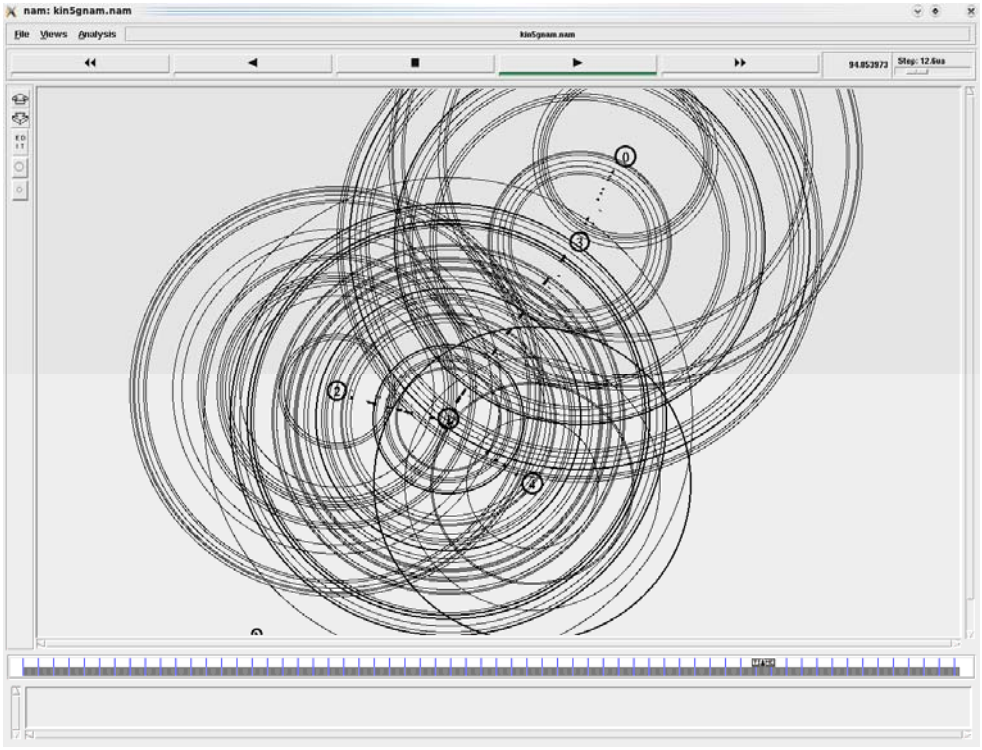
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

«Υλοποίηση πρωτοκόλλου ασυρμάτων τοπικών δικτύων IEEE 802.11g στον προσομοιωτής NS-2»



Του φοιτητή:
Κελεσίδη Κωνσταντίνου

Επιβλέπων καθηγητής:
Πασχάλης Ράπτης

Αρ. Μητρώου:
2234 / 03

Θεσσαλονίκη 2009

Περιεχόμενα

ΠΡΟΛΟΓΟΣ

Η παρούσα διπλωματική εργασία εκπονήθηκε κατά τη διάρκεια του έτους 2009 από τον Κελεσίδη Κωνσταντίνο, φοιτητή του τμηματος Πληροφορικης της σχολής Τεχνολογικών Εφαρομγων του Α.Τ.Ε.Ι. Θεσσαλονίκης.

Η εργασία πραγματεύεται την υλοποίηση του πρωτοκόλλου ασυρμάτων τοπικών δικτύων **IEEE 802.11g** στον προσομοιωτή **NS-2**. Στο θεωρητικό της μέρος αναφέρονται σημαντικές πληροφορίες τόσο για το πρωτόκολλο όσο και για τον προσομοιωτή. Στο πειραματικό της μέρος γίνεται εφαρμογή του πρωτοκόλλου **802.11g** σε 10 σενάρια ασύρματων δικτύων τοπολογίας Ad-Hoc από τα οποία εξάγονται κάποια στατιστικά στοιχεία τα οποία και αναλύονται.

Η υπόδειξη του θέματος έγινε από τον Καθηγητή του τμήματος κ. Πασχάλη Ράπτη, τον οποίο θέλω να ευχαριστήσω για την ανάθεση της πτύχιακης καθώς και για την βοήθεια και τις συμβουλές του κατά την διάρκεια της εκπόνησης της παρούσας εργασίας.

Τέλος θέλω να ευχαριστήσω θερμά την οικογένεια και το φιλικό μου περιβάλλον, που στήριξαν τις προσπάθειές μου σε όλα τα στάδια της υλοποίησης της εργασίας.

ΚΕΦΑΛΑΙΟ 1^{Ο}

Ασύρματα Τοπικά Δίκτυα

Τα δίκτυα υπολογιστών για το σπίτι και τις μικρές επιχειρήσεις είτε για οποιαδήποτε άλλο σκοπό μπορούν να κατασκευαστούν με τη χρήση είτε ενσύρματης είτε ασύρματης τεχνολογίας.

Σε ένα ενσύρματο δίκτυο (LAN), για την σύνδεση των συσκευών χρησιμοποιούνται ομοαξονικά καλώδια ή συνεστραμμένα ζεύγη καλωδίων και προσαρμογείς δικτύου.

Σε ένα ασύρματο τοπικό δίκτυο (WLAN) η αποστολή και λήψη δεδομένων γίνεται μέσω του αέρα με την χρήση ραδιοσυχνοτήτων και υπέρυθρων κυμάτων, για τη μετάδοση πληροφοριών χωρίς φυσική σύνδεση. Στην περίπτωση δε που τα ασύρματα δίκτυα χρησιμοποιούνται για τη διασύνδεση των χρηστών με το βασικό κορμό (backbone) του ενσύρματου δικτύου το σημείο πρόσβασης (access point), συσκευές που χρησιμοποιούνται για αυτό τον λόγο, λαμβάνει, αποθηκεύει προσωρινά (buffer) και μεταδίδει τα δεδομένα μεταξύ του WLAN και του ενσύρματου δικτύου υποδομών. Ένα σημείο πρόσβασης μπορεί να υποστηρίξει μια μικρή ομάδα χρηστών και μπορεί να έχει εμβέλεια από εκατό πόδια σε αρκετές εκατοντάδες.

1.1 Σύγκριση Ασύρματων – Ενσύρματων Δικτύων

Ας εξετάσουμε τα πλεονεκτήματα και τα μειονεκτήματα της κάθε επιλογής.

- Σε ένα ενσύρματο τοπικό δίκτυο (LAN), τα καλώδια που συνδέουν κάθε υπολογιστή μπορεί να είναι άβολα, διότι πρέπει να είναι κρυμμένα σε τοίχους ή τοποθετημένα σε ανοιχτό χώρο έτσι ώστε να μην αποτελούν πηγή κινδύνου.

- Από την άλλη πλευρά, τα καλώδια, τα hubs, τα switches είναι πολύ φθηνότερα σε σχέση με τις αντίστοιχες ασύρματες συσκευές που μπορεί να κοστίζουν τρεις ή τέσσερις φορές περισσότερο.
- Τα ενσύρματα δίκτυα προσφέρουν εύρος ζώνης μέχρι και 100 Mbps.
- Τα ασύρματα δίκτυα που χρησιμοποιούν 802.11b υποστηρίζουν ένα μέγιστο εύρος ζώνης των 11 Mbps, ενώ αυτά που χρησιμοποιούν 802.11a και 802.11g υποστηρίζουν μόνο 54 Mbps.
- Από την άλλη πλευρά, το πλεονέκτημα της μετακίνησης των σταθμών μέσα στα όρια του ασύρματου δικτύου μπορεί να αντισταθμίσει το μειονέκτημα των επιδόσεων.
- Τεχνικώς, τα LANs είναι πιο ασφαλής από ό, τι τα WLANs. Τα σήματα που μεταδίδονται μέσω του αέρα, μπορούν να συλληφθούν και από συσκευές εκτός του δικτύου που απλά έχουν αυτή την δυνατότητα. Ενώ στα ενσύρματα για να υποκλέψει κάποιος τα σήματα θα πρέπει να παρέμβει απευθείας πάνω στο καλώδιο, πράγμα πιο δύσκολο. Ωστόσο, στην πλειονότητα τους, τα ασύρματα δίκτυα προστατεύονται με κάποιο πρότυπο κρυπτογράφησης, το οποίο καθιστά σχεδόν ασύρματες επικοινωνίες εξίσου ασφαλή με εκείνα στα σπίτια ενσύρματα.

ΚΕΦΑΛΑΙΟ 2^{Ο}

Περιγραφή του πρωτοκόλλου 802.11g

Το IEEE 802.11 είναι μια οικογένεια προτύπων της IEEE για ασύρματα τοπικά δίκτυα (WLAN) που είχαν ως σκοπό να επεκτείνουν το 802.3 (Ethernet, το συνηθέστερο πρωτόκολλο ενσύρματης δικτύωσης υπολογιστών) στην ασύρματη περιοχή. Το πρότυπο IEEE 802.11 περιγράφει μόνο τα δύο κατώτερα επίπεδα του OSI (το Physical και το υποεπίπεδο MAC του Data Link), επιτρέποντας έτσι στις όποιες εφαρμογές να εργάζονται όπως ακριβώς θα εργαζόταν πάνω από Ethernet.

2.1 Ιστορικό του 802.11

- Το 1997 η πρώτη έκδοση του 802.11 με δύο μεθόδους διασποράς φάσματος για τη μετάδοση στη ζώνη συχνοτήτων 2,4GHz:
    - FHSS (Frequency Hopping Spread Spectrum) με ρυθμό μετάδοσης 1Mbps, η λειτουργία του εξηγείται παρακάτω.
    - DSSS (Direct Sequence Spread Spectrum) με ρυθμό μετάδοσης 1-2Mbps, η λειτουργία του εξηγείται παρακάτω..
    - Περιλαμβανόταν επίσης και μία υπέρυθρη εκδοχή (IR).

- Το 1999 η έκδοση 802.11b κάνει την εμφάνιση της ανεβάζοντας την ταχύτητα στα 11 Mbps κάνοντας χρήση της μεθόδου DSSS για μετάδοση στη ζώνη συχνοτήτων 2,4GHz. Παράλληλα διατηρήθηκαν οι ρυθμοί λειτουργίας των 1 και 2 Mbps επιτρέποντας στις όποιες συσκευές να πέσουν σε χαμηλότερες ταχύτητες ώστε να διατηρήσουν μια σύνδεση όταν τα σήματα είναι αδύνατα. Σε αυτή την χρονική φάση έχουμε την εξάπλωση των ασύρματων καρτών.
- Την έκδοση 802.11b ακολούθησαν την ίδια χρονιά η έκδοση 802.11a (1999) και λίγα χρόνια μετά η έκδοση 802.11g (2003) κάνοντας χρήση της μεθόδου OFDM για μετάδοση.

- το 802.11a εκπέμπει στη ζώνη συχνοτήτων των 5GHz και υποστηρίζει ταχύτητες μέχρι και 54 Mbps. Η ζώνη συχνοτήτων είναι ο λόγος που το 802.11a δεν είναι συμβατό με τις ασύρματες κάρτες δικτύου οι οποίες υποστηρίζουν 802.11b
- το 802.11g εκπέμπει στη ζώνη συχνοτήτων των 2,4GHz και υποστηρίζει επίσης ταχύτητες μέχρι και 54 Mbps.

• Το 2010 αναμένεται να βγει η έκδοση 802.11n η οποία θα μεταδίδει σε συχνότητες των 2,4GHz αλλά και των 5GHz και θα υποστηρίζει ταχύτητες που θα αγγίζουν τα 600 Mbps.

Έκδοση	Ημερομηνία	Ζώνη συχνοτήτων	Συνήθης ρυθμός μετάδοσης	Ονομαστικός ρυθμός μετάδοσης	Μέθοδοι μετάδοσης	Εμβέλεια εσωτερικών χώρων
802.11	1997	2.4 GHz	0.9 Mbit/s	2 Mbit/s	IR / FHSS / DSSS	~20 m
802.11b	1999	2.4 GHz	4.3 Mbit/s	11 Mbit/s	DSSS	~38 m
802.11a	1999	5 GHz	23 Mbit/s	54 Mbit/s	OFDM	~35 m
802.11g	2003	2.4 GHz	19 Mbit/s	54 Mbit/s	OFDM	~38 m

Εικόνα 2.1, Πίνακας με τα τρέχοντα πρωτόκολλα του 802.11 με τις ζώνες συχνοτήτων που χρησιμοποιούν, τον αριθμό καναλιών, τις τεχνικές διαμόρφωσης σήματος και τις ταχύτητες που μπορούν να επιτύχουν.

2.2 Άλλες Ομάδες Εργασίας του 802.11

Κάποιες από τις υπόλοιπες τρέχουσες ομάδες εργασίας του 802.11 είναι οι εξής:

802.11 d: η ομάδα στοχεύει στην πρόσθεση των απαιτήσεων εκείνων που θα επιτρέψουν στον εξοπλισμό των WLAN 802.11 να λειτουργήσει σε αγορές που δεν εξυπηρετούνται από τα τρέχοντα πρότυπα.

802.11 e: η ομάδα στοχεύει στην ενίσχυση του τρέχοντα μηχανισμού MAC του 802.11, ώστε να επεκτείνει την υποστήριξη για δικτυακές εφαρμογές με απαιτήσεις για ποιότητα υπηρεσιών, να βελτιωθεί η ασφάλεια και να αναβαθμιστούν οι δυνατότητες και η απόδοση του πρωτοκόλλου.

802.11 f: η ομάδα αυτή ασχολείται με τον καθορισμό των απαραίτητων πληροφοριών που πρέπει να ανταλλάσσονται μεταξύ των σημείων πρόσβασης του προτύπου 802.11

802.11 h: η ομάδα στοχεύει στην ενίσχυση των τρεχόντων επίπεδων MAC του προτύπου 802.11 και PHY του 802.11a με επεκτάσεις δικτυακής διαχείρισης και έλεγχου του φάσματος για την διαχείριση της ισχύος μετάδοσης στις ζώνες των 5 GHz. Επιπλέον, στοχεύει στην βελτίωση των μετρήσεων της ενέργειας καναλιού και την ανάπτυξη των κατάλληλων μηχανισμών για δυναμική επιλογή καναλιού και έλεγχο ισχύος μετάδοσης.

802.11 i: η ομάδα στοχεύει στην ενίσχυση του τρέχοντος πρωτόκολλου MAC του προτύπου 802.11, προκειμένου να επιτευχθούν βελτιώσεις στην ασφάλεια.

2.3 Σύντομη Σύγκριση του 802.11g με τα άλλα Πρότυπα

Το 802.11g έχει αποτελέσει επέκταση του 802.11b που ήταν η βάση της πλειοψηφίας των ασύρματων δικτύων.

Λόγο της προς τα πίσω συμβατότητας που υπάρχει, μια ασύρματη κάρτα που υποστηρίζει 802.11b μπορεί να επικοινωνήσει με ένα AP που υποστηρίζει 802.11g και το αντίστροφο στην ταχύτητα των 11 Mbps ή χαμηλότερα ανάλογα με το φάσμα.

Αν σε ένα υπάρχον 802.11b δίκτυο θέλαμε να κάνουμε αναβάθμιση το πρώτο βήμα θα ήταν να αναβαθμίσουμε τα AP ώστε να είναι σε θέση να χρησιμοποιούν 802.11g μέσω μιας απλής αναβάθμισης του Firmware.

Ένα μεγάλο θέμα που απασχολεί τα 2 προαναφερόμενα πρότυπα είναι οι παρεμβολές που προκαλούνται από συσκευές, όπως ασύρματα τηλεφωνά, οι οποίες λειτουργούν στις ίδιες συχνότητες. Αν και αυτό το πρόβλημα γίνεται να διαχειριστεί με περιορισμό των πηγών θορύβου, δεν μπορεί ωστόσο να εξαλείφει.

Η μεγάλη διαφορά του 802.11a με το 802.11g είναι ότι λειτουργεί στην συχνότητα των 5GHz με 12 ξεχωριστά-μη επικαλυπτόμενα κανάλια. Αυτό έχει ως

αποτέλεσμα την δυνατότητα να έχουμε σε μια περιοχή ως και 12 AP χωρίς να παρεμβαίνει το ένα στο άλλο. Το γεγονός αυτό καθίστα εύκολη την ανάθεση ενός καναλιού σε ένα AP και αυξάνει σημαντικά την απόδοση του δικτύου. Επιπλέον στην ζώνη των 5GHz δεν υπάρχουν τόσες παρεμβολές όσες σε αυτή των 2,4GHz. Αυτό συμβαίνει αφενός γιατί η ζώνη των 2,4GHz είναι παγκοσμίως ελεύθερη και αφετέρου γιατί η ζώνη των 5GHz υπόκειται σε νομοθεσίες οι οποίες απαγορεύουν ή περιορίζουν σημαντικά την χρήση της. Στην Ελλάδα αναμένεται πάντως να απελευθερωθεί η χρήση της όπως και σε άλλες ευρωπαϊκές χώρες.

Η συχνότητα λειτουργίας παίζει ρόλο στην απόσταση που μπορεί να καλύψει ένα ασύρματο δίκτυο αλλά και στον αριθμό χρηστών που μπορεί να υποστηρίξει. Με την υψηλότερη συχνότητα που χρησιμοποιεί το πρότυπο 802.11a από τα 802.11b και 802.11g διαφοροποιείται σε 2 πράγματα. Την εμβέλεια και τον αριθμό χρηστών. Έτσι έχουμε για το 802.11a μικρότερη εμβέλεια αλλά περισσότερους χρήστες και για τα πρότυπα 802.11b και 802.11g μεγαλύτερη κάλυψη σε απόσταση αλλά μικρότερο αριθμό χρηστών.

Επίσης ο εξοπλισμός του προτύπου 802.11a είναι ακριβότερος λόγω της μεγαλύτερης συχνότητας λειτουργίας, αλλά και της μικρότερης ζήτησης του προτύπου στην αγορά.

Ένα μεγάλο πρόβλημα με το 802.11a είναι ότι δεν είναι συμβατό με τα 802.11b και 802.11g. δηλαδή ένας χρήστης με ασύρματη κάρτα τεχνολογίας 802.11a δεν μπορεί να συνδεθεί σε ένα AP τεχνολογίας 802.11b ή 802.11g.

2.4 Περιορισμοί συμβατότητας 802.11b και 802.11g

Είναι γνωστό και αναφέρθηκε ήδη πως υπάρχει συμβατότητα μεταξύ 802.11b και 802.11g, ωστόσο λίγοι γνωρίζουν ποιοι είναι οι τεχνικοί περιορισμοί:

1. ένας 802.11b client δεν θα λειτουργήσει αποδοτικότερα αν συνδεθεί με έναν 802.11g router από ότι αν συνδεόταν με έναν 802.11b router.

2. ένας 802.11g client θα λειτουργήσει με χαμηλότερη απόδοση αν συνδεθεί με 802.11b router και για την ακρίβεια θα λειτουργεί με την ταχύτητα του router.

3. όταν σε 1 δίκτυο 802.11g συνδεθούν ανάμεικτα 802.11g και 802.11b clients τότε όλοι θα λειτουργούν στις ταχύτητες των τελευταίων, αν και έχει παρατηρηθεί ότι οι 802.11g clients λειτουργούν λίγο ταχύτερα.

4. θα πρέπει σε ένα ασύρματο δίκτυο να χρησιμοποιείται η ιδία κρυπτογράφηση σε όλες τις συσκευές. Αν αναλογιστούμε ότι οι συσκευές 802.11g μπορούν συχνά να υποστηρίξουν πιο προηγμένες μορφές κρυπτογράφησης από τις αντίστοιχες 802.11b, τότε μπορούμε εύκολα να δούμε το πρόβλημα.

Αξίζει να σημειωθεί πως η συμβατότητα προς τα πίσω του 802.11g με το 802.11b έχει προστατεύσει τις επενδύσεις που είχαν ήδη γίνει. Η διαμόρφωση που χρησιμοποιεί απαιτεί περισσότερη λαμβανόμενη ισχύ, έχει δηλαδή χειρότερη ευαισθησία. Έτσι η εμβέλεια είναι σχετικά μικρότερη από αυτή του 802.11b. Για το λόγο αυτό η χρήση του περιορίζεται για κάλυψη εσωτερικών χώρων, μικρής σχετικά επιφάνειας.

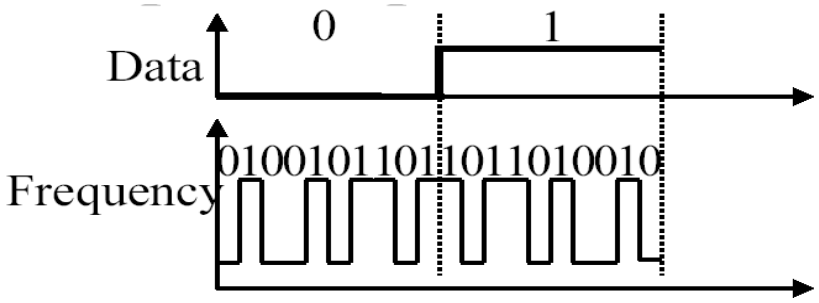
2.5 Τεχνικές DSSS και FHSS

Οι τεχνικές DSSS και FHSS ανήκουν στην οικογένεια Spread Spectrum με κύρια χαρακτηριστικά τους την αντοχή στις παρεμβολές και την δυσκολία υποκλοπής.

2.5.1 DSSS

Η βασική αρχή της DSSS είναι να απλώσει το σήμα σε ένα μεγαλύτερο εύρος συχνοτήτων πολυπλέκοντας το με το «chip code», ώστε να ελαχιστοποιήσει την επίδραση παρεμβολών και του θορύβου. Ονομάζεται "Direct" γιατί το σήμα πληροφορίας, διαμορφώνεται κατευθείαν από τη σειρά κώδικα.

Με αυτή την τεχνική το κάθε bit κωδικοποιείται με τρόπο (chipping code) που είναι γνωστός μόνο στον πομπό και τον δέκτη. Ο λόγος chip/bit ονομάζεται και «spreading ratio» και υποδηλώνει τον βαθμό διασποράς στη ζώνη συχνοτήτων. Η αντίστροφη διαδικασία γίνεται στον δέκτη όπου από τα chips παίρνουμε τα bit.



