πριν σταλεί. Έπειτα στέλνεται το μέγεθος του αρχείου στον client με σκοπό ο client να ρυθμίσει κατάλληλα τον buffer του στο μέγεθος του αρχείου που θα λάβει. Στη συνέχεια αποστέλεται το αρχείο μέσω της σύνδεσης στο client.

Τελος από τον κόμβο λαμβάνεται και η τρίτη εντολή απο το client που του λέει να κλείσει το socket και να τερματήσει τη λειτουργία του.

*Να σημειωθεί ότι ανάλογα τις συνθήκες κατα τις οποίες τρέχει ο κόμβος η IP address είναι πολύ πιθανό να αλλάζει. Γι αυτό το λόγο πρέπει να ελέγχεται η IP διεύθυνση του μηχανήματος πριν απο οποιαδήποτε εκτέλεση του relay_node.py.

Αρχείο: client.py

Σε αυτό το αρχείο υλοποιείται η λειτουργικότητα του client.

Καθώς διαβάζουμε το αρχείο client.py βλέπουμε ότι αρχικοποιούνται κάποια Dictionaries.

Τα end_servers και files2download θα περιέχουν πληροφορίες για με key το allias του end server και τιμή το link του end server και το link του αρχείου αντίστοιχα. Στο relay_nodes περιέχεται το allias του relay node και ένα Relay_node_object με πληροφορίες για το Relay node.

Ακολουθούν τρία dictionaries με πληροφορίες για το μέσο RTT απο το client στον relay node, απο το relay node στον end server και τέλος το άθροισμα των 2 παραπάνω για το συνολικό μέσο RTT απο το client στον end server μέσω του relay node. Και στα τρία dictionaries το allias του relay node χρησιμοποιείται ως key.

Στη συνέχεια βλέπουμε τρία ακόμα dictionaries που περιέχουν πληροφορίες για τα HOPS με τον ίδιο τρόπο που κρατούνται για το RTT.

Εδώ βλέπουμε και τη λίστα για τα threads που θα υλοποιηθούν παρακάτω.

Ξεκινώντας να τρέχει το αρχείο διαβάζονται τα αρχεία end_servers.txt, files2download.txt και relay_nodes.txt. Στη συνέχεια ζητήται απο το χρήστη να εισάγει στο command line το allias του

end_server, το πλήθος των εκτελέσεων του ping προς τον end server και το κριτήριο με το οποίο θα γίνει η επιλογή του καλύτερου μονοπατιού προς το end server.

Ακολουθεί έλεγχος για τη σωστή εισαγωγή υπαρκτού end server allias. Αν δεν υπαρχει το allias που δόθηκε ο χρήστης ενημερώνεται με κατάλληλο μήνυμα.

Στη συνέχεια υπολογίζονται το μέσο RTT και το πλήθος των HOPS προς τον end server που ζητήθηκε.

Αν υπάρχει οποιοδήποτε πρόβλημα στον υπολογισμό τους θεωρούνται -1 και εισέρχονται αντίστοιχα στα dictionaries RTT_SUM και HOPS_SUM με κλειδί “direct2server”.

Στη συνέχεια υλοποιουνται οι συναρτήσεις οι οποίες θα κληθούν για να τρέξει ο client.

get_connections_analytics(node, server_picked,num_of_tests)

Αυτή είναι η συνάρτηση που καλείται κατα τη δημιουργία των νημάτων και χρησιμοποιείται για σύνδεση στον ενδιάμεσο κόμβο.

Σκοπός της συνάρτησης αυτής ειναι:

1. Να υπολογισθεί το μέσο RTT, κάνοντας όσα ping ζήτησε χρήστης στην αρχή του προγράμματος, και τα hops προς το relay node .
2. Να στείλει εντολή στο relay node να υπολογίσει το μέσο RTT , κάνοντας όσα ping. ζήτησε χρήστης στην αρχή του προγράμματος, και τα hops προς τον end-server.
3. Να λάβει τις μετρικές του relay node, να τις αποθηκεύσει στο αντίστοιχα dictionaries.

Με σκοπό να αποφευχθεί οποιοδήποτε πρόβλημα με απρόβλεπτες συμπεριφορές από πολλαπλά νήματα που τρέχουν παράλληλα χρησιμοποιούμε κλειδωμένα νήματα.

download_image_from_relay(node,server_allias)

Σε αυτή τη συνάρτηση υλοποιείται η λήψη του αρχείου απο τον relay_node. Η συνάρτηση χρησιμοποιεί νήματα για τη διαχείριση της σύνδεσης.