Εικόνα 2.2, παράδειγμα κωδικοποίησης με 10 chip

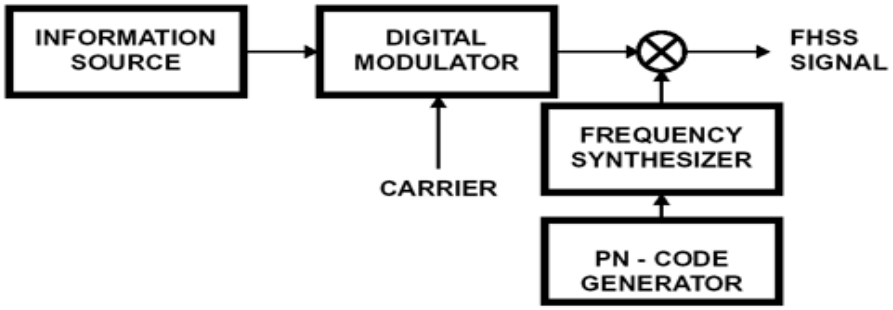
Το πλεονέκτημα της τεχνικής είναι πως και θύμα υποκλοπής να πέσουν τα δεδομένα μας δεν θα είναι εύκολο να τα αναγνωρίσουν γιατί υπάρχουν χιλιάδες τέτοια είδη κωδικοποίησης.

Χρησιμοποίει τεχνική διαμόρφωσης DBPSK(differential binary PSK) για μετάδοση δεδομένων σε ταχύτητα 1Mbps και DQPSK (differential quadrature PSK) για μετάδοση σε ταχύτητα 2Mbps.

Το μεγαλύτερο πλεονέκτημα του είναι η απλότητα αδειοδότησης και η επίτευξη μεγάλων ρυθμών μετάδοσης.

2.5.2 FHSS

Χρησιμοποιεί έναν αριθμό από κανάλια **στενής ζώνης** (η ζώνη των 2.4GHz χωρίζεται σε 75 κανάλια εύρους 1MHz) και εκπέμπει διαδοχικά μεταπηδώντας από συχνότητα σε συχνότητα με μία **προκαθορισμένη σειρά**. Η σειρά αυτή είναι γνωστή και συμφωνημένη στον πομπό και στον δέκτη. Η επιλογή της σειράς των καναλιών γίνεται με ένα **ψευδοτυχαίο τρόπο**. Ο μέγιστος χρόνος εκπομπής σε κάθε κανάλι είναι 400msec.



Εικόνα 2.3, Δημιουργία FHSS σήματος

Χρησιμοποιεί τις 2-level GFSK (Gaussian FSK) και 4-level GFSK για μετάδοση ταχύτητας 2Mbps.

Βασικά πλεονεκτήματα της τεχνικής αυτής είναι οι μικρές παρεμβολές, που επιτρέπουν ταυτόχρονη λειτουργία πολλών συσκευών, μεγαλύτερη αντοχή σε υποκλοπές και δυνατότητα roaming (αναλύεται παρακάτω).

Συνοψίζοντας βλέπουμε ότι η DSSS παρέχει ρυθμούς μέχρι **11 Mbps** , αλλά είναι ευαίσθητη σε παρεμβολές. Η FHSS παρέχει ρυθμούς μόνο μέχρι **3Mbps**, αλλά είναι εξαιρετικά στιβαρή τεχνολογία με εξαιρετική επίδοση σε περιβάλλον που υπάρχουν παρεμβολές , ανακλάσεις. Η τεχνολογία αυτή παρέχει εξαιρετικές υλοποιήσεις κυψελών, παρέχοντας αξιοπιστία στη μετάδοση.

ΚΕΦΑΛΑΙΟ 3^{Ο}

Συσκευές Ασύρματου Δικτύου

Οι συσκευές που παίρνουν μέρος σε ένα ασύρματο δίκτυο είναι οι Ασύρματες Κάρτες Δικτύου, τα Ασύρματα Σημεία Πρόσβασης και οι Ασύρματοι Δρομολογητές.

3.1 Ασύρματες Κάρτες Δικτύου (Wireless NICs)

Είναι οι συσκευές που κάνουν ένα σταθμό ικανό να λαμβάνει και να στέλνει ραδιοκύματα. Όπως οι κάρτες δικτύου Ethernet, οι ασύρματες κάρτες χρησιμοποιούν την τεχνική ή τις τεχνικές με τις οποίες έχουν ρυθμιστεί, προκειμένου να κωδικοποιούν σε ραδιοσήμα μια ροή δεδομένων.

Στην δεκαετία του 1990 οι ασύρματες κάρτες συνδέονταν σε υποδοχές PCMCIA στους φορητούς υπολογιστές. Σήμερα είναι ακόμα διαθέσιμες αλλά συνήθως είναι ενσωματωμένες στις μητρικές. Για τους υπολογιστές γραφείου υπάρχουν PCI ασύρματες κάρτες αλλά και USB συσκευές που χρησιμοποιούνται για αυτό τον λόγο.

Εικόνα 3.1, ασύρματες κάρτες δικτύου.

3.2 Ασύρματα Σημεία Πρόσβασης (Access Points - AP)

Ένα AP συνδέει ασύρματους χρήστες ή σταθμούς σε ένα ενσύρματο τοπικό δίκτυο. Οι σταθμοί δεν επικοινωνούν μεταξύ τους απευθείας αλλα μέσω του AP.

Ένα AP μετατρέπει τα TCP/IP πακέτα από την 802.11 μορφή των frames στη 802.3 μορφή για δίκτυα Ethernet και το αντίστροφο.

Σε ένα ασύρματο δίκτυο οι χρήστες πρέπει να μπορούν να έρθουν σε επικοινωνία με το AP για να αποκτήσουν πρόσβαση στις υπηρεσίες του δικτύου.

Ένα AP είναι μια επιπέδου 2 (στο μοντέλο του OSI) συσκευή που λειτουργεί όπως ένα Ethernet hub. Τα ραδιοκύματα είναι ένα μέσο το όποιο είναι διαμοιραζόμενο και έτσι ένα AP μπορεί και αντιλαμβάνεται όλη την «κίνηση». Και εδώ όπως και στο Ethernet οι συσκευές που θέλουν να χρησιμοποιήσουν το μέσο θα πρέπει να «διαγωνιστούν» για αυτό. Το πώς γίνεται αυτό θα το αναφέρουμε αργότερα.

Εικόνα 3.2, διάφορα access points που κυκλοφορούν στην αγορά.

3.3 Ασύρματοι Δρομολογητές (Wireless Routers)

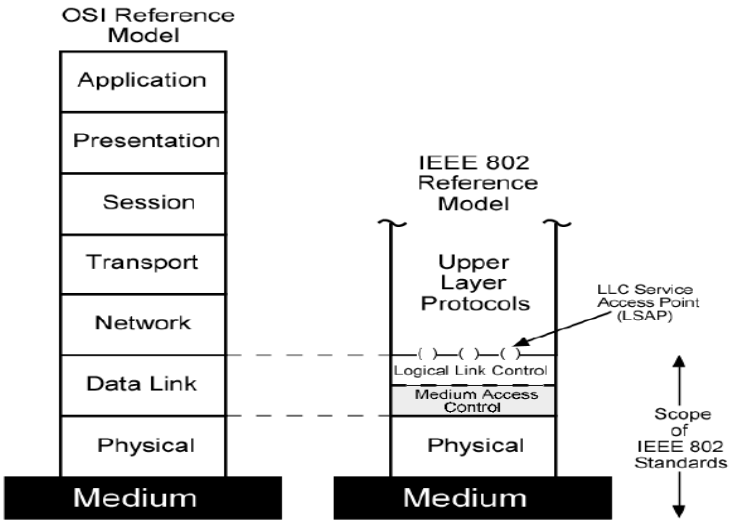
Οι ασύρματοι δρομολογητές παίζουν τον ρόλο του AP, του Ethernet Switch και του δρομολογητή. Είναι ουσιαστικά τρεις συσκευές σε μια. Δηλαδή εκτελεί όλες τις τυπικές λειτουργίες ενός AP όπως αυτές αναφέρθηκαν, με ένα ενσωματωμένο switch 10/100 μπορεί να προσφέρει ενσύρματη συνδεσιμότητα σε άλλες συσκευές ή σε περισσότερα του ενός δίκτυα και τέλος η λειτουργία του δρομολογητή προσφέρει ένα gateway για σύνδεση σε άλλα δίκτυα.

Εικόνα 3.3, διάφορα Wireless Routers που κυκλοφορούν στην αγορά

ΚΕΦΑΛΑΙΟ 4^{Ο}

Επίπεδο MAC

Το δεύτερο επίπεδο του OSI το Data Link χωρίζεται σε δυο υποεπίπεδα, το LLC (logical link control) και το MAC (medium access control). Το υποεπίπεδο MAC λειτουργεί ως interface μεταξύ του φυσικού επιπέδου και της συσκευής του δικτύου. Υποστηρίζει ενσύρματα και ασύρματα δίκτυα.

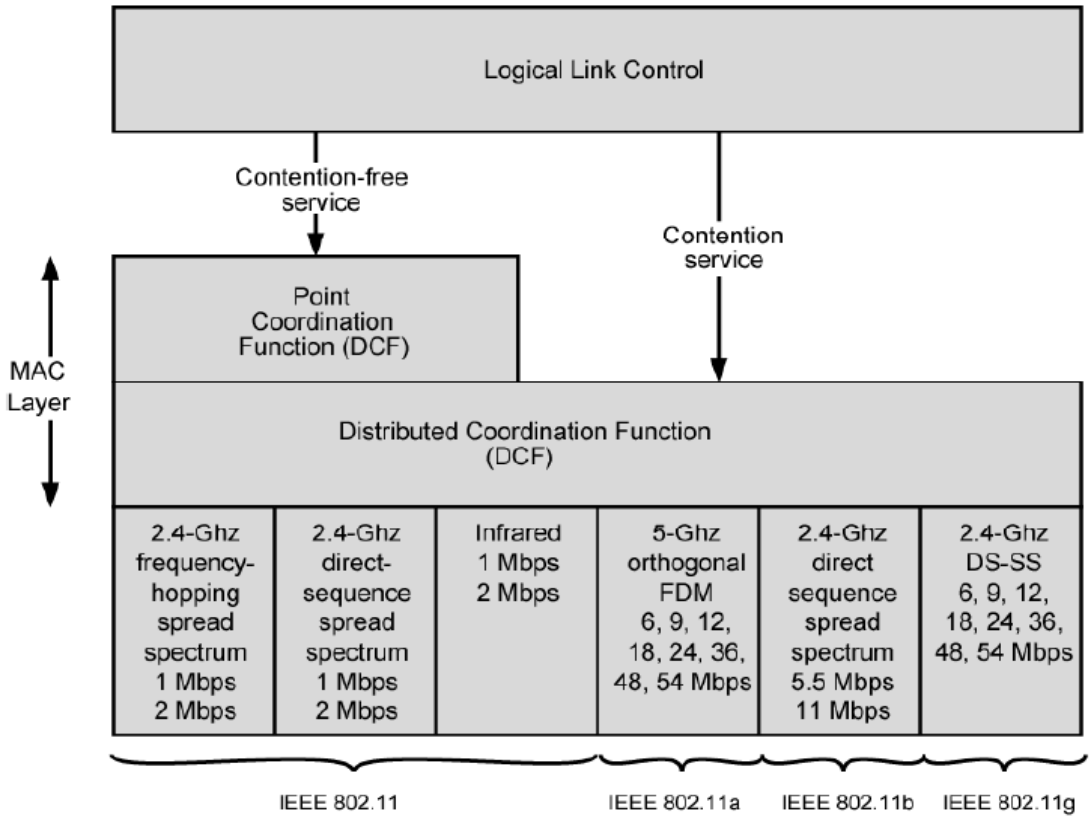


Εικόνα 4.1, επίπεδα του OSI και επίπεδο MAC.

Το 802.11 καλύπτει το φυσικό επίπεδο και το επίπεδο MAC. Το πρωτόκολλο που χρησιμοποιείται για την πρόσβαση στο μέσο είναι το DCF (distributed coordination function) το οποίο αναφέρεται επίσης ως «βασική μέθοδος πρόσβασης». Το DCF είναι ένα CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) πρωτόκολλο.

Επίσης η ομάδα εργασίας του 802.11 περιέλαβε στο πρωτόκολλο την υπηρεσία PCF (point coordination function), η οποία προσφέρεται σε ασύρματα δίκτυα με υποδομή και υποστηρίζει χρονικά εξαρτώμενες υπηρεσίες μέσω ενός μηχανισμού που δεν απαιτεί ανταγωνισμό.

Η DCF λειτουργεί πάνω από το φυσικό επίπεδο και πάνω της λειτουργεί η PCF, χρησιμοποιώντας έτσι τις υπηρεσίες που προσφέρονται στην προηγούμενη.



Εικόνα 4.2, DCF και PCF

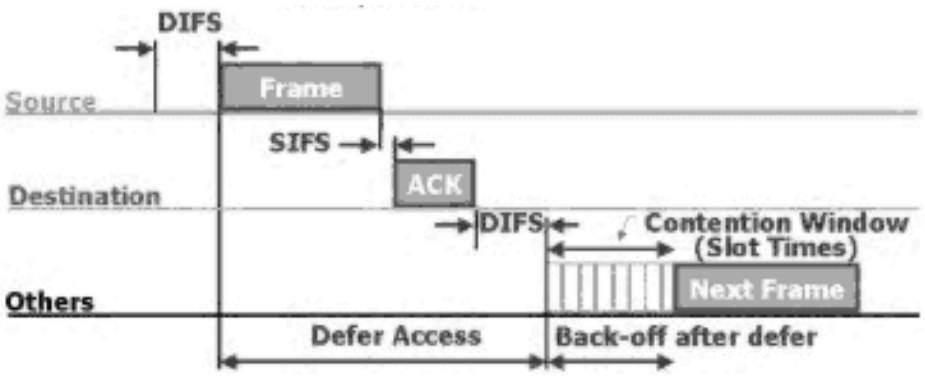
Το επίπεδο MAC παρέχει επιπλέον μηχανισμούς πιστοποίησης, κρυπτογράφησης, ασφάλειας και εξοικονόμησης ενέργειας.

4.1 Πρόσβαση στο Μέσο

Σύμφωνα με την DCF απαιτείται κάθε σταθμός που έχει να μεταδώσει κάτι, να «ακούσει» πρώτα το μέσο. Αν το κανάλι είναι ελεύθερο για χρονικό διάστημα ίσο με ένα IFS τότε μπορεί να εκπέμψει το πακέτο.

Σε διαφορετική περίπτωση, αν ανιχνευτεί κίνηση στο μέσο, θα συνεχίσει να παρατηρεί το κανάλι μέχρι να το βρει ελεύθερο για χρονικό διάστημα ίσο με IFS.

Ο χρόνος μετά την παρέλευση ενός IFS χωρίζεται σε χρονοθυρίδες (slots) και κάθε σταθμός επιτρέπεται να εκπέμπει μόνο κατά την έναρξη μιας χρονοθυρίδας και όχι στο μέσο της. Όταν η τρέχουσα αποστολή τελειώσει ο σταθμός θα περιμένει για ακόμα ένα IFS. Αν το μέσο είναι ελεύθερο τότε περιμένει για ακόμα ένα τυχαίο πλήθος χρονοθυρίδων και έπειτα στέλνει το πακέτο. Παρόλα αυτά αν δυο ή περισσότεροι σταθμοί εκπέμψουν την ίδια στιγμή τότε θα συμβεί σύγκρουση.



Εικόνα 4.3, απεικόνιση του αλγόριθμου οπισθοχώρησης

Το μέγεθος των χρονοθυρίδων εξαρτάται από το φυσικό επίπεδο και είναι ίσο με το διάστημα που χρειάζεται οποιοσδήποτε σταθμός για να εντοπίσει την μετάδοση από οποιοδήποτε άλλο σταθμό στο μέσο. Για υλοποιήσεις με φυσικό επίπεδο FHSS είναι 28μsec και για DSSS 10μsec.

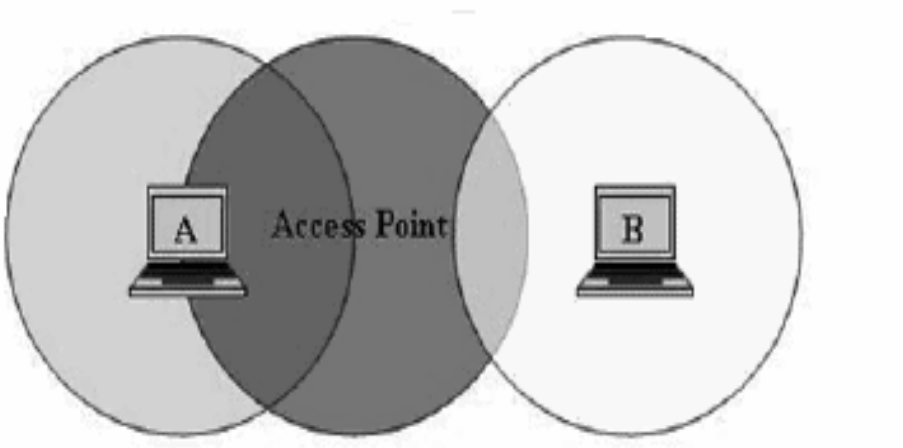
4.2 Τρόπος λειτουργίας CSMA/CA

Όταν ένας σταθμός λάβει ένα πακέτο περιμένει για χρονικό διάστημα ίσο με SIFS και απαντά στον αποστολέα με ένα πλαίσιο επιβεβαίωσης το Acknowledgement (ACK). Έτσι ο σταθμός που επιβεβαιώνει ένα πλαίσιο θα περιμένει λιγότερο από τους άλλους που περιμένουν να στείλουν. Με αυτό τον τρόπο ευνοείται ο πρώτος όσο αφορά την απόκτηση του μέσου. Αυτή η μορφή επιβεβαίωσης είναι και ο μόνος τρόπος να εξασφαλιστεί η αξιοπιστία της επικοινωνίας και να λυθεί το πρόβλημα των συγκρούσεων. Αυτός ο τρόπος επικοινωνίας ονομάζεται και αλγόριθμος της διπλής χειραψίας.

Το CSMA/CA είναι ένα πρωτόκολλο που στόχο έχει να αποφύγει όσο το δυνατόν περισσότερο τις συγκρούσεις. Αυτή είναι και η διάφορα του από το συγγενικό του πρωτόκολλο CSMA/CD που χρησιμοποιείται στο Ethernet και σκοπό έχει να εντοπίζει και όχι να αποφεύγει τις συγκρούσεις. Αναφορικά , ο λόγος που στα ασύρματα δίκτυα αντί του CSMA/CD χρησιμοποιούμε το CSMA/CA είναι η δυσκολία και το μεγάλο κόστος που έχει η παράγωγη ασύρματων καρτών δικτύου που θα έχουν την δυνατότητα να στέλνουν και ταυτόχρονα να λαμβάνουν σήματα.

Όσο αφόρα την τεχνική CSMA/CA μπορεί να προκύψουν κάποια προβλήματα. Πρώτο είναι η απόσβεση του σήματος. Όσο πιο μακριά βρίσκεται ένας σταθμός από τον προορισμό του πακέτου που στέλνει, το σήμα χάνει μέρος της ισχύς του και αυτό μπορεί να δημιουργήσει πρόβλημα σε σταθμούς που βρίσκονται μακριά και διαγωνίζονται για την απόκτηση του μέσου, γιατί πολύ απλά δεν λαβαν , λόγο απόσβεσης, την ειδοποίηση ότι κάποιος άλλος στέλνει.

Δεύτερο είναι το γνωστό πρόβλημα του κρυμμένου σταθμού. Σε αυτή την περίπτωση φανταστείτε δυο σταθμούς που συνδέονται σε ένα AP και βρίσκονται στις αντίθετες πλευρές των ορίων της εμβέλειας του δικτύου. Όντας στην μεγαλύτερη δυνατή απόσταση στην οποία μπορούν να επικοινωνήσουν με το AP δεν είναι σε θέση να επικοινωνήσουν μεταξύ τους. Έτσι δεν μπορούν να αντιληφθούν αν ο απέναντι στέλνει κάτι και σε περίπτωση που θέλουν να εκπέμψουν την ιδία ώρα θα συμβεί σύγκρουση.



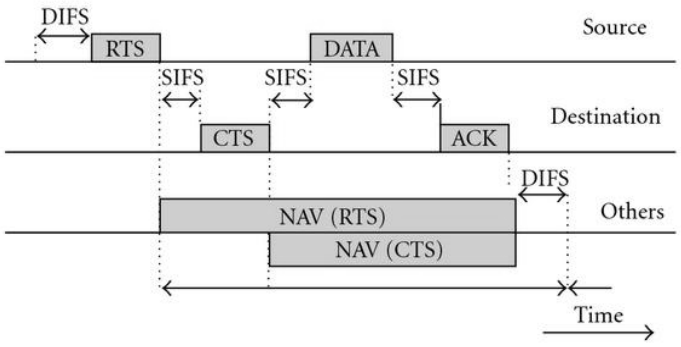
Εικόνα 4.4, παράδειγμα με το πρόβλημα του κρυμμένου σταθμού.

Για να λυθεί αυτό το πρόβλημα το CSMA/CA συμπληρώνεται με την τεχνική RTS/CTS.

4.3 RTS/CTS

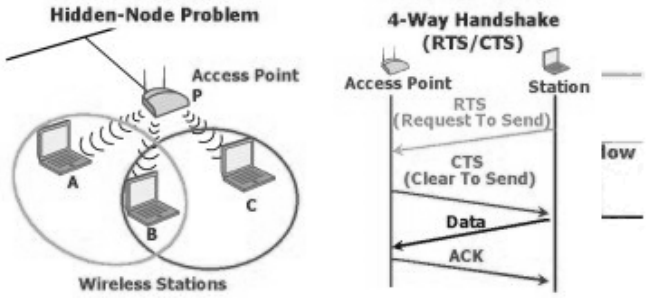
Ο αλγόριθμος αυτός είναι ένας τρόπος να αναβαθμίσουμε τον προηγούμενο αλγόριθμο, από διπλή χειραψία σε τετραπλή χειραψία.

Όταν ένας κόμβος θέλει να στείλει δεδομένα ειδοποιεί τον παραλήπτη αποστέλλοντας ένα frame, το RTS (Request to Send). Αν ο παραλήπτης είναι είναι έτοιμος να δεχτεί απάντα με ένα frame, το CTS (Clear to Send), μετά από χρονικό διάστημα ίσο με SIFS. Ο αποστολέας μπορεί να στείλει και αυτός έπειτα από SIFS. Όποιος άλλος σταθμός λάβει είτε RTS είτε CTS frame απέχει από την αποστολή δεδομένων. Ο χρόνος για τον οποίο τίθεται σε αναμονή περιλαμβάνεται μέσα στα RTS/CTS frames. Τα πακέτα RTS και CTS είναι πολύ μικρά (20 και 14 byte) σε σχέση με το μέγιστο πλαίσιο δεδομένων του 802.11 (2346 byte).



Εικόνα 4.5, ακολουθία πακέτων RTS/CTS και data.

Η χρήση του RTS/CTS σε περιβάλλοντα που χαρακτηρίζονται από μικρά πακέτα δεδομένων και μέσο φορτίο, προσθέτει καθυστέρηση στο δίκτυο λόγο της συνολικής επιβάρυνσης που προκαλεί η λειτουργία του.



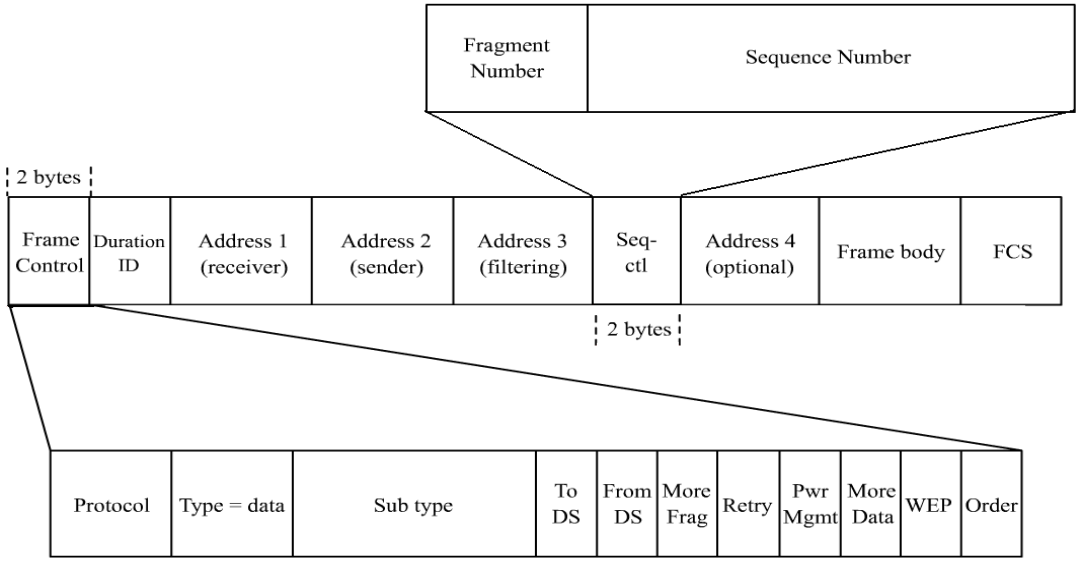
Εικόνα 4.6, Απεικόνιση του προβλήματος του κρυμμένου σταθμού και της τετραπλής χειραψίας

Κάποια πράγματα για τα χρονικά διαστήματα IFS (Inter Frame Space) όπως αυτά φαίνονται στις εικόνες:

1. **Short IFS (SIFS) :** είναι η περίοδος μεταξύ της ολοκλήρωσης της αποστολής ενός πακέτου και της αρχής της μετάδοσης του αντίστοιχου ACK.
2. **Point Coordination IFS (PIFS) :** είναι ένα SIFS συν τον χρόνο μιας χρονοθυρίδας.
3. **Distributed IFS (DIFS) :** είναι ένα PIFS συν τον χρόνο μιας χρονοθυρίδας.

4.4 Πλαίσιο του MAC

Το πλαίσιο που χρησιμοποιεί το επίπεδο MAC έχει την μορφή της παρακάτω εικόνας.



Εικόνα 4.7, το πλαίσιο του MAC

Duration/ID: Καθορίζει τον χρόνο ανάθεσης του καναλιού.

Address Fields: Υπάρχουν 4 πεδία διευθύνσεων των 48 bit, τα οποία δεν χρησιμοποιούνται πάντα. Οι διαφορετικοί τύποι των διευθύνσεων είναι οι εξής:

1. Διεύθυνση προορισμού (Destination address).
2. Διεύθυνση αποστολέα (Source address).
3. Διεύθυνση παραλήπτη (Receiver address).
4. Διεύθυνση μεταδότη (Transmitter address).
5. Basic Service Set ID (BSSID), αναλύεται σε επόμενο κεφάλαιο.

Η address 1 χρησιμοποιείται για τον παραλήπτη (Receiver), η address 2 χρησιμοποιείται για τον μεταδότη (Transmitter), η address 3 χρησιμοποιείται για να φιλτράρει από τον παραλήπτη πακέτα τα οποία απορρίπτονται από ένα BSS διαφορετικό του συσχετιζόμενου.

Sequence control: Χρησιμοποιείται για την αρίθμηση και επανασυναρμολόγηση (de-fragmentation) των κατακερματισμένων πλαισίων καθώς και για την απόρριψη των διπλότυπων πλαισίων. Περιλαμβάνει:

1. Ένα πεδίο των 4 bit που αντιστοιχεί στον αριθμό κατακερματισμού(fragment number). Έχει την τιμή «0» για το πρώτο fragment και αυξάνεται κατά ένα για κάθε επόμενο.
2. Ένα πεδίο των 12-bit που αντιστοιχεί στον αριθμό ακολουθίας(sequence number). Ξεκινά με την τιμή «0» και αυξάνεται κατά ένα για κάθε πακέτο, ανώτερου επιπέδου, που χειρίζεται το MAC. Πακέτα τα οποία μεταδίδονται για δεύτερη φορά διατηρούν τον ίδιο αριθμό ακολουθίας.

Frame body (also Data Field): Είναι τα δεδομένα που μεταφέρει το MAC πλαίσιο, με μέγιστη τιμή τα 2296 bytes. Το ονομάζουμε και MSDU.

Frame Check Sequences (FCS): Χρησιμοποιείται για να ελέγξει την ακεραιότητα του πλαισίου που παραλήφθηκε.

Frame control: Το πεδίο FC περιέχει 11 υποπεδία τα οποία είναι τα εξής:

1. **Protocol version** (2 bit) : υποδεικνύει σε ποια έκδοση του 802.11 MAC είναι το υπόλοιπο πλαίσιο. Επί του παρόντος υπάρχει μόνο μια έκδοση και η προκαθορισμένη τιμή της είναι το «0».

2. **Type** (2 bit), **Subtype** (4 bit) : αυτά τα δυο πεδία προσδιορίζουν τον τύπο του πλαισίου. Το πεδίο Type διαχωρίζει αν το πλαίσιο θα είναι πλαίσιο έλεγχου, διαχείρισης ή δεδομένων. Το Subtype προσδιορίζει την ακριβή λειτουργία του. Πχ αν το type έχει την τιμή «01» και το subtype την τιμή «1011» τότε μιλάμε για ένα RTS frame.
3. **ToDS** (1 bit), **FromDS** (1 bit) : καθορίζουν αν το πλαίσιο προορίζεται για ένα σύστημα διανομής (disrtibution system, DS).
4. **More fragments** (1 bit) : αν πάρει την τιμή «1» τότε η πληροφορία που έρχεται από τα πάνω επίπεδα έχει κατακερματιστεί.
5. **Retry** (1 bit) : 1 αν πρόκειται για επαναμετάδοση προηγούμενου πλαισίου τότε το πεδίο αυτό παίρνει την τιμή «1».
6. **Power management** (1 bit) : για να εξοικονομήσουν ενέργεια πολλές συσκευές έχουν την δυνατότητα να κλείνουν τμήματα του interface του δικτύου τους. Αν το πεδίο αυτό πάρει την τιμή «1» σημαίνει ότι ο αποστολέας θα μπει σε sleep mode.
7. **More data** (1 bit) : Υποδεικνύει πως ο σταθμός έχει και άλλα δεδομένα να στείλει.
8. **WEP** (1 bit) : αν το πλαίσιο είναι κρυπτογραφημένο με την μέθοδο WEP τότε το πεδίο παίρνει την τιμή «1».
9. **Order** (1 bit) : αν η πληροφορία είναι κατακερματισμένη και θέλουμε τα πλαίσια να σταλούν με σειρά το πεδίο αυτό παίρνει την τιμή «1».

4.4 Ασφάλεια

Το πρότυπο 802.11 περιλαμβάνει δυο μεθόδους ασφάλειας. Η πρώτη είναι ο αλγόριθμος WEP ο οποίος επιτρέπει την κρυπτογράφηση των πλαισίων κατά την μετάδοση τους. Η διαδικασία είναι η εξής:

1. Ο αποστολέας παράγει ένα αριθμό, των 32 bit, που σχετίζεται με την ακεραιότητα του ωφελίμου φορτίου του πλαισίου MAC.
2. Χρησιμοποιείται ένα κοινόχρηστο κλειδί κρυπτογράφησης, ως είσοδος σε μια γεννήτρια τυχαίων αριθμών, και παράγεται μια τυχαία ακολουθία bit που έχει μήκος ίσο με το άθροισμα των μηκών του ωφέλιμου φορτίου MAC

και της τιμής ακεραιότητας. Έπειτα τα πεδία αυτά κρυπτογραφούνται με την χρήση ενός δυαδικού πολλαπλασιασμού με την ακολουθία bit που είχε παραχθεί προηγουμένως.

3. Το κρυπτογραφημένο ωφέλιμο φορτίο MAC του σταθμού αποστολέα τοποθετείται σε ένα πλαίσιο MAC και το παραδίδεται στο φυσικό επίπεδο για μετάδοση.
4. Ο σταθμός παραλήπτης κάνοντας χρήση του αλγορίθμου WEP χρησιμοποιεί το ίδιο κλειδί για να αποκρυπτογραφήσει το ωφέλιμο φορτίο MAC και υπολογίζει μια τιμή ακεραιότητας για αυτό το φορτίο. Αν η τιμή είναι ίδια με αυτή που έχει σταλεί μαζί με το πλαίσιο, το ωφέλιμο φορτίο διαβιβάζεται στο επίπεδο LLC .

Η δεύτερη μέθοδος ασφάλειας αφορά την πιστοποίηση μεταξύ δυο σταθμών που επικοινωνούν. Υπάρχουν δυο διαδικασίες πιστοποίησης, η πιστοποίηση ανοικτού συστήματος και η πιστοποίηση κοινόχρηστου κλειδιού.

Η πρώτη διαδικασία αποτελείται από ένα μηχανισμό διπλών χειραψιών και χρησιμοποιείται σε περιπτώσεις που δεν απαιτείται υψηλό επίπεδο ασφάλειας. Σύμφωνα με αυτή την διαδικασία ένας σταθμός αναγγέλλει της επιθυμία του να επικοινωνήσει με έναν άλλο σταθμό ή με ένα AP, εκπέμποντας ένα πλαίσιο πιστοποίησης. Ο σταθμός παραλήπτης αποκρίνεται με ένα άλλο πλαίσιο που προσδιορίζει την επιτυχία ή την αποτυχία της πιστοποίησης.

Η δεύτερη διαδικασία είναι ένας μηχανισμός τετραπλής χειραψίας που χρησιμοποιεί τον αλγόριθμο WEP και περιλαμβάνει τα εξής βήματα:

1. Ο σταθμός που θέλει να εκπέμψει στέλνει σε έναν άλλο σταθμό ένα πλαίσιο πιστοποίησης.
2. Με την παραλαβή του πλαισίου πιστοποίησης ο σταθμός απαντά με την αποστολή ενός άλλου πλαισίου πιστοποίησης που περιέχει μια ακολουθία των 128 byte.
3. Ο αιτούμενος σταθμός κρυπτογραφεί τη ληφθείσα ακολουθία χρησιμοποιώντας τον αλγόριθμο WEP και την στέλνει στον άλλο σταθμό.

4. Η ληφθείσα ακολουθία αποκρυπτογραφείται στο σταθμό παραλήπτη. Αν η αποκρυπτογραφημένη ακολουθία ταιριάζει με εκείνη που έφτασε στον αιτούμενο σταθμό, ο τελευταίος ενημερώνεται για την επιτυχή πιστοποίηση.

4.5 Εξοικονόμηση Ενέργειας

Το πρότυπο 802.11 μπορεί να υποστηρίξει εξοικονόμηση ενέργειας. Αυτό το πετυχαίνει με την προσωρινή αποθήκευση της κυκλοφορίας στους σταθμούς μεταδότες. Όταν ένας κινητός κόμβος βρίσκεται σε «κατάσταση ύπνου» (sleep mode), μη μπορώντας έτσι να λάβει οποιαδήποτε πληροφορία, όλη η κυκλοφορία που προορίζεται για αυτόν αποθηκεύεται στους σταθμούς μεταδότες προσωρινά μέχρι να «ξυπνήσει». Σε ένα ασύρματο δίκτυο υποδομής, οι κινητοί κόμβοι ξυπνούν περιοδικά για να «ακούσουν» τα πλαίσια σηματοδοσίας που στέλνονται από το AP. Ένας σταθμός που λαμβάνει ένα πλαίσιο σηματοδοσίας το οποίο υποδεικνύει ότι το AP έχει αποθηκευμένα στοιχεία για αυτόν , βγαίνει από την κατάσταση ύπνου και ζητά την παραλαβή των δεδομένων. Στα αδόμητα δίκτυα οι σταθμοί που εφαρμόζουν εξοικονόμηση ενέργειας ξυπνούν περιοδικά για να αφουγκραστούν για εισερχόμενα πλαίσια.

ΚΕΦΑΛΑΙΟ 5^{Ο}

Τοπολογίες Ασύρματων Δικτύων

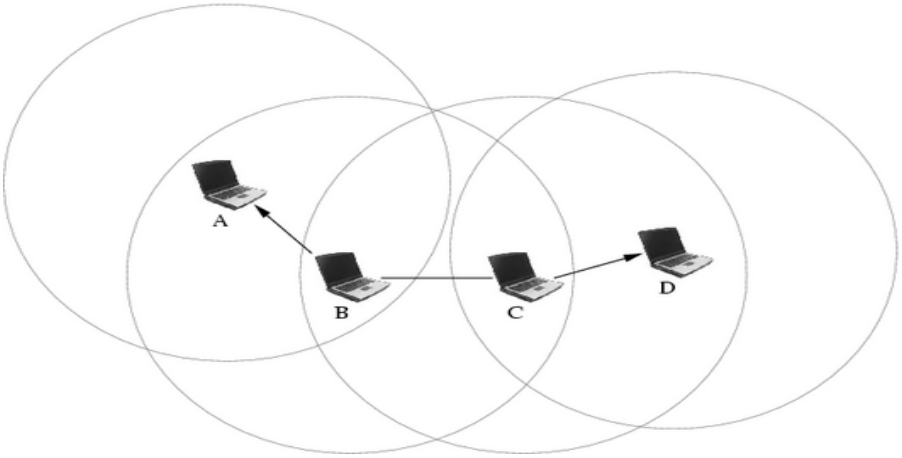
Καλό θα είναι σε αυτό το σημείο να αναφερθούμε και στις τοπολογίες των WLAN. Επιγραμματικά αυτές είναι οι AD-HOC, BSS, ESS. Ακολουθεί μια περιγραφή του τρόπου λειτουργίας της κάθε μιας ξεχωριστά.

5.1 Ad-Hoc

Το πρότυπο IEEE 802.11 αναφέρει την τοπολογία ad-hoc ως IBSS (Independent Basic Service Set). Το IBSS είναι μια Peer-to-Peer τοπολογία ασύρματης δικτύωσης.

Ένα ασύρματο ad-hoc δίκτυο είναι ένα αποκεντρωμένο ασύρματο δίκτυο, δεν υπάρχει δηλαδή κεντρικός σταθμός AP. Κάθε κόμβος είναι πρόθυμος να διαβιβάσει δεδομένα για άλλους κόμβους.

Οι ασύρματοι κόμβοι επικοινωνούν απευθείας ένας προς έναν (peer to peer) και έτσι δυο ή περισσότεροι κόμβοι επικοινωνούν μεταξύ τους. Το δίκτυο απαρτίζεται από μονοπάτια όπου κάθε κόμβος θεωρείται ομότιμος.



Εικόνα 5.1, απλό παράδειγμα ad-hoc δικτύου.

Ένας βασικός περιορισμός ώστε να πραγματοποιηθεί η επικοινωνία μεταξύ 2 κόμβων είναι να βρίσκεται ο ένας εντός της εμβέλειας του άλλου.

Ο καθορισμός των κόμβων που διαβιβάζουν τα δεδομένα γίνεται δυναμικά και με βάση την ικανότητα σύνδεσης του δικτύου. Αυτό έρχεται σε αντίθεση με τα ενσύρματα δίκτυα στα οποία οι routers αναλαμβάνουν την δρομολόγηση των πακέτων. Έρχεται επίσης, σε αντίθεση με τις τοπολογίες ασύρματων δικτύων στις οποίες ένας ειδικός κόμβος, γνωστός ως σημείο πρόσβασης (AP), διαχειρίζεται την επικοινωνία μεταξύ των άλλων κόμβων (wireless managed networks).

Τα πρώτα ad-hoc ασύρματα δίκτυα ήταν τα «packet radio» δίκτυα (PRNETs) τη δεκαετία του 1970, χρηματοδοτούμενα από την DARPA.

Η αποκεντρωμένη φύση των ad-hoc ασύρματων δικτύων τα καθιστά κατάλληλα για ποικίλες εφαρμογές, στις οποίες δεν μπορούμε να βασιστούμε σε AP, και μπορεί να βελτιώσει την επεκτασιμότητα των ασύρματων ad-hoc δικτύων σε σύγκριση με τα wireless managed networks.

Ο ελάχιστος χρόνος διαμόρφωσης (configuration time) καθώς και η γρήγορη εγκατάσταση τους κάνει τα ad-hoc δίκτυα κατάλληλα για περιπτώσεις που δεν προϋπάρχει ασύρματη υποδομή, για κάλυψη μικρών περιοχών όπου δεν είναι απαραίτητη η ύπαρξη ασύρματης υποδομής αλλά και για καταστάσεις έκτακτης ανάγκης όπως φυσικές καταστροφές ή στρατιωτικές επιχειρήσεις.

Στα περισσότερα ασύρματα δίκτυα ad-hoc οι κόμβοι ανταγωνίζονται για την πρόσβαση στο κοινόχρηστο ασύρματο μέσο και συχνά οδηγούνται σε συγκρούσεις.

Για να αποφευχθεί αυτό μία συσκευή που θέλει να εκπέμψει, θα ελέγξει πρώτα αν το μέσο είναι ελεύθερο. Αν είναι ελεύθερο τότε στέλνει τα πακέτα της πληροφορίας μαζί με την διεύθυνση του παραλήπτη. Αν είναι καταλυμένη περιμένει για κάποιο χρονικό διάστημα να ελευθερωθεί.

Την πληροφορία την λαμβάνουν όλοι οι κόμβοι, όμως μόνο ο παραλήπτης την επεξεργάζεται. Οι υπόλοιποι απλά απορρίπτουν τα πακέτα μόλις δουν ότι δεν προορίζονται για αυτούς.

Η περιοχή κάλυψης του δικτύου ονομάζεται Basic Service Area (BSA).

Το σημαντικότερο πλεονέκτημα ενός ad-hoc δικτύου είναι η αξιοπιστία του καθώς μπορεί να προσφέρει πολλά εναλλακτικά μονοπάτια επικοινωνίας μεταξύ 2 κόμβων, αλλά και αυξημένη ταχύτητα.

5.2 BSS

Το Basic Service Set (BSS) είναι το βασικό στοιχείο για την οικοδόμηση ενός IEEE 802.11 ασύρματου τοπικού δικτύου (σύμφωνα με το IEEE 802,11-1999 πρότυπο).

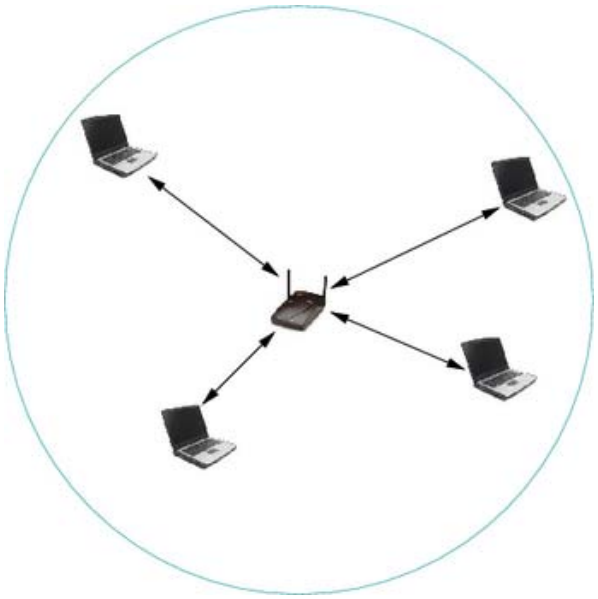
Πρόκειται για μια τοπολογία παρόμοια με το κυψελωτό δίκτυο της κινητής τηλεφωνίας. Σε αυτή την τοπολογία υπάρχει ένας κεντρικός κόμβος , το AP, και σταθμοί που συνδέονται σε αυτόν.

Το AP είναι υπεύθυνο για την διαχείριση των παραμέτρων της ασύρματης επικοινωνίας αλλά και για τον έλεγχο των σταθμών εντός του BSS.

Σε μια τέτοια τοπολογία για να υπάρξει επιτυχής επικοινωνία αρκεί όλοι οι σταθμοί να είναι εντός της εμβέλειας του AP. Δηλαδή για να υπάρξει επικοινωνία μεταξύ 2 σταθμών δεν είναι απαραίτητο να βρίσκεται ο ένας εντός της εμβέλειας του άλλου. Όλα τα μελή του δικτύου επικοινωνούν μεταξύ τους, σύμφωνα με το πρότυπο client-server, μέσω του AP ή αλλιώς «κεντρικού διανομέα».

Κάθε AP έχει ένα όνομα για να ξεχωρίζει από άλλα AP που ενδεχομένως να βρίσκονται στον ίδιο χώρο. Το όνομα αυτό είναι το Service Set identifier ή (SSID). Το SSID είναι αυτό που πρέπει να γνωρίζουμε για να συνδεθούμε σε ένα AP. Από προεπιλογή είναι μέρος της κεφαλίδας κάθε πακέτου που αποστέλλεται μέσω του WLAN.

Είναι πιθανό πολλά AP να έχουν το ίδιο SSID αν παρέχουν πρόσβαση στο ίδιο δίκτυο. Πρόκειται για την τοπολογία ESS που θα εξετάσουμε παρακάτω.



Εικόνα 5.2, απλό παράδειγμα BSS

Κάθε BSS με την σειρά του διαχωρίζεται από τα υπόλοιπα σύμφωνα με ένα μοναδικό χαρακτηριστικό του, το Basic Service Set Identifier (BSSID), σε αντίθεση με το SSID που μπορεί να χρησιμοποιηθεί σε πολλαπλά και πιθανώς επικαλυπτόμενα BSS. Σε μια τοπολογία BSS το BSSID είναι η MAC address του AP που εξυπηρετεί το ασύρματο δίκτυο.