Αφού εγκαθιδρυθεί σύνδεση με τον relay node τότε η συνάρτηση λειτουργεί ακολουθόντας τα παρακάτω βήματα:

1. Να στείλει εντολη στο κόμβο που επιλέχθηκε ότι πρέπει να κατεβάσει το αρχείο και να το στείλει στο client. Μαζί με την εντολή αποστέλεται και το Link για τη λήψη του αρχείου και το allias του relay node.
2. Να λάβει το μέγεθος του αρχείου που θα ληφθει με σκοπό να ρυθμιστεί ο buffer για τη λήψη του αρχείου.
3. Να γίνει λήψη και αποθύκευση του αρχείου σε κατάλληλο directory. Εδώ είναι το client_downloads.
4. Κατά τη λήψη του αρχείου υπολογίζεται και ο χρόνος που διήρκεσε.

Αν κατά τη λήψη του αρχείου απο τον end server στο relay node υπήρχε κάποιο σφάλμα και δεν ολόκληρώθηκε επιτυχώς, τότε στο βήμα 2 γίνεται λήψη και έλεχγος κατάλληλου μηνύματος για σφάλμα και τυπώνεται κατάλληλο μήνυμα. Αν απο τον ενδιάμεσο κόμβο στον client υπήρχε κάποιο σφάλμα που εμπόδισε τη λήψη του αρχείου τότε ο χρήστης ενημέρωνεται με κατάλληλο μήνυμα.

Στο τέλος της συνάρτησης επιστρέφεται True ή False αν η λήψη ολόκληρώθηκε ή όχι αντίστοιχα.

shutdown_relay_node(node)

Αυτή η συνάρτηση στέλνει κατάλληλη εντολη σε κάθε relay node να τερματίσει τη λειτουργία του. Η διαχείρηση της σύνδεσης γίνεται με τη βοήθεια νημάτων.

get_best_path_based_on_RTT

get_best_path_based_on_HOPS

Στις δύο παραπάνω συναρτήσεις επιλέγεται ποιο είναι το ταχύτερο μονοπάτι βάσει του κριτηρίου hops ή latency που έχει επιλέξει ο χρήστης.

Σκοπός της πρώτης είναι να ελέγξει το μέσο RTT από κάθε κόμβο και την απευθείας σύνδεση να επιλέξει εκείνο το μονοπατι με το μικρότερο RTT και να το επιστρέψει. Αν υπήρχε παραπάνω απο ένα μονοπάτι με την ίδια ελάχιστη τιμή RTT τότε προσπαθεί να επιλέξει ταχύτερο μονοπάτι βασιζόμενος στα HOPS. Εαν πάλι βρέθηκε παραπάνω απο ένα μονοπάτι με τα ελάχιστα HOPS τότε η συνάρτηση μας διαλέγει τυχαία.

Η δεύτερη συνάρτηση εκτελεί παρόμοια διαδικασία ψάχνοντας αρχικά το ταχύτερο μονοπάτι με κριτήριο το πλήθος των HOPS. Αν αποτύχει σε αυτό συνεχίζει χρησιμοποιώντας ως κριτήσριο το μεσο RTT. Αν πάλι αποτύχει τότε διαλέγει τυχαία.

print_Choice_menu

Εδώ απλά τυπώνεται το Menu με τις επιλογές που δίνονται στο χρήστη.

Στη συνέχεια ακολουθεί η βασική λειτουργία του client που καλεί όλες τις παραπάνω συναρτήσεις.

Να σημειωθεί ότι client διαβάζει αυτόματα τα αρχεία end_servers.txt,files2download.txt και relay_nodes.txt για λόγους ευκολίας του χρήστη και του προγραμματιστή.

Όταν client αρχίσει να τρεχει, ζητείται απο το χρήστη να εισάγει το allias του end server, το πλήθος των φορών που θέλει να κάνει ping στον end server και το κριτήρι βάσει του οποίου θα γίνει η επιλογή του ταχύτερου μονοπατιού.

Αφού υπολογισθεί και τυπωθεί το μέσο RTT και το πλήθος των hops από το Client στον end server τότε ο χρήστης καλέιται να επιλέξει ένα απο τα ακόλουθα βήματα:

1. Direct Mode. Απευθείας λήψη του αρχείου απο τον end server.
2. Full Mode. Μπορεί να κάνει μετρικές προς όλους τους relay nodes και να διαλέξει το ταχυτερο μονοπάτι,συμπεριλαμβανομένου και του απευθείας μονοπατιού, για τη λήψη του αρχείου.

3. Exit. Εδώ μπορεί να τερματίσει το πρόγραμμα και να στείλει εντολη σε όλα τα relay nodes με τα οποία έχει επικοινωνήσει να τερματίσουν τη λειτουργία τους.