Αξίζει να αναφέρουμε ότι η βασική μορφή ενός BSS είναι αυτή που 2 κόμβοι επικοινωνούν μεταξύ τους, δηλαδή μια τοπολογία IBSS.

5.3 ESS

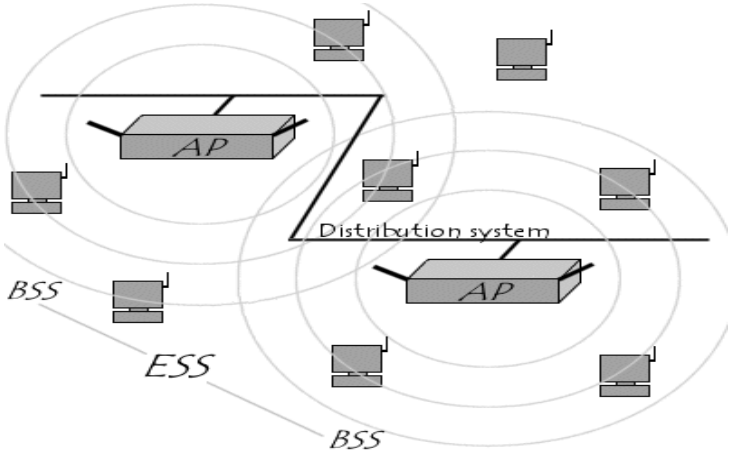
Όταν ένα BSS δεν μπορεί να παράσχει επαρκής κάλυψη μπορούμε να ενώσουμε δυο ή περισσότερα μέσω ενός κοινού συστήματος διανομής (common distribution system). Μια τέτοια δομή καλείται ESS.

Το κοινό σύστημα διανομής μπορεί να είναι μια ενσύρματη σύνδεση των AP ή μπορεί να επιτευχτεί με την χρήση συγκεκριμένων κόμβων που θα κάνουν δυνατή την ασύρματη σύνδεση των επιμέρους δικτύων.

Τα BSS που απαρτίζουν το ESS διαχωρίζονται με βάση το BSSID τους.

Το κοινό σύστημα διανομής επιτρέπει σε πολλά AP ενός ESS να εμφανίζονται πως ανήκουν στο ίδιο BSS. Αυτό επιτυγχάνεται με την χρήση του ίδιου SSID σε όλα τα AP του ESS.

Το δεύτερο πλεονέκτημα αυτής της τοπολογίας είναι η υπηρεσία roaming που παρέχεται στους σταθμούς. Δηλαδή η δυνατότητα ενός σταθμού να μετακινείται ελεύθερα αλλάζοντας BSSs χωρίς να χρειάζεται να του κάνουμε ρυθμίσεις ή να καταλαβαίνει διαφορές ούτε ο σταθμός αλλά ούτε και το δίκτυο.



Εικόνα 5.3, απλό παράδειγμα ESS

Τα BSS επικοινωνούν μεταξύ τους για την μεταφορά πακέτων αλλά και για την αμοιβαία πληροφόρηση τους για τις τυχών εναλλαγές των BSSs που πραγματοποιούν οι σταθμοί.

5.4 Υπηρεσίες του Συστήματος Διανομής

Άλλες υπηρεσίες που προσφέρει το σύστημα διανομής και υλοποιούνται μόνο στα AP είναι οι:

1. Association
2. Disassociation
3. Distribution
4. Integration
5. Reassociation

Οι υπηρεσίες αυτές υλοποιούνται με λογισμικό και όχι με επιπλέον υλικό και έτσι η μεγάλη διαφορά κόστους που συνήθως υπάρχει ανάμεσα στις αντίστοιχες συσκευές δεν δικαιολογείται από την πλευρά του πραγματικού κόστους τους.

Association

Υπηρεσία με την οποία δημιουργείται μία λογική σύνδεση μεταξύ ενός ασύρματου σταθμού και ενός AP. Κάθε σταθμός σχετίζεται με ένα AP, πριν του επιτραπεί να στείλει δεδομένα. Ο ασύρματος σταθμός επικαλείται την υπηρεσία αυτή μόνο μία φορά κατά την είσοδο του στο BSS. Κάθε σταθμός σχετίζεται με μόνο ένα AP και ένα AP μπορεί να σχετιστεί με πολλούς σταθμούς.

Disassociation

Υπηρεσία που σκοπό έχει να επιβάλλει σε σταθμό να εγκαταλείψει μία συσχέτιση με ένα AP ή για ένα σταθμό να ενημερώσει το AP ότι δεν χρειάζεται πλέον τις υπηρεσίες του. Όταν ένας σταθμός αποσυσχετιστεί, πρέπει να ξεκινήσει μία καινούργια συσχέτιση με ένα AP. Ένα AP μπορεί να αναγκάσει ένα ή περισσότερους σταθμούς να απομακρυνθούν, λόγω περιορισμένων πόρων ή γιατί το AP απομακρύνεται από το δίκτυο. Όταν ο σταθμός ενημερωθεί ότι δεν θα έχει πλέον τις υπηρεσίες ενός AP, μπορεί να επικαλεστεί την υπηρεσία αποσυσχέτισης ώστε να ειδοποιήσει το AP ότι η λογική σύνδεση μεταξύ τους δεν απαιτείται πλέον. Οι σταθμοί πρέπει να αποσυσχετίζονται όταν αφήνουν το δίκτυο. Η αποσυσχέτιση έχει τη μορφή ειδοποίησης και μπορεί να σταλεί από οποιοδήποτε από τα συσχετιζόμενα μέρη και κανένα από τα δύο δεν μπορεί να την αρνηθεί.

Re-association

Η επανασυσχέτιση επιτρέπει σε ένα σταθμό να αλλάξει τη τρέχουσα συσχέτιση του με ένα AP. Είναι παρόμοια υπηρεσία με τη συσχέτιση με τη διαφορά ότι περιέχει πληροφορία για το AP στο οποίο ο σταθμός ήταν πριν συσχετισμένος. Ένας σταθμός χρησιμοποιεί την υπηρεσία αυτή καθώς μετακινείται διαρκώς σε ένα ESS δίκτυο, χάνει την επαφή με το AP με το οποίο είχε συσχετιστεί αρχικά και χρειάζεται να συσχετιστεί με κάποιο καινούργιο. Με την υπηρεσία αυτή. Στέλνοντας πληροφορία για το προηγούμενο AP με το οποίο είχε συσχετιστεί, το καινούργιο AP μπορεί να επικοινωνήσει με το προηγούμενο και να αποκτήσει τα

πακέτα τα οποία μπορεί να έχουν παραμείνει εκεί προς παράδοση στον σταθμό. Η υπηρεσία επανασυσχέτισης αρχικοποιείτε πάντα από τον σταθμό.

Distribution

Η διανομή είναι βασική υπηρεσία η οποία παρέχεται από έναν 802.11 σταθμό. Ο σταθμός χρησιμοποιεί την υπηρεσία κάθε φορά που στέλνει ένα MAC πλαίσιο προς το σύστημα διανομής. Το σύστημα διανομής έπειτα αναλαμβάνει τη διανομή του χρησιμοποιώντας την πληροφορία που έχει αποκτήσει με τις υπηρεσίες συσχέτισης. Ο σταθμός πρέπει να έχει συσχετιστεί με ένα AP ώστε να γίνει η προώθηση των πλαισίων σωστά.

Integration

Η υπηρεσία αυτή συνδέει ένα δίκτυο 802.11 WLAN σε άλλα LANs ενσύρματα ή ασύρματα. Ένα portal είναι αυτό που υλοποιεί την υπηρεσία αυτή. Τυπικά βρίσκεται σε ένα AP, μπορεί όμως και να είναι τμήμα ενός διαφορετικού δικτύου. Η υπηρεσία αυτή μεταφράζει πλαίσιο 802.11 σε πλαίσια που μπορούν να μεταδοθούν σε άλλο δίκτυο και το ανάστροφο.

Roaming

Τώρα που γνωρίζουμε τις διαδικασίες συσχέτισης μπορούμε να κατανοήσουμε καλύτερα την υπηρεσία roaming.

Όταν ένας σταθμός βρεθεί εντός εμβέλειας ενός ή περισσοτέρων AP, διαλέγει εκείνο το AP το οποίο έχει καλύτερο σήμα ή μικρότερο αριθμό λαθών. Η διαδικασία αυτή λέγεται Joining a Basic Service Set. Όταν γίνει αποδεκτή η συσχέτιση από το AP, ο σταθμός συντονίζεται στο κανάλι εκπομπής του AP. Περιοδικά γίνεται ανίχνευσή των καναλιών και στην περίπτωση που βρεθεί κανάλι με καλύτερα χαρακτηριστικά, γίνεται επανασυσχέτιση με το καινούργιο AP και συντονισμός του σταθμού στην καινούρια συχνότητα. Η επανασυσχέτιση μπορεί να γίνει λόγω φυσικής μετακίνησης του σταθμού ή μπορεί να γίνει σαν αποτέλεσμα υψηλού φόρτου στο δίκτυο. Η λειτουργία αυτή γνωστή ως "load balancing" κατανέμει τον συνολικό φόρτο του WLAN με αποτελεσματικό τρόπο στην ασύρματη δομή του. Αυτός ο δυναμικός τρόπος συσχέτισης επιτρέπει την

διάρθρωση ενός WLAN με πολύ ευρεία κάλυψη απλώς δημιουργώντας μία σειρά από 802.11 κυψέλες. Για να πετύχει μια τέτοια σχεδίαση πρέπει οι κυψέλες να σχεδιαστούν σωστά, δηλαδή να γίνει επιλογή της τοποθεσίας, της συχνότητας, των κεραιών.

5.5 Υπηρεσίες που υλοποιούνται στους Ασύρματους Σταθμούς

Επιπλέον υπηρεσίες που υλοποιούνται σε κάθε ασύρματο σταθμό είναι οι εξής:

1. Authentication
2. Deauthentication
3. Privacy
4. Data Delivery

Authentication

Ορίζονται διαδικασίες αυθεντικοποίησης ώστε να ελεγχθεί η πρόσβαση στο WLAN. Ο σκοπός της αυθεντικοποίησης είναι να παρέχει έλεγχο πρόσβασης όμοιο με αυτόν στα ενσύρματα LAN. Παρέχει ένα μηχανισμό για ένα σταθμό να προσδιορίζει άλλον. Χωρίς απόδειξη της ταυτότητας του ένας σταθμός δεν επιτρέπεται να χρησιμοποιεί το WLAN. Υπάρχουν δύο τύποι αυθεντικοποίησης:

1. Open system authentication

Είναι ο εξ' ορισμού τρόπος, είναι πολύ απλός και έχει δύο βήματα. Πρώτα ο σταθμός που θέλει να κάνει την αυθεντικοποίηση στέλνει ένα πλαίσιο αυθεντικοποίησης το οποίο περιέχει την ταυτότητα του. Ο άλλος σταθμός στέλνει πίσω ένα πλαίσιο που περιέχει την πληροφορία αναγνώρισης ή μη της ταυτότητας του αποστολέα.

2. Shared key authentication

Ο κάθε σταθμός έχει λάβει ένα κρυφό κλειδί, μέσω ενός καναλιού το οποίο είναι ανεξάρτητο του 802.11 δικτύου. Οι σταθμοί κάνουν αυθεντικοποίηση μέσω της κοινής γνώσης του κρυφού κλειδιού. Η υλοποίηση αυτή απαιτεί την κρυπτογράφηση μέσω αλγορίθμου WEP, Wired Equivalent Privacy.

De-authentication

Η υπηρεσία αυτή αφορά την απομάκρυνση ενός σταθμού που είχε προηγούμενα αυθεντικοποιηθεί από το δίκτυο. Το μήνυμα από-αυθεντικοποίησης έχει το νόημα ειδοποίησης και δεν μπορεί να απορριφθεί. Το αντίστοιχο πλαίσιο μπορεί να σταλεί από ένα σταθμό ή από το AP.

Privacy

Η υπηρεσία αυτή είναι προαιρετική. Όλα τα δεδομένα που στέλνονται και λαμβάνονται μεταξύ του AP και των συσχετιζόμενων σταθμών του, κωδικοποιούνται με κάποιο κλειδί.

Η υπηρεσία αυτή έχει σκοπό να παρέχει ένα ισοδύναμο επίπεδο προστασίας με αυτό που παρέχεται στα ενσύρματα δίκτυα, όπου η φυσική πρόσβαση είναι περιορισμένη. Παρέχει προστασία στα δεδομένα στο κομμάτι της διαδρομής τους στο ασύρματο μέσο. Δεν παρέχει πλήρη προστασία από άκρο σε άκρο μεταξύ εφαρμογών που λειτουργούν σε ένα μικτό δίκτυο. Στο ασύρματο δίκτυο όλοι ο σταθμοί καθώς και άλλες συσκευές μπορούν να αφουγκραστούν τα δεδομένα που ανταλλάσσονται, και έτσι να θέσουν σημαντικά προβλήματα ασφαλείας στο δίκτυο.

Data Delivery

Παρόμοια με αυτή που παρέχεται από άλλα δίκτυα IEEE 802. Η υπηρεσία αυτή παρέχει αξιόπιστη μεταφορά των πακέτων δεδομένων από ένα σταθμό στον άλλο. Ο όρος αξιόπιστη μεταφορά σημαίνει ότι θα ζητηθεί επανεκπομπή των πακέτων αν διαπιστωθεί ότι αυτά έχουν λάθη.

Όσοι περισσότεροι σταθμοί είναι συνδεδεμένοι σε ένα AP τόσο πιο πολύ πέφτει η απόδοση του τελευταίου, καθώς το μέσο μετάδοσης είναι κοινό.

Αν σε αυτό συνυπολογίσουμε πλήθος άλλων προβλημάτων , όπως υψηλό επίπεδο θορύβου στην επικοινωνία, μεγάλες αποστάσεις μεταξύ των σταθμών κλπ θα δούμε πως η μείωση της απόδοσης του δικτύου είναι αισθητή.

Για να μπορέσει λοιπόν το AP να διαχειριστεί μεγάλο φόρτο εργασίας μειώνοντας παράλληλα το ποσοστό των σφαλμάτων χρησιμοποιεί την τεχνική του fragmentation στην επικοινωνία.

Ο κατακερματισμός, στα ελληνικά, καθορίζει το μέγεθος των πακέτων που εκπέμπονται. Τα μικρότερα πακέτα έχουν περισσότερες πιθανότητες να φτάσουν ανέπαφα στον προορισμό τους μειώνοντας παράλληλα την συχνότητα λαθών. Αλλά ταυτόχρονα μειώνεται η απόδοση του συστήματος μεταδίδοντας περισσότερο overhead και λιγότερη χρήσιμη πληροφορία.

ΚΕΦΑΛΑΙΟ 6^{Ο}

Πρόγραμμα Προσομοιωσεων NS-2

Το NS-2 είναι ένας προσομοιωτής δικτύων προσανατολισμένος από γεγονότα (discrete event simulator) γραμμένος σε C++ και OTcl.

Λειτουργεί σε επίπεδο πακέτων και παρέχει στήριξη στην προσομοίωση πολλών πρωτοκόλλων (όπως TCP, UDP, FTP, HTTP, routing protocols, multicast protocols) πάνω από ενσύρματα και ασύρματα δίκτυα, τοπικά ή δορυφορικά.

Το NS ξεκίνησε ως παραλλαγή του προσομοιωτή REAL το 1989 και από τότε έχει αναπτυχτεί σημαντικά.

Το NS παρέχει αξιοπιστία ωστόσο δεν είναι ένα πρόγραμμα στο οποίο θα πρέπει κάποιος να βασίζεται ανεπιφύλακτα μιας και είναι ένα προϊόν συνεχούς έρευνας και ανάπτυξης. Διαρκώς ανακαλύπτονται και διορθώνονται bugs του προγράμματος.

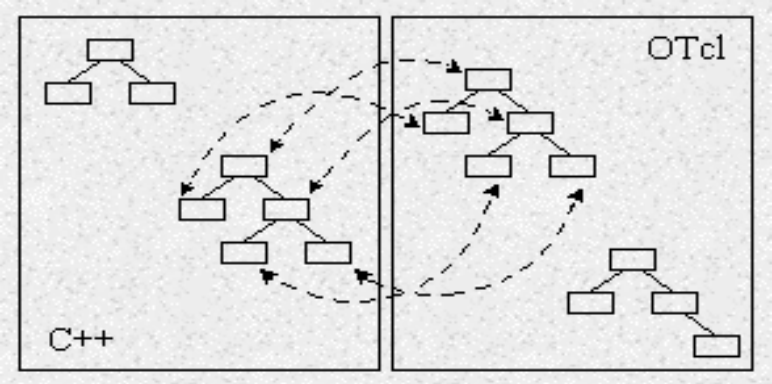
Οι χρήστες του NS είναι υπεύθυνοι να επαληθεύουν για τον εαυτό τους ότι οι προσομοιώσεις τους είναι απαλλαγμένες από σφάλματα.

6.1 Δομή του NS-2

Όπως ήδη αναφέραμε το NS είναι γραμμένο σε C++ και διαθέτει ένα διερμηνέα OTcl. Ο προσομοιωτής υποστηρίζει μια ιεραρχία κλάσεων C++ (compiled hierarchy) και μια παρόμοια OTcl (interpreted hierarchy). Οι 2 αυτές ιεραρχίες σχετίζονται στενά και στα μάτια του χρήστη φαίνεται να υπάρχει μια 1-1 συσχέτιση.

Η ρίζα αυτής της ιεραρχίας είναι η κλάση TclObject. Οι χρήστες δημιουργούν τα αντικείμενα της προσομοίωσης μέσω του διερμηνέα της OTcl ο οποίος τα αρχικοποιεί συνδέοντας τα με τα αντίστοιχα αντικείμενα της compiled hierarchy.

Το NS χρησιμοποιεί δυο γλώσσες προγραμματισμού γιατί ο προσομοιωτής έχει δυο διαφορετικά πράγματα να κάνει.



Εικόνα 6.1, η διυκότητα του NS-2

Από την μια η λεπτομερής προσομοίωση των πρωτοκόλλων επικοινωνίας απαιτεί μια γλώσσα που μπορεί να διαχειριστεί τα bytes και τα headers των πακέτων αλλά και να εφαρμόσει αποτελεσματικά αλγορίθμους πάνω σε μεγάλες ποσότητες πληροφορίας. Για τις ανάγκες αυτές είναι σημαντικότερος ο χρόνος εκτέλεσης (run-time speed) έναντι του χρόνου διόρθωσης & επαναπροσδιορισμού της προσομοίωσης (turn-around time), που περιλαμβάνει τις διαδικασίες εύρεσης και διόρθωσης λαθών, επανάληψη της μεταγλώττισης και της εκτέλεσης.

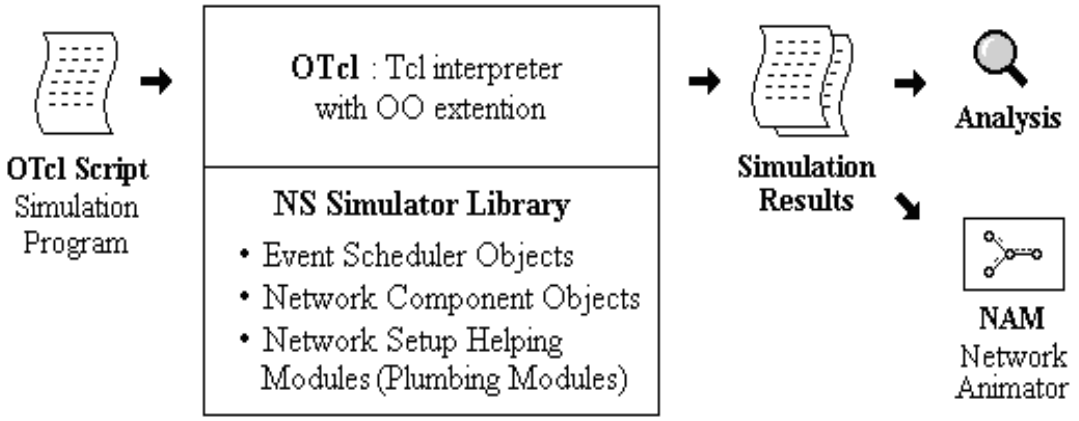
Στον αντίποδα, ένα μεγάλο κομμάτι της έρευνας των δικτύων αποτελεί πάντα η τροποποίηση και μεταβολή των παραμέτρων της προσομοίωσης. Για αυτή την ανάγκη σημαντικότερος είναι ο χρόνος επανάληψης (αλλαγή του μοντέλου και re-run).

Το NS λοιπόν καλύπτει τις ανάγκες του αυτές με την χρήση 2 γλωσσών:

Η C++, η οποία είναι γρήγορη και αποδοτική αλλά δύσκολη για να την τροποποιήσεις ενδείκνυται για την εφαρμογή ενός λεπτομερούς πρωτοκόλλου.

Η OTcl, η οποία είναι πολύ πιο αργή αλλά πολύ πιο ευμετάβλητη είναι ιδανική για τις ρυθμίσεις της προσομοίωσης.

Το NS μέσω του Tclcl(Tcl/C++ interface) μπορεί και εμφανίζει τα αντικείμενα και τις μεταβλητές και στις 2 γλώσσες.



Εικόνα 6.2, απλοποιημένη οπτική του χρήστη.

Βήμα 1: γράφουμε το αρχείο της προσομοίωσης σε OTcl.

Βήμα 2: το script θα το τρέξει ο OTcl interpreter.

Βήμα 3: θα μας επιστρέψει τα αποτελέσματα της προσομοίωσης.

Βήμα 4: μπορούμε να αναλύσουμε τα αποτελέσματα και να κάνουμε γραφικά χρησιμοποιώντας το NAM.

6.2 NS έκδοση 3 (NS-3)

Από τον Ιούλιο του 2006 έχει ξεκινήσει το project του NS-3 και προβλέπεται να διαρκέσει 4 χρόνια. Τελευταία έκδοση είναι η ns-3.4, και κυκλοφόρησε τον Απρίλιο του 2009. Το πρόγραμμα χρηματοδοτείτε από το πανεπιστήμιο της Ουάσινγκτον, από το Georgia Institute of Technology και το κέντρο ερευνών Internet ICSI με την υποστήριξη του Planète research group στο INRIA (Institut national de recherche en informatique et automatique).