Στο 1ο βήμα γίνεται η απευθείας λήψη του αρχείου απο τον end server χωρίς να γίνει οποιαδήποτε επικοινωνία με κάποιο ενδίαμεσο κόμβο. Υπολογίζεται ο χρόνος για τη λήψη του αρχείου. Ο χρήστης ενημερώνεται με κατάλληλα μηνήματα για την επιτυχία της λήψης.

Στο 2ο βήμα πραγματοποιήται η διαδικασία για την επιλογή του καλύτερου μονοπατιού.

Αρχικά γίνεται καλείται η thread function **get_connections_analytics** για κάθε relay node με σκοπό να γίνει η συλλογή των μετρικών προς κάθε ένα κόμβο.Στη συνέχεια υπολογίζονται το μέσο RTT και το πλήθος των HOPS απο τον client στον end server μέσω του ενδιάμεσου κόμβου. Συνεχίζοντας καλείται η **get_best_path_based_on_HOPS** ή η **get_best_path_based_on_RTT** ανάλογα το κριτήριο που έχει διαλέξει ο χρήστης για την επιλογή του ταχύτερου μονοπατιού. Σε αυτή τη περίπτωση να αξίζει να σημειωθεί ότι το ταχύτερο μονοπάτι μπορεί να είναι και απευθείας απο το client στον end server. Αν ειναι αυτή η περίπτωση με τη βοήθεια της **direct_download** γίνεται η απευθείας λήψη του αρχείου.

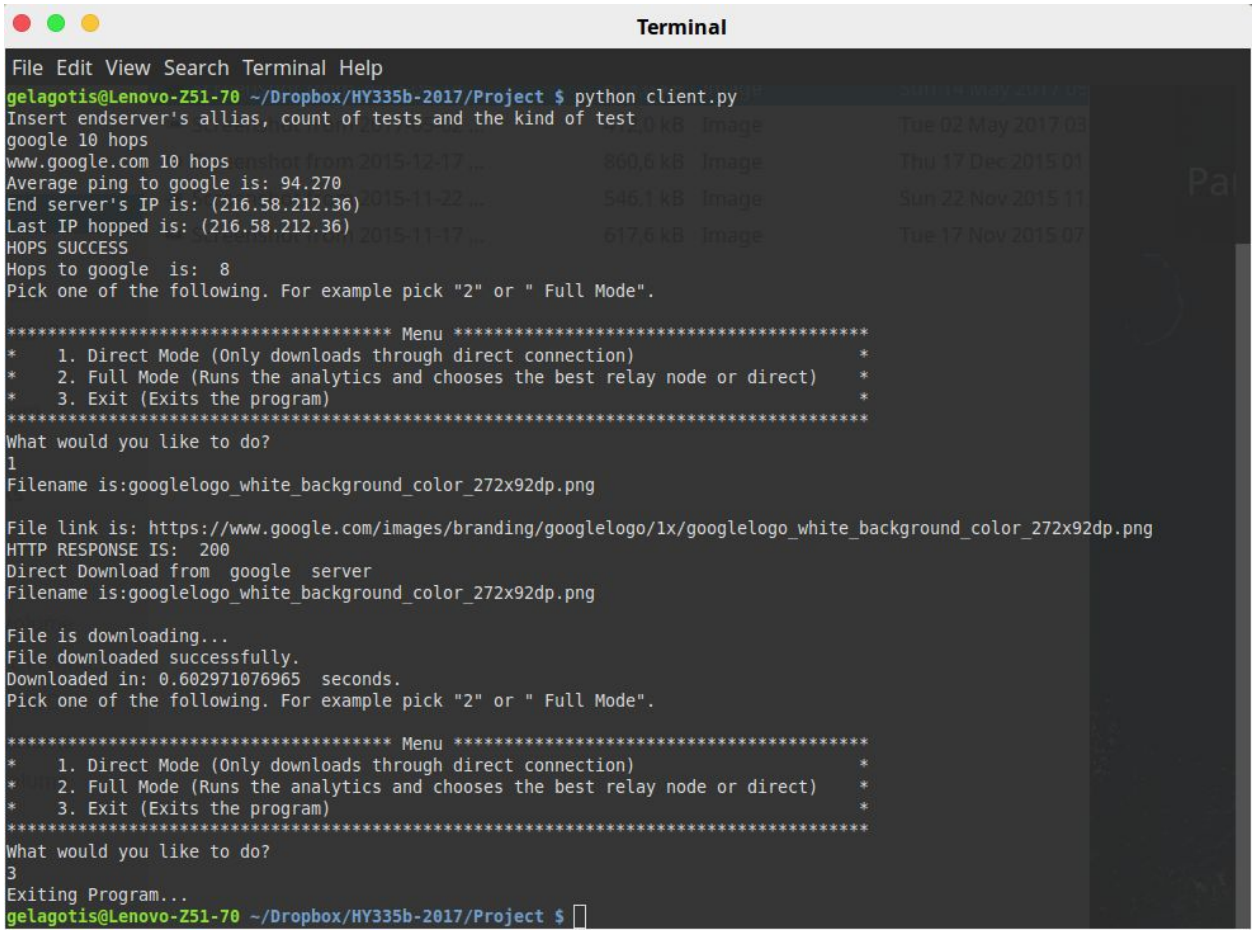
Αν το ταχύτερο μονοπάτι είναι μέσω κάποιου ενδιάμεσου κόμβου τότε καλλείται η Thread Function **download_image_from_relay** που μέσω αυτής θα γίνει η λήψη του αρχείου.