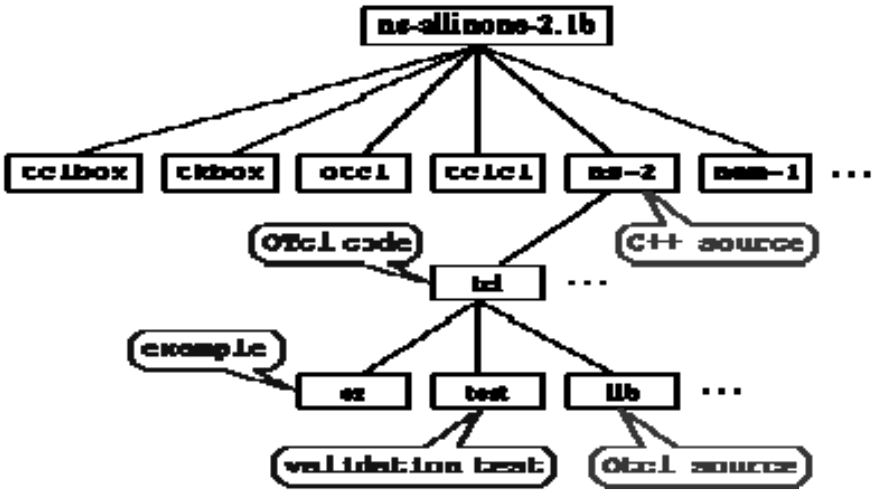
Το NS-3 είναι, όπως και το NS-2, ένας προσομοιωτής δικτύων προσανατολισμένος από γεγονότα (discrete event simulator) συστήματα Internet. Φιλοδόξει να αντικαταστήσει το NS-2 ενώ δεν προβλέπεται να υπάρχει συμβατότητα προς τα πίσω με το NS-2. Επιτρέπει στους ερευνητές να μελετήσουν τα πρωτοκόλλα Internet και μεγάλης κλίμακας συστήματα σε ένα ελεγχόμενο περιβάλλον.

6.3 Εγκατάσταση του NS-2

Πριν προχωρήσουμε περαιτέρω καλό είναι να αναφερθεί που βρίσκουμε το πρόγραμμα NS-2 και πως το κάνουμε εγκατάσταση.

Το NS-2 τρέχει σε αρκετές πλατφόρμες όπως είναι οι : FreeBSD, Linux, SunOS και Solaris. Το NS-2 επίσης μπορεί να εγκατασταθεί και να τρέξει σε περιβάλλον Windows μέσω του CYGWIN.

Απλά σενάρια τρέχουν σε οποιοδήποτε απλό μηχάνημα, ωστόσο πολύ μεγάλα σενάρια απαιτούν την ύπαρξη μεγάλης κεντρικής μνήμης.



Εικόνα 6.3, η δομή των φακέλων εγκατάστασης του NS-2

Υπάρχουν 2 τρόποι για να εγκαταστήσουμε το NS-2:

1. **From the pieces**, εξοικονόμηση χώρου στον δίσκο.
2. **All at once**, πιο γρήγορη εγκατάσταση.

Για να μπορέσει να τρέξει το NS-2 χρειάζεται να εγκαταστήσουμε τα ακόλουθα πακέτα:

1. **Tcl release 8.4.14,** το βρίσκουμε στην διεύθυνση: **http://www.tcl.tk/software/tcltk/downloadnow84.tml**
2. **Tk release 8.4.14,** το βρίσκουμε στην διεύθυνση: **http://www.tcl.tk/software/tcltk/downloadnow84.tml**

3. **OTcl release 1.13** , το βρίσκουμε στην διεύθυνση: **http://sourceforge.net/project/showfiles.php?group_id=30174&package_id=22199&release_id=492753**
4. **TclCL release 119,** το βρίσκουμε στην διεύθυνση: **http://sourceforge.net/project/showfiles.php?group_id=30174&package_id=26142&release_id=492756**
5. **nam-1 release 1.13** (προαιρετικό), το βρίσκουμε στην διεύθυνση: **http://sourceforge.net/project/showfiles.php?group_id=149743&package_id=169585&release_id=492767**
6. **xgraph release 12.1** (προαιρετικό), το βρίσκουμε στην διεύθυνση: **http://www.isi.edu/nsnam/xgraph/**
7. **perl version 5.003 ή και νεότερη** (προαιρετικό), το βρίσκουμε στην διεύθυνση: http://www.perl.com/download.csp
8. **TCL-debug version 1.7** (προαιρετικό, διατίθενται για να βοηθήσει στον εντοπισμό σφαλμάτων), το βρίσκουμε στην διεύθυνση: **http://expect.nist.gov/tcl-debug/**
9. **dmalloc version 4.8.0** (προαιρετικό, debugging μνήμης) , το βρίσκουμε στην διεύθυνση: **http://dmalloc.com/**
10. **sgb2ns conversion program** (προαιρετικό, απαιτείται να γίνει η μετατροπή GT-ITM εξόδου σε μορφή ns-2), το βρίσκουμε στην διεύθυνση: **http://www.isi.edu/nsnam/dist/sgb2ns.tar.gz**
11. **tiers2ns conversion program** (προαιρετικό, που απαιτούνται για να μετατραπούν Βαθμίδες εξόδου σε μορφή ns-2), το βρίσκουμε στην διεύθυνση: **http://www.isi.edu/nsnam/dist/topogen/tiers2ns.awk**
12. **Cweb and sgb source code** (προαιρετικά, απαιτείται να δημιουργηθεί SGB-βιβλιοθήκη που χρησιμοποιείται από τα προγράμματα GT-ITM και το sgb2ns), τα βρίσκουμε στις διευθύνσεις:

    - **ftp://labrea.stanford.edu/pub/cweb/cweb3.4g.tar.gz**
    - **ftp://labrea.stanford.edu/pub/sgb/sgb.tar.gz**

Φυσικά θα χρειαστεί να κατεβάσουμε και το NS-2, το οποίο θα βρούμε στην διεύθυνση:
http://sourceforge.net/project/showfiles.php?group_id=149743&package_id=169584&release_id=588644

6.3.1 Οδηγίες εγκατάστασης From the pieces:

1. Κατεβάζουμε τα πακέτα που χρειαζόμαστε, αυτά που αναφέρθηκαν προηγουμένως.
2. Κάνουμε Unpack τα OTcl, TclCL και NS στον ίδιο φάκελο.
3. Εγκαθιστούμε τα OTcl, TclCL και NS ως εξής:

   - **cd into the OTcl directory**
   - **run ./configure**
   - **run make**

   - **cd into the TclCL directory**
   - **run ./configure**
   - **run make**

   - **cd into the ns directory**
   - **run ./configure**
   - **run make**

4. Πιστοποιούμε ότι εγκαταστάθηκε και λειτουργεί σωστά:

   - **./validate**

Στο βήμα 3 για το Unix, ένα απλό «. / Configure» θα προσπαθήσει να ανιχνεύσει αυτόματα τα πακέτα που χρειάζεται το NS για να οικοδομηθεί. Η αυτόματη ανίχνευση ψάχνει σε λογικά μέρη (όπως / usr / local) και τον κατάλογο πάνω από τον τρέχων direcory. Αν έχετε εγκαταστήσει τα πακέτα αλλού μπορείτε να το πείτε στο NS με επιλογές που ακολουθούν το ./ configure. Το «. / configure –help» δίνει μια πλήρη λίστα των επιλογών.

6.3.2 Οδηγίες εγκατάστασης All at once:

Επί του παρόντος το All at once περιλαμβάνει τα εξής:

1. **Tcl release 8.4.18** (απαιτούμενο στοιχείο)
2. **Tk release 8.4.18** (απαιτούμενο στοιχείο)
3. **Otcl release 1.13** (απαιτούμενο στοιχείο)
4. **TclCL release 1.19** (απαιτούμενο στοιχείο)
5. **Ns release 2.33** (απαιτούμενο στοιχείο)
6. **Nam release 1.13** (προαιρετικό στοιχείο)
7. **Xgraph version 12** (προαιρετικό στοιχείο)
8. **CWeb version 3.4g** (προαιρετικό στοιχείο)
9. **SGB version 1.0 (?)** (προαιρετικό στοιχείο, βασίζεται sgblib για όλες τις πλατφόρμες UNIX τύπου)
10. **Gt-itm gt-itm and sgb2ns 1.1** (προαιρετικό στοιχείο)
11. **Zlib version 1.2.3** (προαιρετική, αλλά να απαιτείται για να χρησιμοποιηθεί το Nam)

Οι οδηγίες που θα ακολουθήσουν είναι αυτές που χρησιμοποίησα και εγώ για τις ανάγκες της παρούσας εργασίας και αναφέρονται για την εγκατάσταση του NS2 στην UNIX πλατφόρμα Ubuntu 7.04.

1. Το πρώτο αναγκαίο είναι η ύπαρξη σύνδεσης internet.
2. Έπειτα πρέπει να κατεβάσουμε και να εγκαταστήσουμε το πρόγραμμα. Οι εντολές είναι οι ακόλουθες και αφορούν την έκδοση 2.31 (Δεν υπάρχει κάποια ουσιαστική διαφορά στην εγκατάσταση άλλων εκδόσεων):

```
$ wget
http://nchc.dl.sourceforge.net/sourceforge/nsnam/ns-
allinone-2.31.tar.gz
$ tar -xzvf ns-allinone-2.31.tar.gz
$ cd ns-allinone-2.31
$ sudo apt-get install build-essential autoconf
automake libxmu-dev
```

3. αφού κάνουμε και αυτό τρέχουμε την επόμενη εντολή:

```
$./install
```

4. Πηγαίνουμε στον φάκελο /etc και πληκτρολογούμε το εξής:

```
$ gedit ~/.bashrc
```

5. και στο τέλος του αρχείου που ανοίγει προσθέτουμε τις γραμμές που ακολουθούν αντικαθιστώντας το /yourname/ με τον δικό μας προσωπικό φάκελο.

```
# LD_LIBRARY_PATH
OTCL_LIB/home/yourname/ns-allinone-2.31/otcl-1.13
NS2_LIB=/home/yourname/ns-allinone-2.31/lib
X11_LIB=/usr/X11R6/lib
USR_LOCAL_LIB=/usr/local/lib
export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$X1
1_
LIB:$USR_LOCAL_LIB

# TCL_LIBRARY
TCL_LIB=/home/yourname/ns-allinone-
2.31/tcl8.4.14/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB

# PATH
XGRAPH=/home/yourname/ns-allinone-2.31/bin:
/home/yourname/ns-allinone-
2.31/tcl8.4.14/unix:/home/yourname/ns-allinone-
2.31/tk8.4.14/unix
NS=/home/yourname/ns-allinone-2.31/ns-2.31/
NAM=/home/networklab/ns/ns-allinone-2.31/nam-1.13/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

6. Στην γραμμή εντολών πληκτρολογούμε:

```
source ~/.bashrc
```

7. Αν η εγκατάσταση γίνει επιτυχώς κάνουμε επανεκκίνηση στο σύστημα και πληκτρολογούμε στην γραμμή εντολών:

```
$ ns
```

Μετά από αυτό θα εμφανιστεί στην οθόνη ένα «%». Πληκτρολογούμε exit για να επανέλθουμε στη γραμμή εντολών.

8. τελευταίο βήμα είναι η επικύρωση της εγκατάστασης. Αυτό γίνεται με την ακόλουθη εντολή και είναι ίσως το πιο χρονοβόρο τμήμα της εγκατάστασης:

```
$ cd ns-2.31
$ ./validate
```

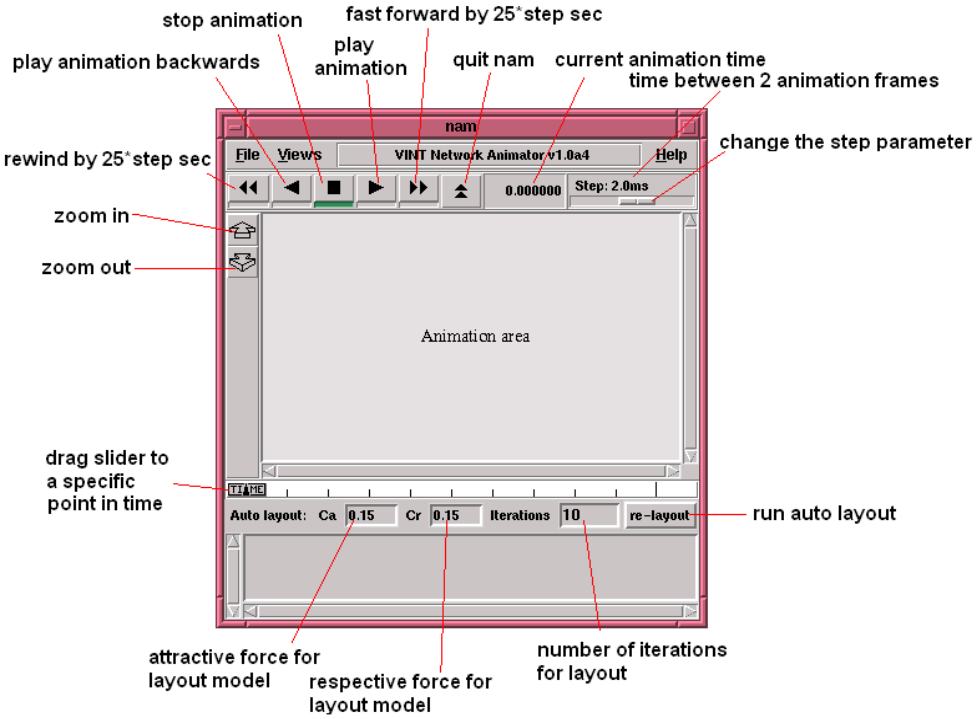
ΚΕΦΑΛΑΙΟ 7^{Ο}

NAM και XGRAPH

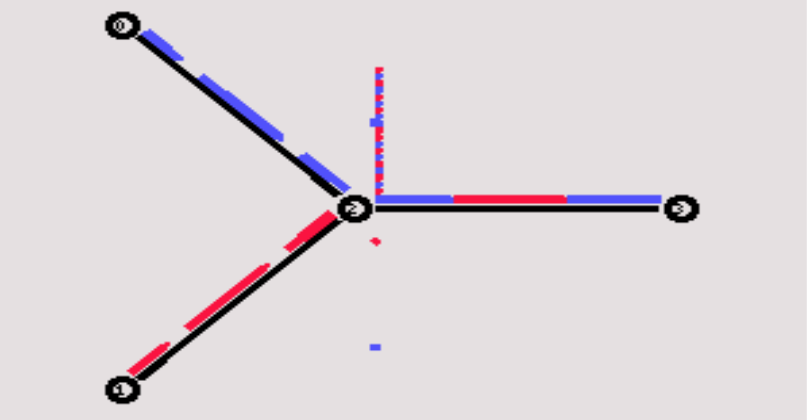
Πριν προχωρήσουμε στο πειραματικό μέρος θα ήθελα να αναφερθώ σε 2 πακέτα που χρησιμοποίησα στο NS-2, το NAM και το XGRAPH και αποδειχτήκαν χρήσιμα εργαλεία.

7.1 NAM

Το Nam είναι ένα εργαλείο γραφικής απεικόνισης που βασίζεται στην γλώσσα TCL/TK και χρησιμοποιείται για να εμφανίζει την τοπολογία του δικτύου, την παρακολούθηση της κίνησης των αρχείων και των πακέτων όπως αυτά συμβαίνουν στην προσομοίωση. Το Nam ξεκίνησε στο "*Lawrence Berkeley National Laboratory*" και έχει εξελιχθεί σημαντικά τα τελευταία χρόνια. Η ανάπτυξη του ήταν μια συνεχής προσπάθεια για συνεργασία με το Vint project.



Εικόνα 7.1, τα κουμπιά του NAM και οι λειτουργίες τους.

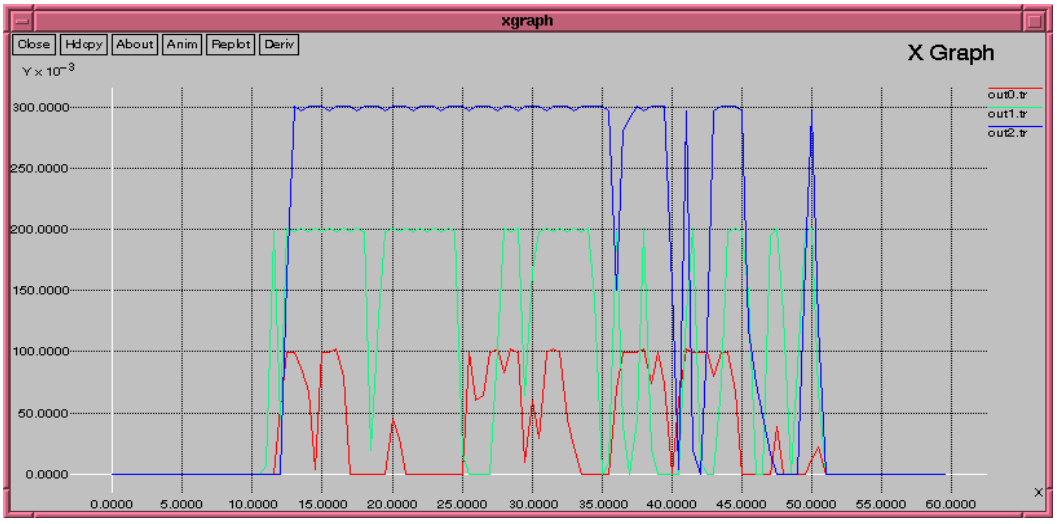


Εικόνα 7.2, κίνηση μεταξύ κόμβων και απόρριψη πακέτων όπως καταγράφηκε από το NAM.

7.2 XGRAPH

Το XGRAPH είναι ένας γενικού σκοπού plotter δεδομένων δυο διαστάσεων (x,y) με διαδραστικά κουμπιά για panning, ζουμ, εκτύπωση, και επιλογές οθόνης. Παράγει γραφήματα από οποιονδήποτε αριθμό αρχείων δεδομένων και μπορεί να χειριστεί θεωρητικά απεριόριστα μεγέθη δεδομένων καθώς και κάθε αριθμό αρχείων δεδομένων.

Το XGRAPH παράγει WYSIWYG PostScript, PDF, και MIF αρχεία και περιλαμβάνει τη δυνατότητα να καθορίσετε χρώματα για την σχεδίαση πολλαπλών στοιχείων καθώς και η γραμμή του πάχους. Υποστηρίζει επίσης την αυτόματη αλλαγή μεγέθους του παραθύρου και κάνει ζουμ σε οποιαδήποτε περιοχή της γραφικής παράστασης απλά σύροντας ένα πλαίσιο γύρω από την περιοχή.



Εικόνα 7.3, παράδειγμα γραφικής απεικόνισης του XGRAPH.

ΚΕΦΑΛΑΙΟ 8^{Ο}

Πειραματικό Μέρος

Φτάσαμε στο πιο ουσιαστικό μέρος της παρούσας εργασίας, το πειραματικό μέρος. Ο πιο εύκολος τρόπος να γνωριστούμε με την άγλωσσα TCL, την οποία χρησιμοποιούμε για τα σενάρια των προσομοιώσεων, είναι να μελετήσουμε την δομή ενός απλού παραδείγματος. Στην συνέχεια θα παραθέσω πιο σύνθετα αρχεία τα οποία θα αποτελούν και το βασικό μέρος του πειράματος.

8.1 Απλό Παράδειγμα

Το ακόλουθο παράδειγμα είναι ένα καλό ξεκίνημα για να εξοικειωθεί κάποιος με την γλώσσα Tcl και είναι αυτό που αναφέρεται σε κάθε εγχειρίδιο του NS-2. Δεν θα παρεκκλίνω λοιπόν από την εκπαιδευτική του αξία και θα το παραθέσω ελαφρώς βελτιωμένο με κάποια δικά μου σχόλια.

Τα Tcl scripts μπορούν να γραφούν σε οποιοδήποτε text editor με κατάληξη ”.tcl”.

Ας ονομάσουμε το παράδειγμα μας «example1.tcl». Πρώτη ενέργεια μας είναι να δημιουργήσουμε ένα αντικείμενο της τάξης Simulator. Αυτό γίνεται με την εντολή:

```
set ns [new Simulator]
```

Στην συνέχεια ανοίγουμε ένα αρχείο στο οποίο θα αποθηκευτούν τα στοιχεία (data trace) που είναι απαραίτητα για το NAM. Οι εντολές είναι οι ακόλουθες:

```
set nf [open out.nam w]
$ns namtrace-all $nf
```

Η πρώτη γραμμή ανοίγει το αρχείο, το ονομάζει «out.nam» και του δίνει το file handle 'nf'. Στη δεύτερη γραμμή λέμε στο αντικείμενο Simulator που δημιουργήσαμε πιο πάνω να γράψει όλα τα παραπάνω δεδομένα προσομοίωσης, που είναι σημαντικά και χρήσιμα για το NAM, σε αυτό το αρχείο.

Το επόμενο βήμα είναι να προσθέσουμε μια διαδικασία που θα κλείνει το αρχείο και θα ξεκινάει το NAM. Θα ονομάσουμε την διαδικασία αυτή «**finish**». Οι εντολές είναι οι ακόλουθες:

```
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
        exit 0
}
```

Στην παρούσα φάση αν δεν έχετε προγραμματιστικό ταλέντο ή ιδιαίτερη προϋπηρεσία δεν είναι εύκολο να καταλάβετε τι θέλουμε να πετύχουμε με αυτό το απόσπασμα κώδικα. Θα γίνει ξεκάθαρο στην πορεία όταν θα δούμε τον κώδικα ολοκληρωμένο.

Η επόμενη γραμμή λέει στο αντικείμενο Simulator να εκτελέσει την διαδικασία «finish» 5,0 δευτερόλεπτα μετά από την έναρξη της προσομοίωσης. Παρατηρήστε ότι το Ns μας παρέχει έναν πολύ απλό τρόπο για να προγραμματίσουν τα γεγονότα με την " **at**" εντολή.

```
$ns at 5.0 "finish"
```

Η τελευταία γραμμή είναι και αυτή που αρχίζει την προσομοίωση:

```
$ns run
```

Ολοκληρομένο το αρχείο μέχρι στιγμής :

```
#Create a simulator object
set ns [new Simulator]
#Open the nam trace file
set nf [open out.nam w]
```

```
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
#Close the trace file
        close $nf
    #Execute nam on the trace file
        exec nam out.nam &
        exit 0
}
#Call the finish procedure after 5 seconds simulation time
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

Στο επόμενο βήμα του παραδείγματος μας θα προσθέσουμε στο σενάριο της προσομοίωσης, που αυτή την στιγμή είναι κενό, δυο κόμβους και θα δημιουργήσουμε μια σύνδεση μεταξύ τους.

Οι εντολές που δημιουργούν τους 2 κόμβους θα πρέπει να εισαχθούν πριν από την γραμμή που ορίζει την διάρκεια της προσομοίωσης και είναι οι ακόλουθες:

```
set n0 [$ns node]
set n1[$ns node]
```

Βλέπουμε ότι ένα νέο αντικείμενο τύπου **node** (κόμβος) δημιουργείται με την εντολή «**$ ns node**». Στον παραπάνω κώδικα δημιουργεί δύο κόμβους και θα τους δίνει τα handles «**n0**» και «**n1**».

Η επόμενη γραμμή που είναι αυτή που συνδέει τους δύο κόμβους.

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

Η γραμμή αυτή λέει στο αντικείμενο **Simulator** να συνδέσει τους κόμβους «**n0**» και «**n1**» με μια duplex γραμμή, με το εύρος ζώνης **1Mbit**, delay **10ms** και μια ουρά **DropTail**.

Αν αποθηκεύσουμε τώρα το αρχείο και το τρέχουμε με την εντολή «**ns example1.tcl**» θα ξεκινήσει αυτόματα το NAM και θα πρέπει να δούμε ένα σχήμα που να μοιάζει με την παρακάτω εικόνα και θα είναι ουσιαστικά αυτό που περιγράψαμε στο σενάριο μας.

Ολοκληρωμένο το αρχείο μέχρι στιγμής :

```
#Create a simulator object
set ns [new Simulator]
#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
    #Close the trace file
        close $nf
    #Execute nam on the trace file
        exec nam out.nam &
        exit 0
}
#Create two nodes
set n0 [$ns node]
set n1 [$ns node]
#Create a duplex link between the nodes
```

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

Μέχρι στιγμής έχουμε φτιάξει μόνο την τοπολογία. Επόμενο βήμα είναι να βάλουμε κίνηση δεδομένων μεταξύ των κόμβων. Θα ορίσουμε λοιπόν κίνηση στέλνοντας δεδομένα από τον κόμβο «n0» στον κόμβο «n1». Αυτό θα το πετύχουμε δημιουργώντας ένα αντικείμενο τύπου agent στο n0 που θα στέλνει τα δεδομένα και ένα αντικείμενο τύπου **agent** στο n1 που θα τα λαμβάνει. Οι εντολές για να το πετύχουμε αυτό είναι οι ακόλουθες:

```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

Πιο αναλυτικά οι πρώτες 2 γραμμές δημιουργούν έναν **UDP agent (udp0)** και τον επιθέτουν στον κόμβο **n0**. Η τρίτη γραμμή δημιουργεί μια γεννήτρια CBR κίνησης, δηλαδή μια γεννήτρια παραγωγής σταθερού ρυθμού bit (constant bit rate generator) . Η τέταρτη και πέμπτη γραμμή ορίζουν το packetSize στα **500 byte** και το ενδιάμεσο χρόνο μεταξύ των πακέτων που αποστέλλονται τα **0,005 sec** (δηλαδή 200 πακέτα ανά δευτερόλεπτο). Η έκτη γραμμή εναποθέτει την γεννήτρια CBR στον UDP agent udp0.

Σχετικές παραμέτρους για κάθε τύπο agent μπορούμε να βρούμε στο εγχειρίδιο του ns στην σελίδα http://www.isi.edu/nsnam/ns/tutorial/

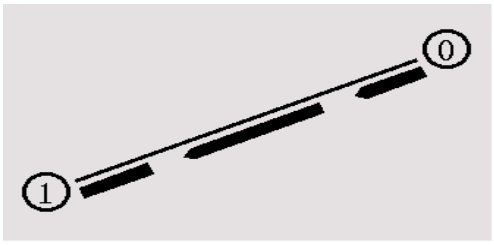
Οι επόμενες γραμμές θα δημιουργήσουν έναν **Null agent, τον null0** ο οποίος θα λαμβάνει την κίνηση (**traffic sink**) και τον επιθέτουμε στον κόμβο n1.

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

Έπειτα από αυτό μένει να συνδέσουμε τους agent μεταξύ τους, να ορίσουμε το χρονικό σημείο στο οποίο θα αρχίσει η αποστολή δεδομένων και το χρονικό σημείο που θα σταματήσει. Στο παράδειγμα μας ξεκινά στο 0,5 και τελειώνει στο 4,5.Οι εντολές για αυτό είναι οι ακόλουθες:

```
$ns connect $udp0 $null0
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
```

Τώρα μπορούμε να αποθηκεύσουμε το αρχείο και να ξεκινήσουμε πάλι την προσομοίωση. Αυτή την φορά όταν κάνουμε κλικ στο «**play**», του παράθυρου του NAM, θα δούμε ότι μετά από 0,5 δευτερόλεπτα προσομοίωσης, κόμβος 0 αρχίζει να στέλνει πακέτα κόμβο 1. Η εικόνα θα έχει μορφή παρόμοια με την εικόνα που ακολουθεί.



Η τελική έκδοση του αρχείου θα είναι η εξής:

```
#Create a simulator object
set ns [new Simulator]
#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
    #Close the trace file
```

```
            close $nf
        #Execute nam on the trace file
            exec nam out.nam &
            exit 0
    }
    #Create two nodes
    set n0 [$ns node]
    set n1 [$ns node]
    #Create a duplex link between the nodes
    $ns duplex-link $n0 $n1 1Mb 10ms DropTail
    #Create a UDP agent and attach it to node n0
    set udp0 [new Agent/UDP]
    $ns attach-agent $n0 $udp0
    # Create a CBR traffic source and attach it to udp0
    set cbr0 [new Application/Traffic/CBR]
    $cbr0 set packetSize_ 500
    $cbr0 set interval_ 0.005
    $cbr0 attach-agent $udp0
    #Create a Null agent (a traffic sink) and attach it
to node n1
    set null0 [new Agent/Null]
    $ns attach-agent $n1 $null0
    #Connect the traffic source with the traffic sink
    $ns connect $udp0 $null0
    #Schedule events for the CBR agent
    $ns at 0.5 "$cbr0 start"
    $ns at 4.5 "$cbr0 stop"
    #Call the finish procedure after 5 seconds of
simulation time
    $ns at 5.0 "finish"
    #Run the simulation
    $ns run
```

8.2 Προσομοιώσεις – Βασικό Πείραμα

Πριν παραθεσω τον κωδικα των προσομοιωσεων θα αναφερω λιγα πραγματα για το προτοκολλο δρομολογησης που χρησιμοποειται στα σεναρια, το DSDV καθώς και για τα αρχεία κίνησης και συνδέσεων.

8.2.1 Destination-Sequenced Distance-Vector

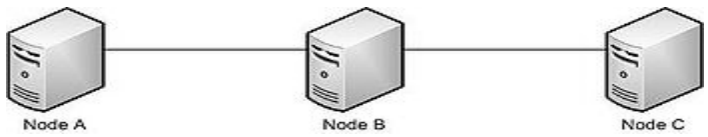
Το Destination-Sequenced Distance-Vector Routing (DSDV) έχει ως στόχο να διατηρήσει την απλότητα του αλγορίθμου Bellman-Ford και να αποφύγει το πρόβλημα των ατέρμονων βρόγχων (looping).

Σαν όλα τα Distance-Vector πρωτόκολλα έτσι και σε αυτό διατηρείται ένας πίνακας δρομολόγησης σε κάθε σταθμό. Κάθε εγγραφή του πίνακα αυτού περιλαμβάνει :

1. όλους τους πιθανούς προορισμούς ,
2. την απόσταση προς τον προορισμό μετρημένη σε hops,
3. τον επόμενο σταθμό προς τον προορισμό.

Κάθε κόμβος περιοδικά μεταδίδει στους γειτονικούς τους κόμβους τις τρέχουσες αποστάσεις των συντομότερων διαδρομών προς κάθε προορισμό.

Το DSDV προσεγγίζει κάθε σταθμό ως router και προσθέτει σε κάθε εγγραφή του πίνακα δρομολόγησης έναν αύξων αριθμό.



Εικονα 8.3

Για παράδειγμα, ο πίνακας δρομολόγησης του κόμβου Α στο δίκτυο της παραπάνω εικονας είναι

Destination	Next Hop	Number of Hops	Sequence Number	Install Time
A	A	0	A 46	001000
B	B	1	B 36	001200
C	B	2	B 28	001500

Ο πίνακας περιλαμβάνει περιγραφή όλων των δυνατών διαδρομών που είναι προσβάσιμες από κόμβο Α, μαζί με το επόμενο hop, τον αριθμό των επόμενων hop και τον αύξοντα αριθμό.

Μεταδωση πληροφοριων.

Οι πληροφορίες δρομολόγησης μεταδίδονται σε broadcast.

Τα updates μεταδίδονται περιοδικά και όταν συμβεί κάποια αλλαγή στην τοπολογία. Ο αύξων αριθμός προσδίδεται από τον προορισμό και είναι ζυγός αριθμός. Αν μια σύνδεση πέσει τότε ο αύξων αριθμός γίνεται μονός.

Full dumps μεταδίδονται σε αραιά διαστήματα και περιέχουν όλες τις πληροφορίες του κόμβου που τις απέστειλε. Τα Incremental dumps περιλαμβάνουν τις πληροφορίες που άλλαξαν από το τελευταίο Full dump.

Επιλογή διαδρομής

Για την επιλογή της διαδρομής χρησιμοποιούμε τον αύξων αριθμό διαλέγοντας τον πιο πρόσφατο.

Αν ληφθεί μια νέα διαδρομή που έχει τον ίδιο αύξων αριθμό τότε διαλέγουμε εκείνη με τον μικρότερο αριθμό hops.

Οι νεοεισαχθείσες διαδρομές παίρνουν προτεραιότητα για να αποσταλούν στους γειτονικούς σταθμούς.

Καταχωρήσεις του πινάκα οι οποίες δεν ενημερώνονται για κάποιο χρονικό διάστημα διαγράφονται.

Πλεονεκτήματα και Μειονεκτήματα του πρωτοκόλλου

Το DSDV είναι κατάλληλο για την δημιουργία μικρών ad hoc δικτύων. Ωστόσο απαιτεί συχνή ανανέωση των πινάκων δρομολόγησης. Αυτό σημαίνει κατανάλωση ενέργειας και αυξημένη χρήση του bandwidth ακόμα και όταν το δίκτυο δεν είναι απασχολημένο.

Επιπλέον κάθε φορά που η τοπολογία του δικτύου αλλάζει, ένας νέος αύξων αριθμός είναι απαραίτητος πριν να έχουμε σύγκλιση του δικτύου. Έτσι το DSDV δεν είναι κατάλληλο για highly dynamic networks.

Περά από το DSDV και άλλα πρωτόκολλα έχουν χρησιμοποιήσει παρόμοιες τεχνικές. Το πιο γνωστό distance vector πρωτόκολλο είναι το AODV, το οποίο μπορεί να χρησιμοποιήσει απλούστερες αλληλουχίας heuristics.

Στην παρούσα εργασία χρησιμοποίησα το DSDV γιατί τα σενάρια που προσομοίωσα δεν είχαν μεγάλο αριθμό κόμβων.

8.2.2 Αρχεία Κίνησης - Συνδέσεων

Στο βασικό κομμάτι του πειραματικού μέρους προσομοιώνεται κίνηση ασύρματου δικτύου για κάποιο χρονικό διάστημα. Σε αυτό το χρονικό διάστημα πρέπει να ορίσουμε κίνηση στους κόμβους του δικτύου, και να ορίσουμε και τις συνδέσεις.

Τον γεωγραφικό χώρο του δικτύου τον ορίζουμε στο TCL Script που περιγράφει την προσομοίωση και θα το δούμε πιο αναλυτικά αργότερα. Την κίνηση και τις συνδέσεις των κόμβων του δικτύου αλλά και τις παραμέτρους που μπορούμε να εφαρμόσουμε πάνω σε αυτά (πχ. Ταχύτητα κίνησης, είδος σύνδεσης κλπ) μπορούμε να τα περάσουμε παραμετρικά στο TCL Script.

Το NS-2 μέσω των αρχείων «setdest» και «cbrgen» μας δίνει την δυνατότητα να δημιουργήσουμε αρχεία κειμένου τα οποία παράγουν τυχαία κίνηση και μια τυχαία αλληλουχία συνδέσεων μεταξύ των κόμβων αντίστοιχα. Ας τα δούμε αναλυτικότερα

8.2.2.1 Τυχαία Κίνηση Ασύρματων Κόμβων

Είπαμε ήδη ότι η τυχαία κίνηση των κόμβων γίνεται με την χρήση της εντολής «**setdest**». Η «**setdest**» εκτελείται σύμφωνα με την παρακάτω μορφή στον φάκελο **/home/yourfolder/ns-allinone-2.31/ns-2.31/indep-utils/cmu-scen-gen** και συντάσσεται ως εξής:

./setdest -n Α -p Β -M Γ -t Δ -x Ε -y Ζ > arxeio_kinisis_komvon

- Το «-n Α» ορίζει τον αριθμό των κόμβων, Α κόμβοι,
- Το «-p Β» ορίζει τον χρόνο παύσης, Β sec χρονικό διάστημα,
- Το «-M Γ» ορίζει την ταχύτητα των κόμβων, Μ m/sec
- Το «-t Δ» ορίζει τον χρόνο της προσομοίωσης, Δ sec χρονικό διάστημα
- Τα «-x Ε» και «-y Ζ» ορίζουν τις διαστάσεις τοθ γεωγραφικού χώρου στον οποίο θα κινούνται οι κόμβοι
- Το αποτέλεσμα της διαδικασίας θα αποθηκευτεί στο αρχείο **«arxeio_kinisis_komvon».**

Οι εντολές που χρησιμοποίησα στην παρούσα εργασία είναι οι ακόλουθες:

1. **./setdest -n 5 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi5komvoi**
2. **./setdest -n 10 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi10komvoi**
3. **./setdest -n 15 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi15komvoi**
4. **./setdest -n 20 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi20komvoi**
5. **./setdest -n 25 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi25komvoi**
6. **./setdest -n 30 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi30komvoi**
7. **./setdest -n 35 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi35komvoi**
8. **./setdest -n 40 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi40komvoi**
9. **./setdest -n 45 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi45komvoi**
10. **./setdest -n 50 -p 2.0 -M 10 -t 120 -x 500 -y 500 > kinisi50komvoi**

Όπως βλέπουμε οι προσομοιώσεις θα περιγράψουν ασύρματα δίκτυα από 5 έως 50 κόμβους αυξανόμενους κάθε φορά κατά 5 επιπλέον κόμβους. Οι κόμβοι θα κινούνται με **10 m/sec**, για **120 sec** διάρκεια προσομοίωσης σε ένα γεωγραφικό χώρο **500m** επί **500m** και με χρόνο παύσης τα **2 sec**.

8.2.2.2 Τυχαίες Συνδέσεις Ασύρματων Κόμβων

Παρόμοια με την τυχαία κίνηση των κόμβων δημιουργούμαι και τις τυχαίες συνδέσεις. Στον ίδιο φάκελο με πριν στην γραμμή εντολών πληκτρολογούμε:

ns cbrgen.tcl -type A -nn B -seed Γ -mc Δ -rate E > TuxaiesSundeseis

- Το «-type A» ορίζει τον τύπο της σύνδεσης (TCP / CBR),
- Το «-nn B» ορίζει τον αριθμό των κόμβων, Β κόμβοι,
- Το «-seed Γ» ορίζει την γεννήτρια των τυχαίων χρόνων έναρξης της επικοινωνίας
- Το «-mc Δ» ορίζει τον μέγιστο αριθμό συνδέσεων, Δ συνδέσεις,
- Το «-rate E» ορίζει το χρονικό διάστημα μεταξύ των CBR πακέτων.
- Το αποτέλεσμα της διαδικασίας θα αποθηκευτεί στο αρχείο «**TuxaiesSundeseis**»

Οι εντολές που χρησιμοποίησα στην παρούσα εργασία είναι οι ακόλουθες:

1. **ns cbrgen.tcl -type tcp -nn 5 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis5komvon**
2. **ns cbrgen.tcl -type tcp -nn 10 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis10komvon**
3. **ns cbrgen.tcl -type tcp -nn 15 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis15komvon**
4. **ns cbrgen.tcl -type tcp -nn 20 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis20komvon**
5. **ns cbrgen.tcl -type tcp -nn 25 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis25komvon**
6. **ns cbrgen.tcl -type tcp -nn 30 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis30komvon**
7. **ns cbrgen.tcl -type tcp -nn 35 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis35komvon**
8. **ns cbrgen.tcl -type tcp -nn 40 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis40komvon**
9. **ns cbrgen.tcl -type tcp -nn 45 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis45komvon**
10. **ns cbrgen.tcl -type tcp -nn 50 -seed 0 -mc 30 -rate 0 > tuxaies_sundeseis50komvon**

8.2.3 Παράδειγμα αρχείου προσομοίωσης

Για να γινει κατανοητο το αρχειο που περιγραφει την προσομοιωση ενος ασυρματου δικτιου θα το παραθεσω αυτουσιο με καποια σχολια οπου κρινω οτι χρειαζονται. Στην **Tcl** οτι ακολουθει το συμβολο ¨#¨ θεωρείτε σχολιο. Το συνολο των σεναριων προσομοιωσεων θα το παραθεσω σε παραρτημα στο τελος της παρουσας εργασιας.

```
set opt(chan)  Channel/WirelessChannel        # τύπος του καναλιού
set opt(prop)  Propagation/TwoRayGround # μοντελο ραδιομετάδωσης
# οι τιμες που μπορει να παρει ειναι οι : FreeSpace, TwoRayGround,
# Shadowing. Στην 1η περιπτοσι υποθετουμε οτι η διαδρομη μεταξυ πομπου
 # και δεκτη ειναι αμεση, στην 2η συνυπολογιζεται και η ανακλαση των
# σηματων στο εδαφος και τελος στην 3η προσομοιωνεται και αποσβεση του
# σηματος.
set opt(netif) Phy/WirelessPhy ;# οριζει να ειναι ασυρματο το interface #του δικτυου
set opt(mac)        Mac/802_11 ;# οριζεται ο τυπος της MAC του ασυρματου # interface

set opt(ifq)        Queue/DropTail/PriQueue ;# οριζει τον τυπο της ουρας
# στο interface
set opt(ll)         LL ;# link layer
set opt(ant)        Antenna/OmniAntenna ;# οριζει τον τυπο της κεραιας
set opt(x)          500 ;# οριζει την διασταση X της τοπογραφιας
set opt(y)          500 ;# οριζει την διασταση Y της τοπολογιας
set opt(cp)  "/home/kele/tcl-files/arxeia_sundeseon_komvon/sundeseis5komvon"
# οριζει το αρχειο συνδεσεων
set opt(sc)    "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi5komvoi"
# οριζει το αρχειο κινησης
set opt(ifqlen)     50 ;# οριζει τον μεγιστο αριθμο πακετων στο ifq
set opt(seed)  0.0  ;# seed for random number gen.
set opt(nn)         5   ;# οριζει τον αριθμο των κομβων
set opt(stop)  120.0 ;# οριζει τον χρονο της προσομοιωσης
set opt(rp)              DSDV;# οριζει το προτοκολλο ασυρματης δρομολογησης
```

```
#  Other default settings
set AgentTrace             ON
set RouterTrace            OFF
set MacTrace               OFF

LL set mindelay_               50us
LL set delay_                  25us
LL set bandwidth_              0     ;# not used

Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_       0
Agent/TCPSink set dport_       0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460


Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

#ρυθμιση της κεραιας ωστε να ειναι 1,5 μετρο πανο απο
τους κομβους.
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

#παραμετροποιηση του κοινου μεσου.
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0
     # παραμετροποιηση της MAC
     $opt(mac) set CWMin_ 15; # IEEE 802.11g uses
                              #a lower CWMin...
     $opt(netif) set bandwidth_    54e6
     $opt(mac) set dataRate_       54e6
     $opt(mac) set basicRate_ 6e6
     $opt(mac) set PLCPDataRate_   6e6
     $opt(mac) set PreambleLength_ 0.000016

# check for boundary parameters and random seed
```

```
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# δημιουργια στιγμιοτυπου Simulator
set ns_   [new Simulator]

# δημιουργια καναλιου
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

#δημιουργια αρχειου trace
set tracefd  [open kin5gtr.tr w]

# δημιουργια αρχειου για το NAM
set namtrace [open kin5gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# καθορισμος της τοπολογιας
$topo load_flatgrid $opt(x) $opt(y)

# δημιουργια αντικειμενου God
 set god_ [create-god $opt(nn)]

# παραμετροποιηση του αντικειμενου κομβου
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan


# δημιουργια των 5 κομβων
for {set i 0} {$i < $val(nn) } {incr i} {
```

```
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0          ;# disable random
motion
}

#καθορισμος του μεγεθους του κομβου
for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}

# συνδεση αρχειων συνδεσεων-κινησης
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# καθορισμος του μεγεθους του κομβου στο NAM
for {set i 0} {$i < $opt(nn)} {incr i} {
     # το 20 καθοριζει το μεγεθος στο NAM συμφωνα με το
σεναριο μας. Η
     # μεθοδος καλειται αφου καθοριστει το αρχειο κινησης
    $ns_ initial_node_pos $node_($i) 20
}

# καθοριζει ποτε τελειονει η προσομοιωση
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"

proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}


# informative headers for CMUTracefile
```

```
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```

ΚΕΦΑΛΑΙΟ 9^{Ο}

Μετρήσεις - Συμπεράσματα

Σε αυτό το κεφάλαιο θα μελετήσουμε τις μετρήσεις κάποιων βασικών ιδιοτήτων κάθε δικτύου του Throughput και του Delay/

9.1 Throughput (Διαμεταγωγή)

Στην πληροφορική ονομάζουμε **διαμεταγωγή** ή **ρυθμαπόδοση** ή **ταχύτητα μεταφοράς πληροφορίας** τον χρονικό ρυθμό με τον οποίον ένας υπολογιστής, ένα δίκτυο υπολογιστών ή ένα υποσύστημα ενός υπολογιστή αποστέλλει ή λαμβάνει δεδομένα. Το **Throughput** αποτελεί ένα μέτρο της χωρητικότητας ενός τηλεπικοινωνιακού καναλιού ή μίας επικοινωνιακής σύνδεσης. Οι συνδέσεις αυτές συνήθως χαρακτηρίζονται από το **bit rate** τους, δηλαδή πόσα bit μεταδίδονται μέσω αυτών των συνδέσεων σε ένα δευτερόλεπτο (bits/sec).

9.2 Network Delay (Καθυστέρηση Δικτύου)

Η πρώτη σημαντική ιδιότητα ενός δικτύου που μπορεί να μετρηθεί ποσοτικά είναι το **Delay**. Η καθυστέρηση ενός δικτύου καθορίζει το πόσο χρονικό διάστημα χρειάζεται ένα bit για να ταξιδέψει από έναν υπολογιστή σε έναν άλλο μέσο του δικτύου. Η καθυστέρηση μετριέται σε δευτερόλεπτα ή κλάσματα δευτερολέπτου. Μπορεί να διαφέρει ελαφρώς, ανάλογα με τη θέση του ζεύγους των υπολογιστών που επικοινωνούν.