Στο 3ο και τελευταίο βήμα ο χρήστης δίνει εντολή στο client να τερματίσει. Με τη σειρά του ο client δίνει εντολή σε όλους με τους οποίου ήρθε σε επικοινωνία να τερματήσουν μέσω της Thread Function **shutdown_relay_node**. Στη συνέχεια τερματίζει ο client.

Σχολιασμός Αποτελεσμάτων

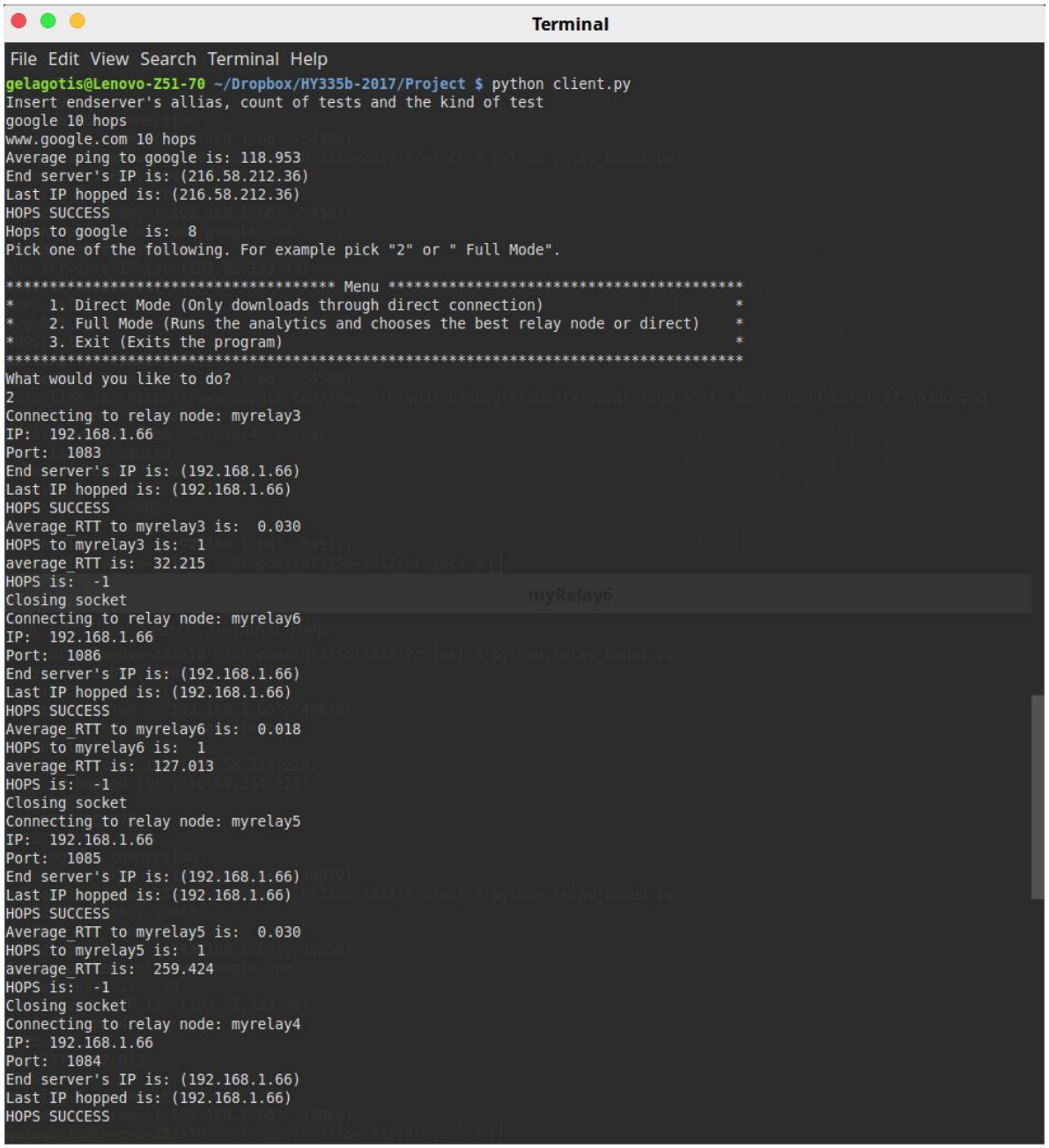
Υπήρχαν περιπτώσεις που κάποιο αποτέλεσμα ήταν αναμενόμενο. Ένα παράδειγμα είναι να γίνει λήψη του αρχείου απο τον end server www.google.com όπου το εργαλέιο ping και το trecaroute επέστραφαν τιμή επιτυχώς κάθε φορα. Αν ο χρήστης διάλεγε ως κριτήριο το πλήθος των HOPS τότε η απευθείας σύνδεση ήταν πάντα ταχύτερη γιατί μέσω οποιουδήποτε ενδιάμεσου κόμβου υπήρχε πάντα τουλάχτιστον ένα hop παραπάνω απο το απευθείας μονοπάτι, δεδομένου ότι τρέχαμε το σύστημα είτε τοπίκα είτε στα Debian της σχολής. Σε διαφορετική περίπτωση το αποτέλεσμα ίσως είναι απίθανο να προβλεφθεί. Όσο αφορά τη χρήση του latency ως κριτηρίου για την επιλογή του ταχύτερου μονοπατιού υπήρχαν αρκετές περιπτώσεις όπου το απευθείας μονοπάτι δεν επιλέχθηκε ώς το ταχύτερο, αλλά κάποιο μονοπάτι μέσω ενδιάμεσου κόμβου.

Στο παρακάτω στιγμιότυπο φαίνεται η επιτυχής λήψη ενος αρχείου απο τον end server www.google.com. Ζητήσαμε να στον client να κάνει 10 φορές ping και να επιλέξει καλύτερο κριτήριο βάσει των hops. Στο μενού που επιλέξαμε Direct Mode για τη λήψη και μετά επιλέξαμε Exit ώστε να τερματίσει το πρόγραμμα.

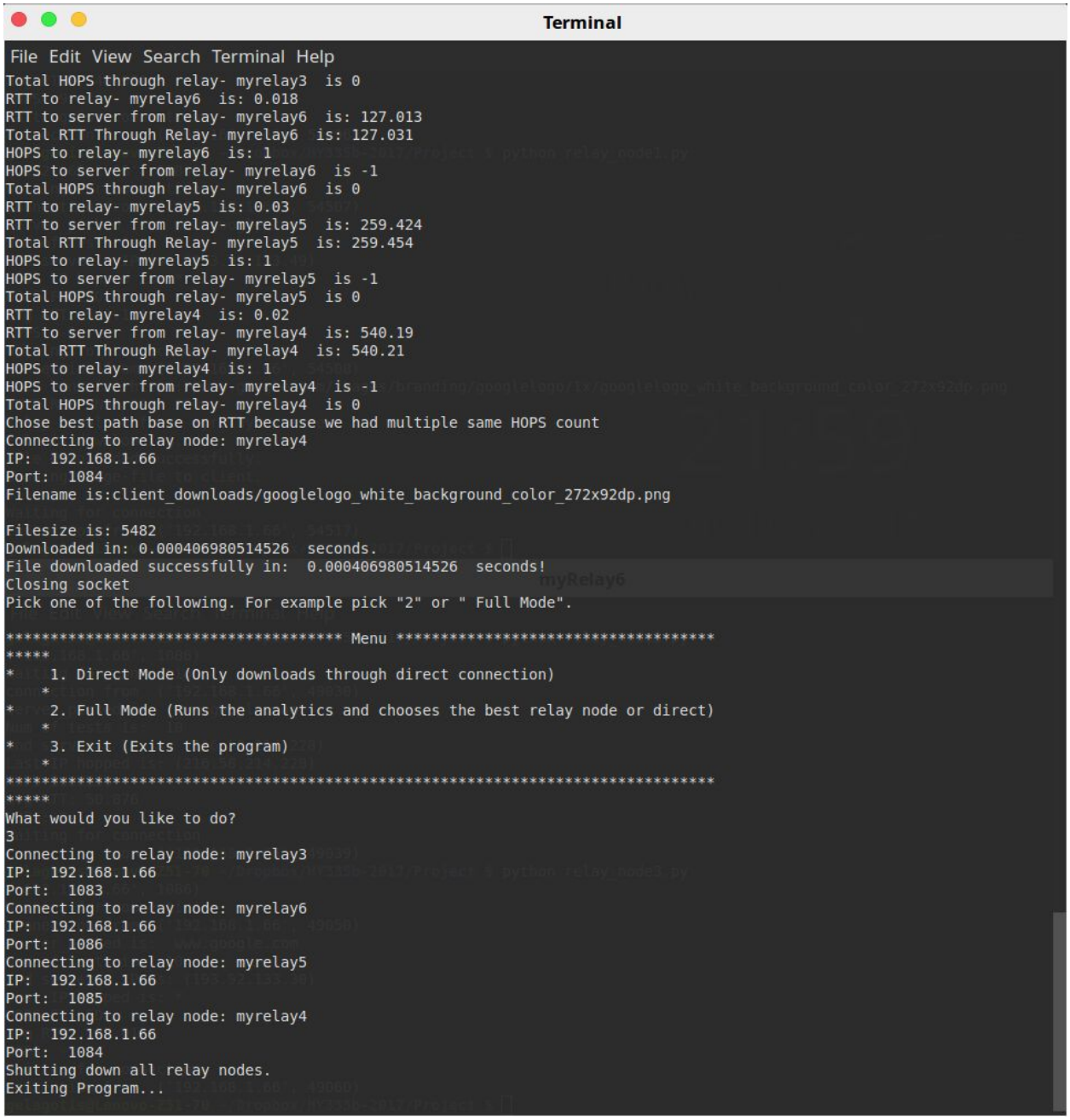


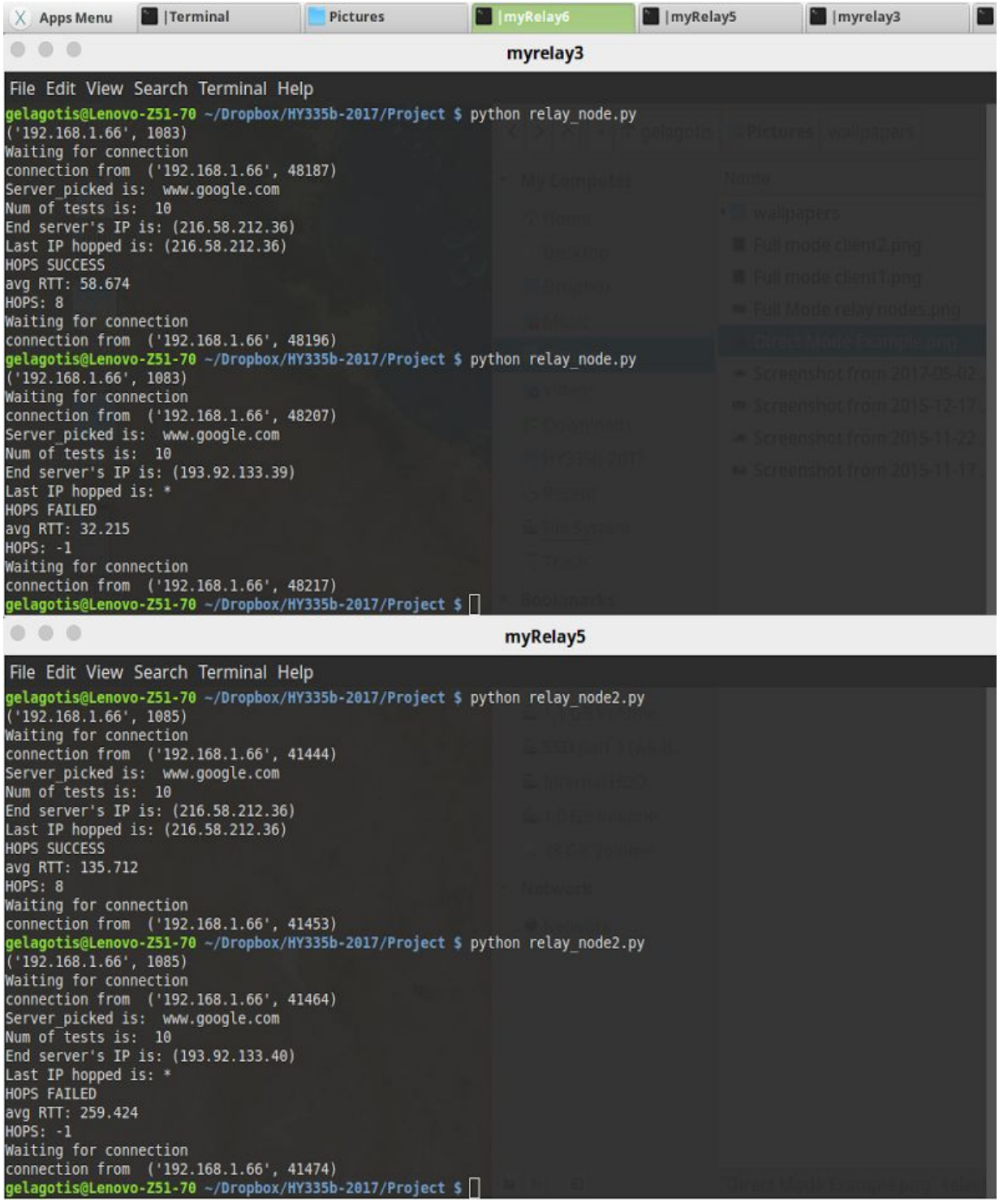
Στα παρακάτω στιγμιότυπα βλέπουμε τι γίνεται στη περίπτωση που το σύστημα τρέξει πλήρως επικοινωνώντας με ενδιάμεσους κόμβους.