Αν και οι χρήστες ενδιαφέρονται μόνο για τη συνολική καθυστέρηση του δικτύου, οι μηχανικοί κάνουν πιο ακριβείς μετρήσεις. Έτσι, συνήθως έχουμε την μέγιστη και μέση καθυστέρηση, και επίσης μπορούν να διαχωρίζουν την καθυστέρηση σε άλλες υποκατηγορίες:

- **Processing delay** : η ώρα που χρειάζεται ένας δρομολογητής για να αναλάβει τη διαδικασία του πακέτου κεφαλίδας
- **Queuing delay** : ο χρόνος που το πακέτο βρίσκεται σε ουρά δρομολόγησης.
- **Transmission delay** : ο χρόνος που χρειάζεται για να προωθηθεί το πακέτο.
- **Propagation delay** : ο χρόνος που χρειάζεται το σήμα να μεταδοθεί στο μέσο επικοινωνίας.

Υπάρχει ένα ελάχιστο επίπεδο καθυστέρηση που θα οφείλεται στο χρόνο που χρειάζεται για να μεταδοθεί ένα πακέτο σειριακά μέσω ενός link.Σε αυτό προστίθεται μια πιο μεταβλητό επίπεδο καθυστέρησης που οφείλεται στην συμφόρηση του δικτύου (network congestion).

9.3 Εξαγωγή Στατιστικών

Στατιστικά στοιχεία για τα σενάρια μπορούμε να βγάλουμε με την χρήση κάποιου script. Το script που χρησιμοποίησα είναι γραμμένο σε awk και υπάρχει σε αρκετά forum σχετικά με το NS2.

9.3.1 Script εξαγωγής δεδομένων «genstats» :

```
#! /usr/bin/awk -f
#
# Parse a ns2 wireless trace file and generate the
following stats:
# - number of flows (senders)
# - time of simulation run
# - number of packets sent (at the Application)
# - number of packets received (at the Application)
# - number of packets dropped (at the Application)
# - number of collisions (802.11)
# - average delay
# - average throughput
# - average traffic rate (measured)
#
# Last updated: April 5, 2002 0113 mm
function average (array) {
    sum = 0;
    items = 0;
    for (i in array) {
        sum += array[i];
        items++;
```

```
    }
# printf("DEBUG sum is %d, items is %d\n", sum, items);

if (sum == 0 || items == 0)
        return 0;
    else
        return sum / items;
}

function max( array ) {
    for (i in array) {
        if (array[i] > largest)
            largest = array[i];
    }
    return largest;
}

function min(array) {
    for (i in array) {
        if (0 == smallest)
            smallest = array[i];
        else if (array[i] < smallest)
            smallest = array[i];
    }
    return smallest;
}
BEGIN {
    total_packets_sent = 0;
    total_packets_received = 0;
    total_packets_dropped = 0;
    first_packet_sent = 0;
    last_packet_sent = 0;
    last_packet_received = 0;
}
{
    event = $1;
    time = $2;
    node = $3;
    type = $4;
    reason = $5;
    packetid = $6;

# strip leading and trailing _ from node
    sub(/^_*/, "", node);
    sub(/_*$/, "", node);

    if ( time < simulation_start || simulation_start == 0
)
        simulation_start = time;
    if ( time > simulation_end )
        simulation_end = time;
```

```
    if ( reason == "COL" )
            total_collisions++;

    if ( type == "AGT" ) {
        nodes[node] = node; # to count number of nodes
        if ( time < node_start_time[node] ||
node_start_time[node] == 0 )
            node_start_time[node] = time;

        if ( time > node_end_time[node] )
            node_end_time[node] = time;

        if ( event == "s" ) {
           flows[node] = node; # to count number of flows

            if ( time < first_packet_sent ||
first_packet_sent == 0 )
                first_packet_sent = time;
            if ( time > last_packet_sent )
                last_packet_sent = time;
            # rate
            packets_sent[node]++;
            total_packets_sent++;

            # delay
            pkt_start_time[packetid] = time;
        }
        else if ( event == "r" ) {
            if ( time > last_packet_received )
                last_packet_received = time;
            # throughput
            packets_received[node]++;
            total_packets_received++;

            # delay
            pkt_end_time[packetid] = time;
        }
        else if ( event == "D" ) {
            total_packets_dropped++;
# pkt_end_time[packetid] = time; # EXPERIMENTAL
        }
    }
}
END {
    print "" > "throughput.dat";
    print "" > "rate.dat";
    number_flows = 0;
    for (i in flows)
        number_flows++;

    # find dropped packets
    if ( total_packets_sent != total_packets_received ) {
```

```
        printf("***OUCH*** Dropped Packets!\n\n");
        for ( packetid in pkt_start_time ) {
            if ( 0 == pkt_end_time[packetid] ) {
                total_packets_dropped++;
#               pkt_end_time[packetid] = simulation_end;
# EXPERIMENTAL
            }
        }
    }

    for (i in nodes) {
        if ( packets_received[i] > 0 ) {
            end = node_end_time[i];
            start = node_start_time[i - number_flows];
            runtime = end - start;
            if ( runtime > 0 ) {
                throughput[i] = packets_received[i]*8000
/ runtime;
                printf("%d %f %f %d\n", i, start, end,
throughput[i]) >> "./throughput.dat";
            }
        }
        # rate - not very accurate
        if ( packets_sent[i] > 2 ) {
            end = node_end_time[i];
            start = node_start_time[i];
            runtime = end - start;
            if ( runtime > 0 ) {
                rate[i] = (packets_sent[i]*8000) /
runtime;
                printf("%d %f %f %d\n", i, start, end,
rate[i]) >> "./rate.dat";
            }
        }
    }

    # delay
    for ( pkt in pkt_end_time) {
        end = pkt_end_time[pkt];
        start = pkt_start_time[pkt];
        delta = end - start;
        if ( delta > 0 ) {
            delay[pkt] = delta;
            printf("%d %f %f %f\n", pkt, start, end,
delta) >> "./delay.dat";
        }
    }

    # offered load
    total_runtime = last_packet_sent - first_packet_sent;
    if ( total_runtime > 0 && total_packets_sent > 0)
```

```
        load = ((total_packets_sent *
8000)/total_runtime) / 2000000; # n=o overhead

  printf ("       RUN   OFFERED PACKETS  PACKETS  PACKETS
AVERAGE    MAX        MIN        AVERAGE      AVERAGE");
  printf (" FLOWS TIME  LOAD    SENT     RECEIVED DROPPED
COLLISIONS DELAY      DELAY      DELAY      THROUGHPUT
TRAFFIC RATE");
  printf (" ----- ----- ------- -------- -------- -------
- ---------- ---------- ---------- ---------- ----------
- ------------");

printf("%5d %5.1f %7.4f %8d %8d %8d %10d %10.4f %10.4f
%10.4f %12d %12d\n",
           number_flows,
           total_runtime,
           load,
           total_packets_sent,
           total_packets_received,
           total_packets_dropped,
           total_collisions,
           average(delay),
           max(delay),
           min(delay),
           average(throughput),
           average(rate));

    printf("%5d %5.1f %7.4f %8d %8d %8d %10d %10.4f
%10.4f %10.4f %12d %12d\n",
           number_flows,
           total_runtime,
           load,
           total_packets_sent,
           total_packets_received,
           total_packets_dropped,
           total_collisions,
           average(delay),
           max(delay),
           min(delay),
           average(throughput),
           average(rate)) >> "./stats.dat";
}
```

9.3.2 Εντολές Εξαγωγής Στατιστικών για κάθε Σενάριο

Οι εντολές με τις οποίες θα πάρουμε τα στατιστικά στοιχεία εφαρμόζονται πάνω στα trace αρχεία που πήραμε όταν τρέξαμε τα σενάρια των προσομοιώσεων στο NS2 και επί του προκειμένου είναι οι ακόλουθες.

Σε γραμμή εντολών, στον φάκελο που βρίσκονται τα trace αρχεία, δηλαδή τα αρχεία με κατάληξη «.tr», γράφουμε:

```
gawk -f genstats.awk kin5gtr.tr

gawk -f genstats.awk kin10gtr.tr

gawk -f genstats.awk kin15gtr.tr

gawk -f genstats.awk kin20gtr.tr

gawk -f genstats.awk kin25gtr.tr

gawk -f genstats.awk kin30gtr.tr

gawk -f genstats.awk kin35gtr.tr

gawk -f genstats.awk kin40gtr.tr

gawk -f genstats.awk kin45gtr.tr

gawk -f genstats.awk kin50gtr.tr
```

Το αποτέλεσμα μετά την εκτέλεση κάθε εντολής, εφόσον όλα είναι σωστά θα έχει την ακόλουθη μορφή:

OUCH Dropped Packets!

	RUN	OFFERED	PACKETS	PACKETS	PACKETS		AVERAGE	MAX	MIN	AVERAGE	AVERAGE
FLOWS	TIME	LOAD	SENT	RECEIVED	DROPPED	COLLISIONS	DELAY	DELAY	DELAY	THROUGHPUT	TRAFFIC RATE
5	78.0	8.9471	174451	174030	421	3558	0.0157	25.4447	0.0002	2320641	4324016

9.4 Στατιστικά Αποτελέσματα

Τα αποτελέσματα που θα ακολουθήσουν βγήκαν από μετρήσεις που έγιναν σε σενάρια ασύρματων τοπικών δικτύων με μέγεθος από πέντε έως πενήντα κόμβους, ανά πέντε κόμβους, χρησιμοποιώντας το πρωτόκολλο 802.11g, τοπολογία AD-HOC, τυχαία κίνηση των κόμβων σε προκαθορισμένη περιοχή και τυχαίες συνδέσεις μεταξύ των κόμβων.

ΑΡΙΘΜΟΣ ΚΟΜΒΩΝ	FLOWS	RUN TIME	OFFERED LOAD	PACKETS SENT	PACKETS RECEIVED	MAX DELAY	MIN DELAY	AVERAGE TRAFFIC RATE
5	5	78.0	89.471	174451	174030	254.447	0.0002	4324016
10	9	119.7	89.157	266736	263795	216.838	0.0002	3050347
15	11	103.1	121.914	314226	312730	26.069	0.0002	3021796
20	10	81.6	128.089	261185	260792	57.058	0.0002	3537928
25	24	108.5	125.051	339130	336886	42.169	0.0002	1413830
30	25	116.7	124.814	364280	361704	42.169	0.0002	1299958
35	28	115.3	107.249	309268	307256	846.714	0.0002	1082184
40	22	89.2	117.475	261872	259462	457.144	0.0002	1428957
45	29	119.6	99.656	297996	295599	204.340	0.0002	998284
50	29	119.6	98.438	294355	292729	828.423	0.0002	977461

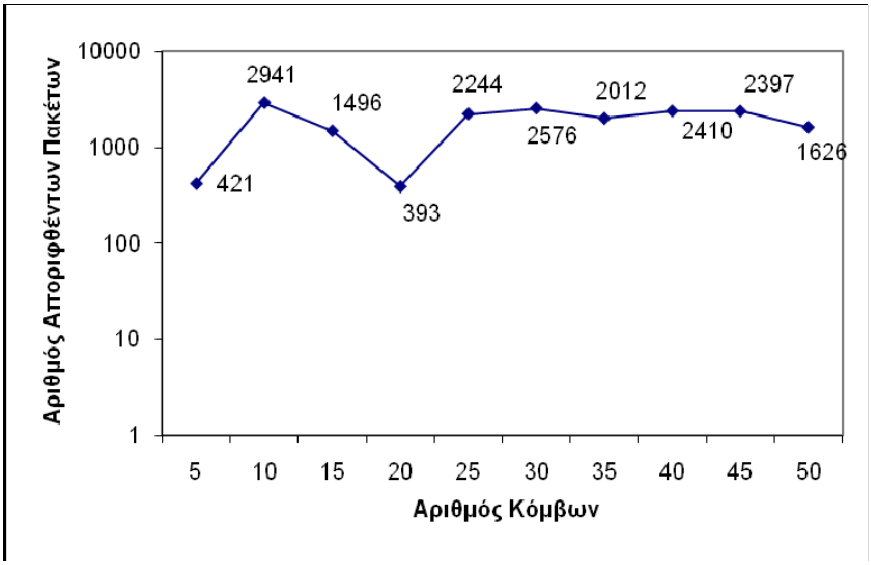
Πίνακας 9.1, δευτερεύοντα στατιστικά στοιχεία των σεναρίων προσομοιώσεων ασύρματων δικτύων.

ΑΡΙΘΜΟΣ ΚΟΜΒΩΝ	PACKETS DROPPED	COLLISIONS	AVERAGE DELAY	AVERAGE THROUGHPUT
5	421	3558	0.0157	2320641
10	2941	17810	0.0298	2274113
15	1496	36219	0.0322	2569097
20	393	19270	0.0234	4980620
25	2244	68490	0.0642	1085013
30	2576	77225	0.0665	1026120
35	2012	74255	0.0668	911054
40	2410	49141	0.0514	135314
45	2397	92258	0.0760	741245
50	1626	98793	0.0810	744837

Πίνακας 9.2, πρωτεύοντα στατιστικά στοιχεία των σεναρίων προσομοιώσεων ασύρματων δικτύων.

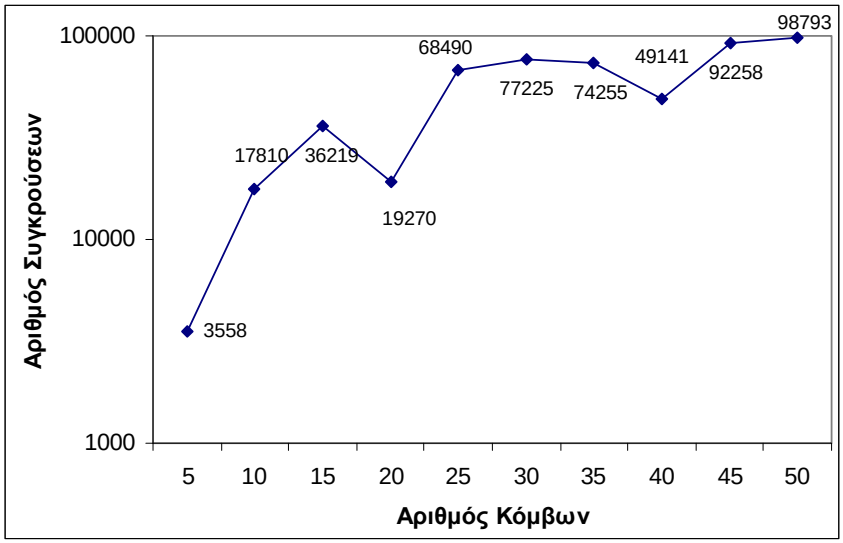
Μπορούμε να τα δούμε και με γραφική απεικόνιση, έτσι θα μπορέσουμε να βγάλουμε συμπεράσματα ευκολότερα.

9.4.1 Διάγραμμα 1, Αριθμός Απορριφθέντων Πακέτων.



Στο σχεδιάγραμμα βλέπουμε πως κυμαίνεται ο αριθμός των απορριφθέντων πακέτων. Επειδή τα σενάρια προσομοιώνουν κινητούς κόμβους και τυχαίες συνδέσεις δεν μπορούμε να βγάλουμε κάποιο ασφαλές συμπέρασμα για το ποια παράμετρος επηρεάζει τον αριθμό αυτό. Πάντως πλην των περιπτώσεων των 5 και 20 κόμβων φαίνεται ο αριθμός αυτός να κινείτε στα ιδία πλαίσια για όλα τα σενάρια.

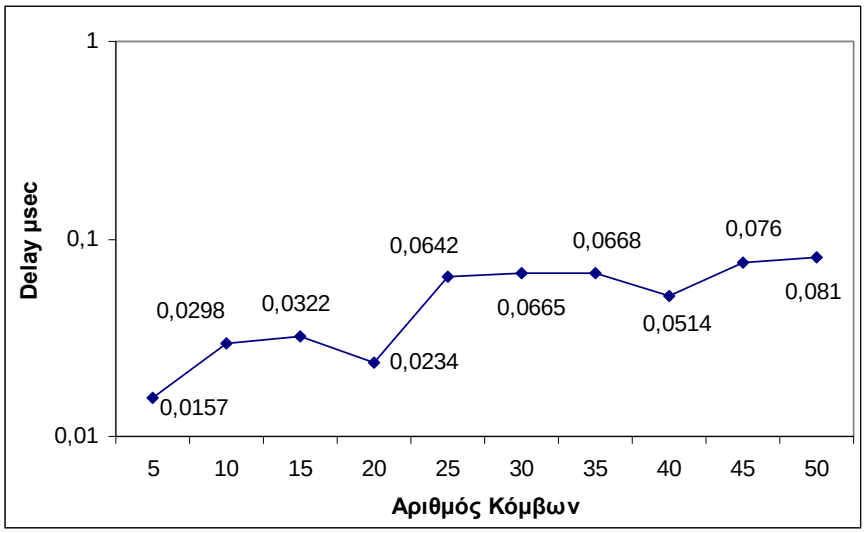
9.4.2 Διάγραμμα 2, Αριθμός συγκρούσεων



Στο διάγραμμα αυτό βλέπουμε το αναμενόμενο. Όσο αυξάνεται ο αριθμός των κόμβων ανά σενάριο αυξάνεται και ο αριθμός των συγκρούσεων. Και εδώ

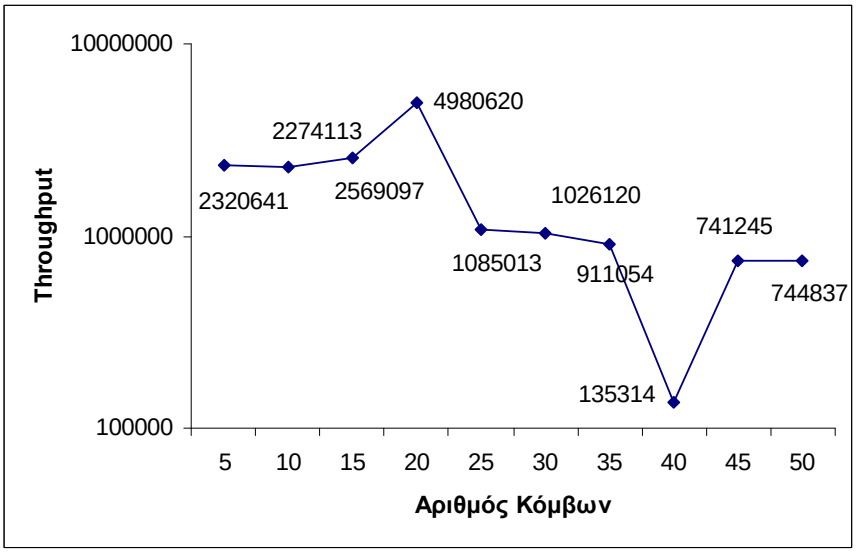
εξαιρείται η περίπτωση των 20 κόμβων αλλά και των 40, που είναι οι πιο εμφανής, αλλά αυτό είναι κάτι που οφείλεται στο τυχαίο παράγοντα των συνδέσεων μεταξύ των κόμβων αλλά και των μεταξύ τους αποστάσεων που αλλάζουν καθώς αυτοί μετακινούνται στον χώρο.

9.4.3 Διάγραμμα 3, Μέση Καθυστέρηση (Delay)



Στο διάγραμμα αυτό βλέπουμε πως κυμαίνεται η καθυστέρηση στα σενάρια. Όσο αυξάνεται ο αριθμός των σταθμών αυξάνεται και η καθυστέρηση στην επικοινωνία. Στην περίπτωση των 20 και των 40 κόμβων το delay παρεκκλίνει της ανοδικής πορείας. Αυτό συμβαίνει γιατί όπως είδαμε στο προηγούμενο διάγραμμα πέφτει ο αριθμός των συγκρούσεων. Άρα ένας παράγοντας που παίζει ρόλο στην διαμόρφωση του delay είναι και ο αριθμός συγκρούσεων.

9.4.4 Διάγραμμα 4, Μέση Μεταγωγή (Average Throughput)



Στο διάγραμμα αυτό βλέπουμε ότι το throughput αυξάνεται μέχρι το σενάριο των 20 κόμβων, έπειτα μειώνεται μέχρι τους 40 κόμβους. Στους 25 και στους 40 κόμβους η πτώση είναι ραγδαία, όπως ραγδαία είναι και η αύξηση του Throughput στους 45 κόμβους.