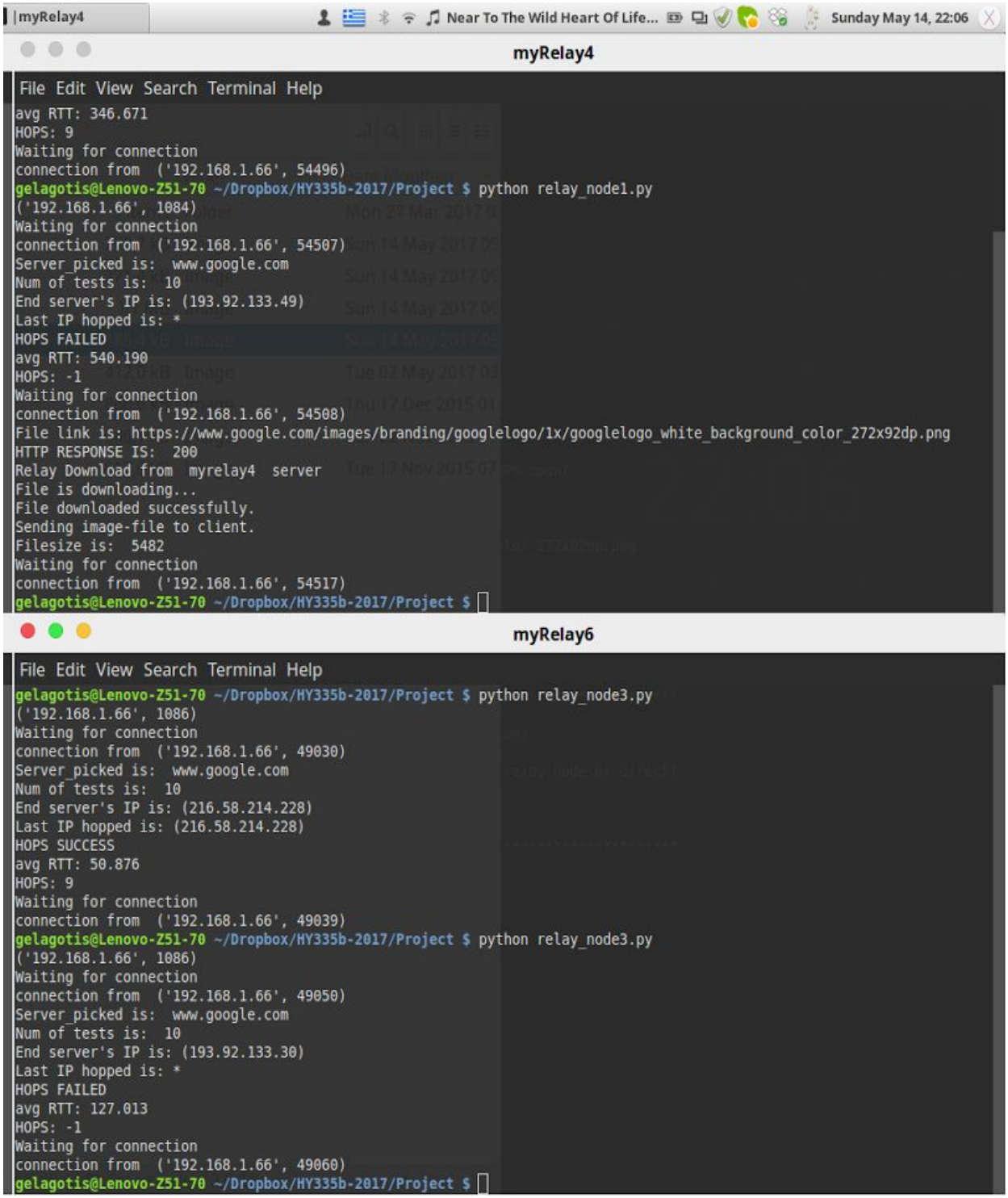
Αρχικα ζητάμε απο το client να συνδεθει στο allias google, να κάνει ping και να διαλέξει καλύτερο μονοπάτι βάσει του πλήθους των hops.



Στη συνέχεια επιλέγουμε Full Mode για τη ληψη του αρχείου και μετά Exit για την έξοδο απο το σύστημα. Βλεπουμε το σχετικό μήνυμα ότι η επιλογή του ταχύτερου μονοπατιού έγινε βάσει του RTT γιατί υπήρχαν μονοπατια με ίδια ελάχιστο πληθος HOPS και η λήψη του αρχείου ολοκληρώνεται επιτυχώς.




Apps Menu
|Terminal
Pictures
|myRelay6
|myRelay5
|myrelay3
myrelay3
File Edit View Search Terminal Help
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node.py
('192.168.1.66', 1083)
Waiting for connection
connection from ('192.168.1.66', 48187)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (216.58.212.36)
Last IP hopped is: (216.58.212.36)
HOPS SUCCESS
avg RTT: 58.674
HOPS: 8
Waiting for connection
connection from ('192.168.1.66', 48196)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node.py
('192.168.1.66', 1083)
Waiting for connection
connection from ('192.168.1.66', 48207)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (193.92.133.39)
Last IP hopped is: *
HOPS FAILED
avg RTT: 32.215
HOPS: -1
Waiting for connection
connection from ('192.168.1.66', 48217)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $
myRelay5
File Edit View Search Terminal Help
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node2.py
('192.168.1.66', 1085)
Waiting for connection
connection from ('192.168.1.66', 41444)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (216.58.212.36)
Last IP hopped is: (216.58.212.36)
HOPS SUCCESS
avg RTT: 135.712
HOPS: 8
Waiting for connection
connection from ('192.168.1.66', 41453)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node2.py
('192.168.1.66', 1085)
Waiting for connection
connection from ('192.168.1.66', 41464)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (193.92.133.40)
Last IP hopped is: *
HOPS FAILED
avg RTT: 259.424
HOPS: -1
Waiting for connection
connection from ('192.168.1.66', 41474)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $


myRelay4
Near To The Wild Heart Of Life...
Sunday May 14, 22:06
myRelay4
File Edit View Search Terminal Help
avg RTT: 346.671
HOPS: 9
Waiting for connection
connection from ('192.168.1.66', 54496)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node1.py
('192.168.1.66', 1084)
Waiting for connection
connection from ('192.168.1.66', 54507)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (193.92.133.49)
Last IP hopped is: *
HOPS FAILED
avg RTT: 540.190
HOPS: -1
Waiting for connection
connection from ('192.168.1.66', 54508)
File link is: https://www.google.com/images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png
HTTP RESPONSE IS: 200
Relay Download from myrelay4 server
File is downloading...
File downloaded successfully.
Sending image-file to client.
Filesize is: 5482
Waiting for connection
connection from ('192.168.1.66', 54517)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $
myRelay6
File Edit View Search Terminal Help
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node3.py
('192.168.1.66', 1086)
Waiting for connection
connection from ('192.168.1.66', 49030)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (216.58.214.228)
Last IP hopped is: (216.58.214.228)
HOPS SUCCESS
avg RTT: 50.876
HOPS: 9
Waiting for connection
connection from ('192.168.1.66', 49039)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $ python relay_node3.py
('192.168.1.66', 1086)
Waiting for connection
connection from ('192.168.1.66', 49050)
Server_picked is: www.google.com
Num of tests is: 10
End server's IP is: (193.92.133.30)
Last IP hopped is: *
HOPS FAILED
avg RTT: 127.013
HOPS: -1
Waiting for connection
connection from ('192.168.1.66', 49060)
gelagotis@Lenovo-Z51-70 ~/Dropbox/HY335b-2017/Project $

Στα δύο παραπάνω στιγμιότυπα βλέπουμε τους ενδιάμεσους κόμβους και ανταποκρίνονται στην επικοινωνία με τον client. Παρατηρούμε ότι στο κόμβο myRelay4 τυπώνεται HTTP RESPONSE 200 που δηλωει ότι το Link για το αρχείο είναι έγκυρο και έτσι γίνεται η λήψη του αρχείου απο τον end server και η αποστολή του στο client.