Παραρτημα

Στο παραρτημα θα παρατεθουν οι κωδικες των δεκα αρχειων προσομοιοσης αλλα και ενδεικτικα από ένα αρχειο για τις τυχαιες συνδεσεις και τυχαια κινηση του σεναριου των 5 κομβων

ΣΕΝΑΡΙΟ 5 ΚΟΜΒΩΝ

Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 5 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

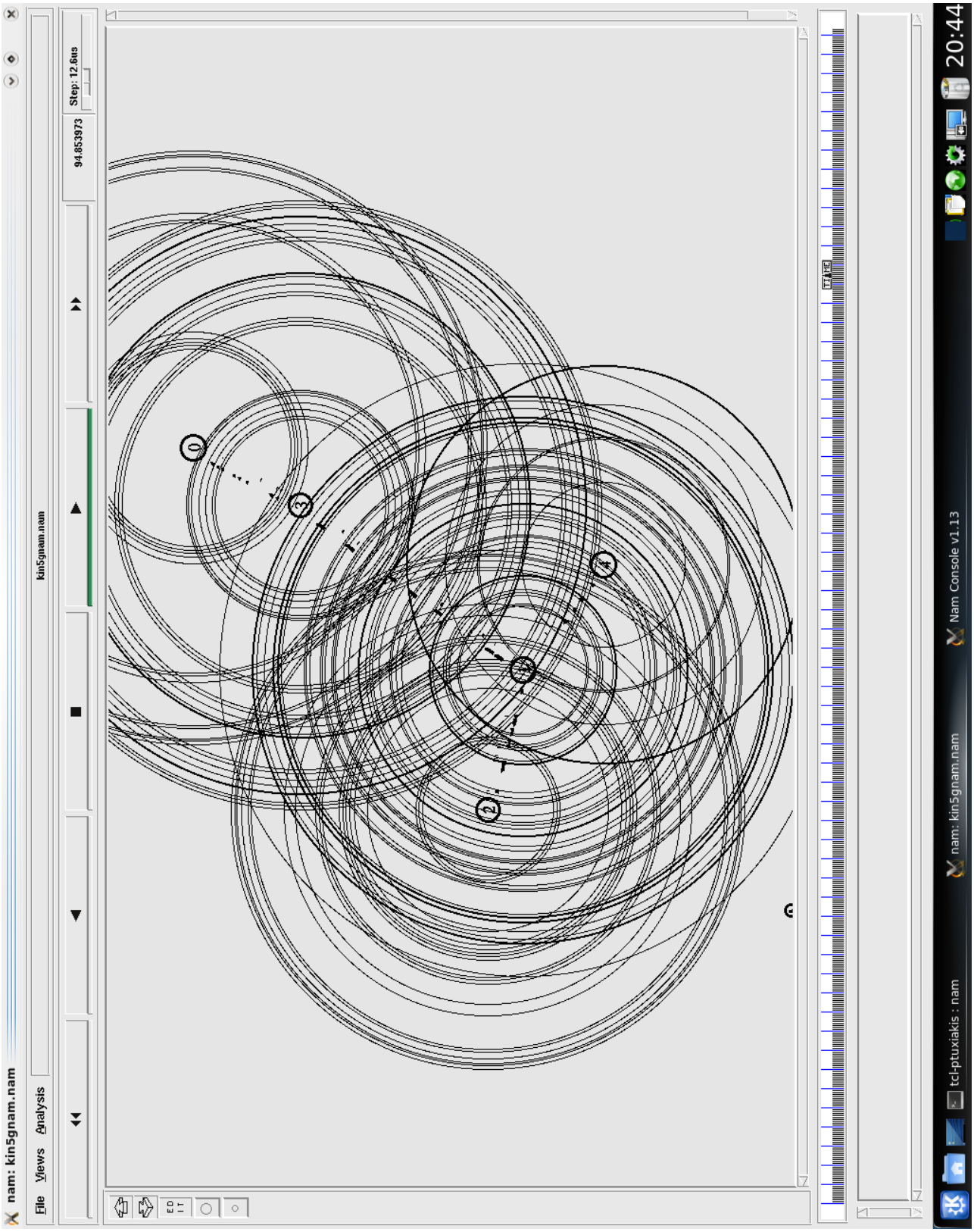
```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500               ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)"/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis5komvon"          ;
set opt(sc)"/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi5komvoi"
               ;

set opt(ifqlen)     50             ;# max packet in ifq

set opt(seed)            0.0       ;# seed for random
number gen.

set opt(nn)              5         ;# number of
mobilenodes
set opt(stop)            120.0     ;# simulation time

set opt(adhocRouting)    NOAH      ;# routing protocol
```

```
set opt(rp)             DSDV      ;# routing protocol

# ============= Other default settings
======================

set AgentTrace          ON
set RouterTrace         OFF
set MacTrace            OFF

LL set mindelay_              50us
LL set delay_                 25us
LL set bandwidth_             0    ;# not used

Agent/Null set sport_         0
Agent/Null set dport_         0

Agent/CBR set sport_          0
Agent/CBR set dport_          0

Agent/TCPSink set sport_      0
Agent/TCPSink set dport_      0

Agent/TCP set sport_          0
Agent/TCP set dport_          0
Agent/TCP set packetSize_     1460



Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

           $opt(mac) set CWMin_ 15 ;# IEEE 802.11g uses a
lower CWMin...
```

```
     $opt(netif) set bandwidth_    54e6
     $opt(mac) set dataRate_       54e6
     $opt(mac) set basicRate_      6e6
     $opt(mac) set PLCPDataRate_   6e6
     $opt(mac) set PreambleLength_      0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin5gtr.tr w]
set namtrace [open kin5gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
           -macType $opt(mac) \
           -ifqType $opt(ifq) \
           -ifqLen $opt(ifqlen) \
           -antType $opt(ant) \
           -propType $opt(prop) \
           -phyType $opt(netif) \
           -topoInstance $topo \
           -agentTrace ON \
           -routerTrace ON \
           -macTrace ON \
           -movementTrace OFF \
           -channel $chan
```

```
for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
#20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"

proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
```

```
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.1, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 10 ΚΟΜΒΩΝ

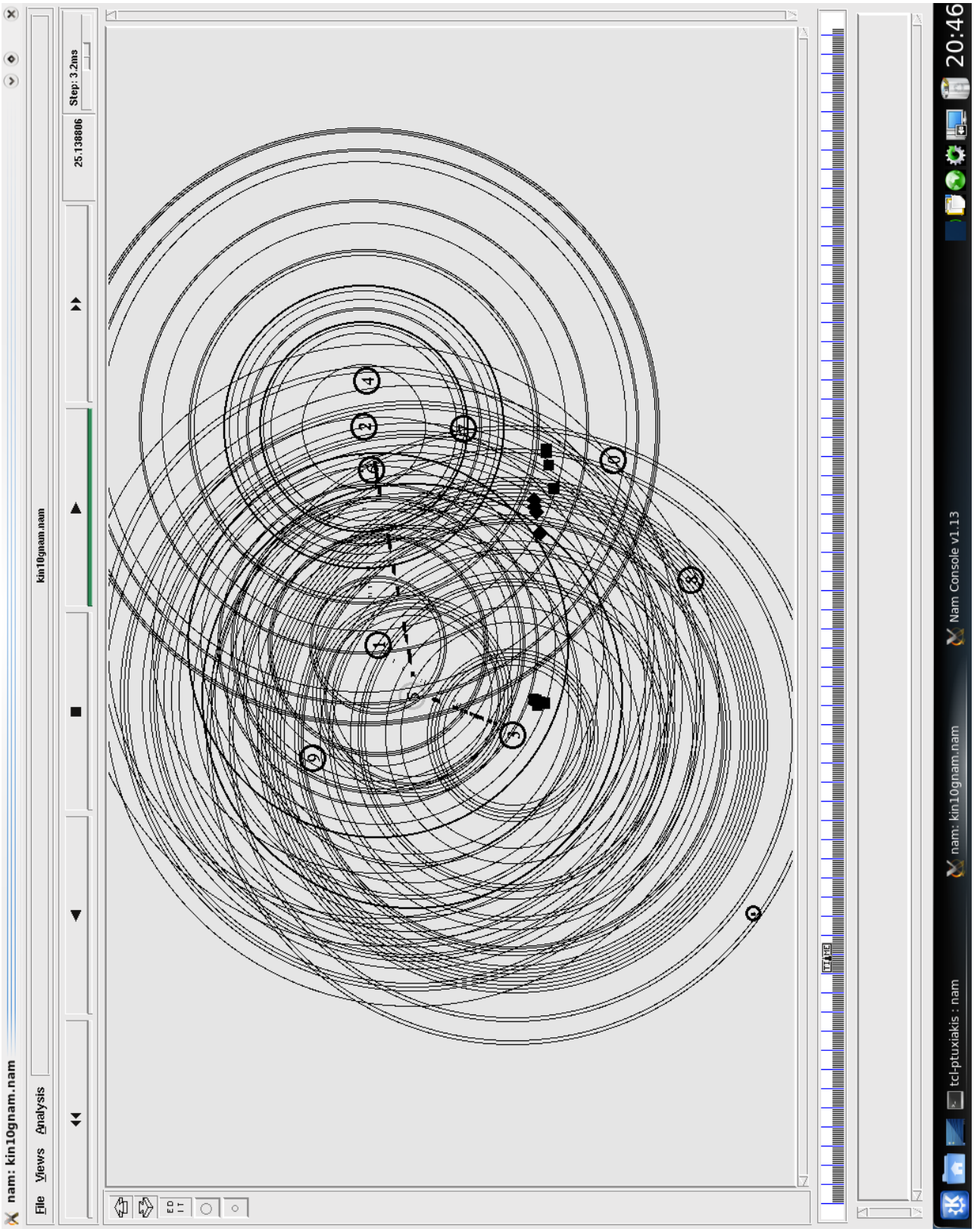
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 10 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)              "/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis10komvon"      ;
set opt(sc)            "/home/kele/tcl-
files/arxeia_kinisis_komvon/kinisi10komvoi"          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              10   ;# number of mobilenodes
set opt(stop)            120.0     ;# simulation time

set opt(adhocRouting)    NOAH  ;# routing protocol
set opt(rp)              DSDV  ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF

LL set mindelay_             50us
```

```
LL set delay_                   25us
LL set bandwidth_               0    ;# not used

Agent/Null set sport_           0
Agent/Null set dport_           0

Agent/CBR set sport_            0
Agent/CBR set dport_            0

Agent/TCPSink set sport_        0
Agent/TCPSink set dport_        0

Agent/TCP set sport_            0
Agent/TCP set dport_            0
Agent/TCP set packetSize_       1460

Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters ##above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

           $opt(mac) set CWMin_          15   ;# IEEE
802.11g uses a lower CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
```

```
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin10gtr.tr w]
set namtrace [open kin10gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)



# create God
 set god_ [create-god $opt(nn)]
# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}

# source connection-pattern and node-movement scripts
```

```
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.2, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 15 ΚΟΜΒΩΝ

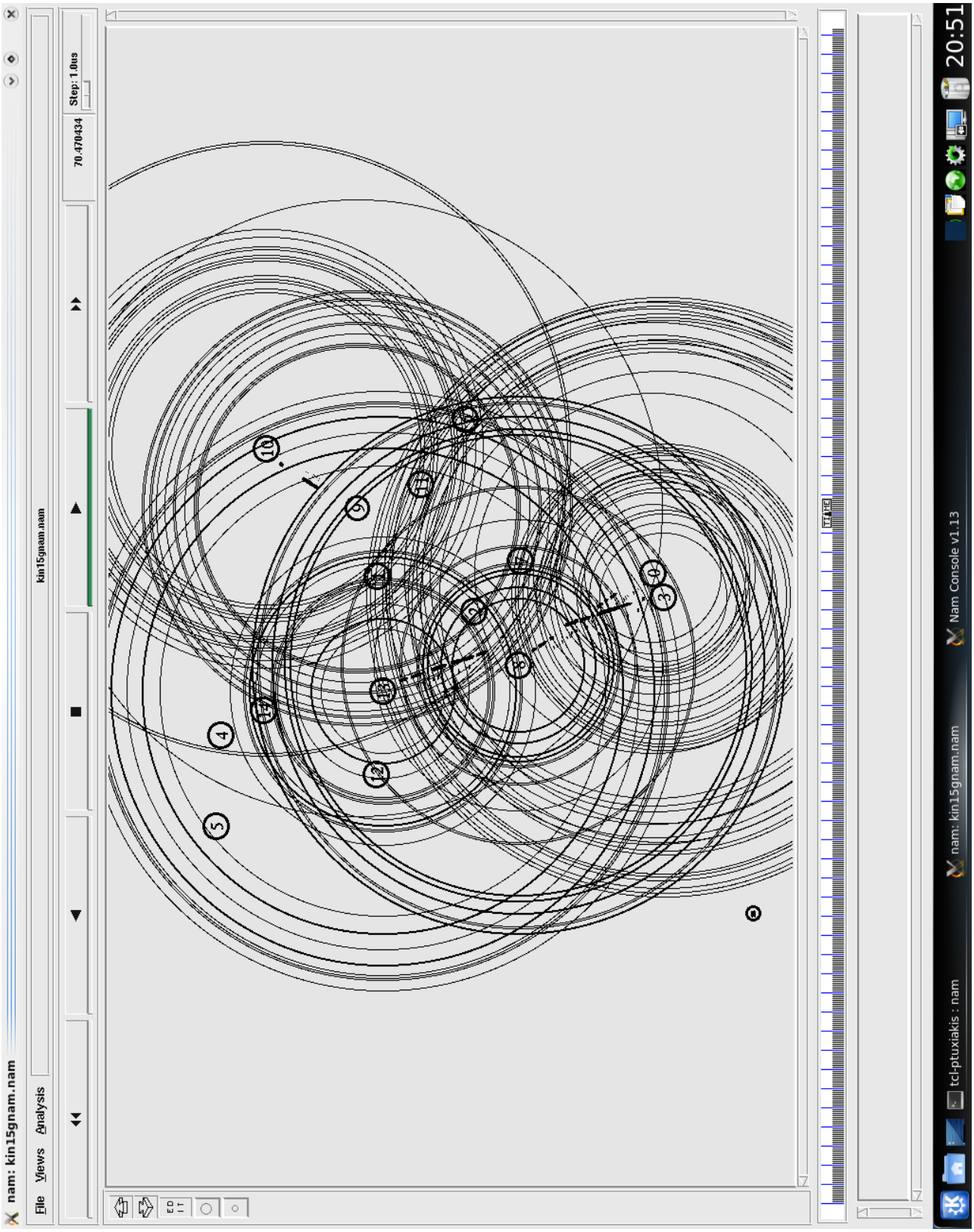
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 15 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)              "/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis15komvon"      ;
set opt(sc)              "/home/kele/tcl-
files/arxeia_kinisis_komvon/kinisi15komvoi"          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              15  ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH      ;# routing protocol
set opt(rp)              DSDV      ;#routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF

LL set mindelay_              50us
```

```
LL set delay_                   25us
LL set bandwidth_               0    ;# not used

Agent/Null set sport_           0
Agent/Null set dport_           0

Agent/CBR set sport_            0
Agent/CBR set dport_            0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_            0
Agent/TCP set dport_            0
Agent/TCP set packetSize_       1460



Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

      $opt(mac) set CWMin_ 15 ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_    54e6
     $opt(mac) set dataRate_       54e6
     $opt(mac) set basicRate_      6e6
     $opt(mac) set PLCPDataRate_   6e6
     $opt(mac) set PreambleLength_      0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
```

```
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin15gtr.tr w]
set namtrace [open kin15gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)
# create God
 set god_ [create-god $opt(nn)]
# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}

# source connection-pattern and node-movement scripts
```

```
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}
# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your
    # scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}

for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.3, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 20 ΚΟΜΒΩΝ

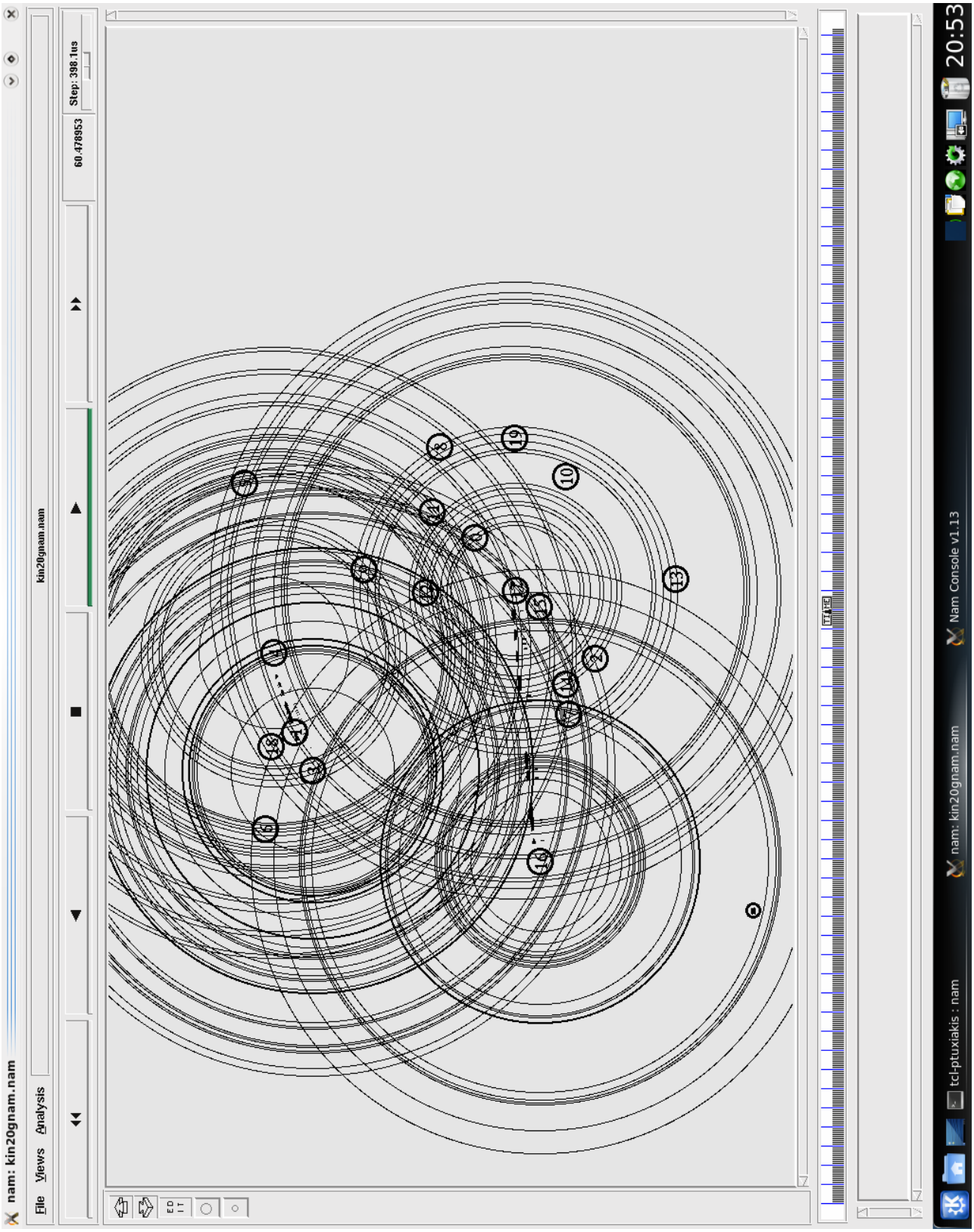
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 20 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp) "/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis20komvon"      ;
set opt(sc) "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi20komvoi"
          ;

set opt(ifqlen)          50                            ;
set opt(seed)            0.0

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0 ;# seed for random number
gen.

set opt(nn)              20      ;# number of mobilenodes
set opt(stop)            120.0   ;# simulation time

set opt(adhocRouting)    NOAH     ;# routing protocol
set opt(rp)              DSDV     ;# routing protocol

# ============ Other default settings
=======================
```

```
set AgentTrace             ON
set RouterTrace            OFF
set MacTrace               OFF

LL set mindelay_               50us
LL set delay_                  25us
LL set bandwidth_              0    ;# not used

Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460

Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15 ;# IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
```

```
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin20gtr.tr w]
set namtrace [open kin20gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)

# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]          $node_($i)
random-motion 0          ;# disable random motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
```

```
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.4, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 25 ΚΟΜΒΩΝ

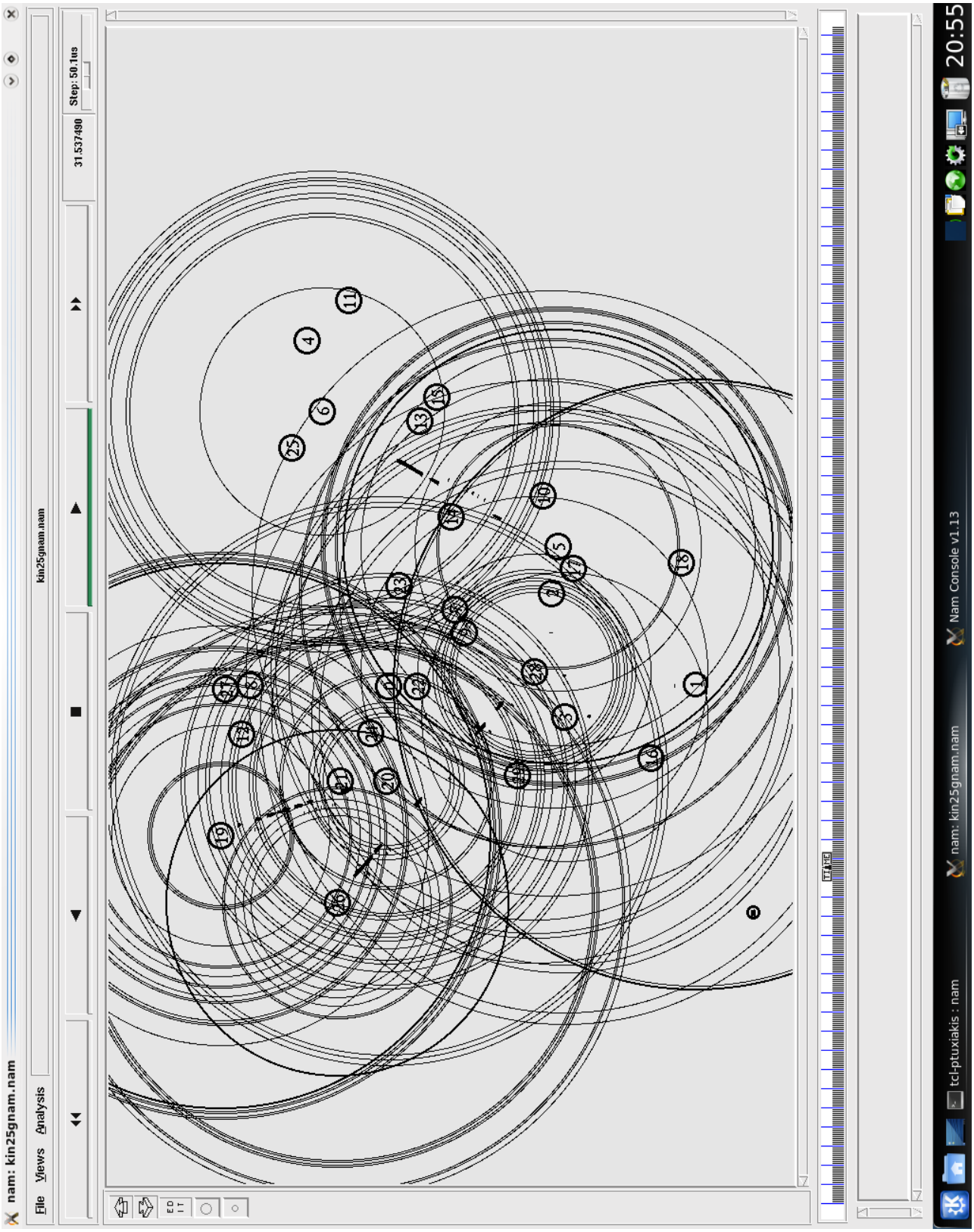
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 25 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)       500               ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp) "/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis25komvon"       ;
set opt(sc) "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi25komvoi"
          ;


set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              25    ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH      ;# routing protocol
set opt(rp)              DSDV      ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF
```

```
LL set mindelay_         50us
LL set delay_            25us
LL set bandwidth_        0    ;# not used

Agent/Null set sport_         0
Agent/Null set dport_         0

Agent/CBR set sport_     0
Agent/CBR set dport_     0

Agent/TCPSink set sport_      0
Agent/TCPSink set dport_      0

Agent/TCP set sport_          0
Agent/TCP set dport_          0
Agent/TCP set packetSize_     1460

Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
```

```
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin25gtr.tr w]
set namtrace [open kin25gnam.nam w]

$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)

# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
```

```
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.5, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 30 ΚΟΜΒΩΝ

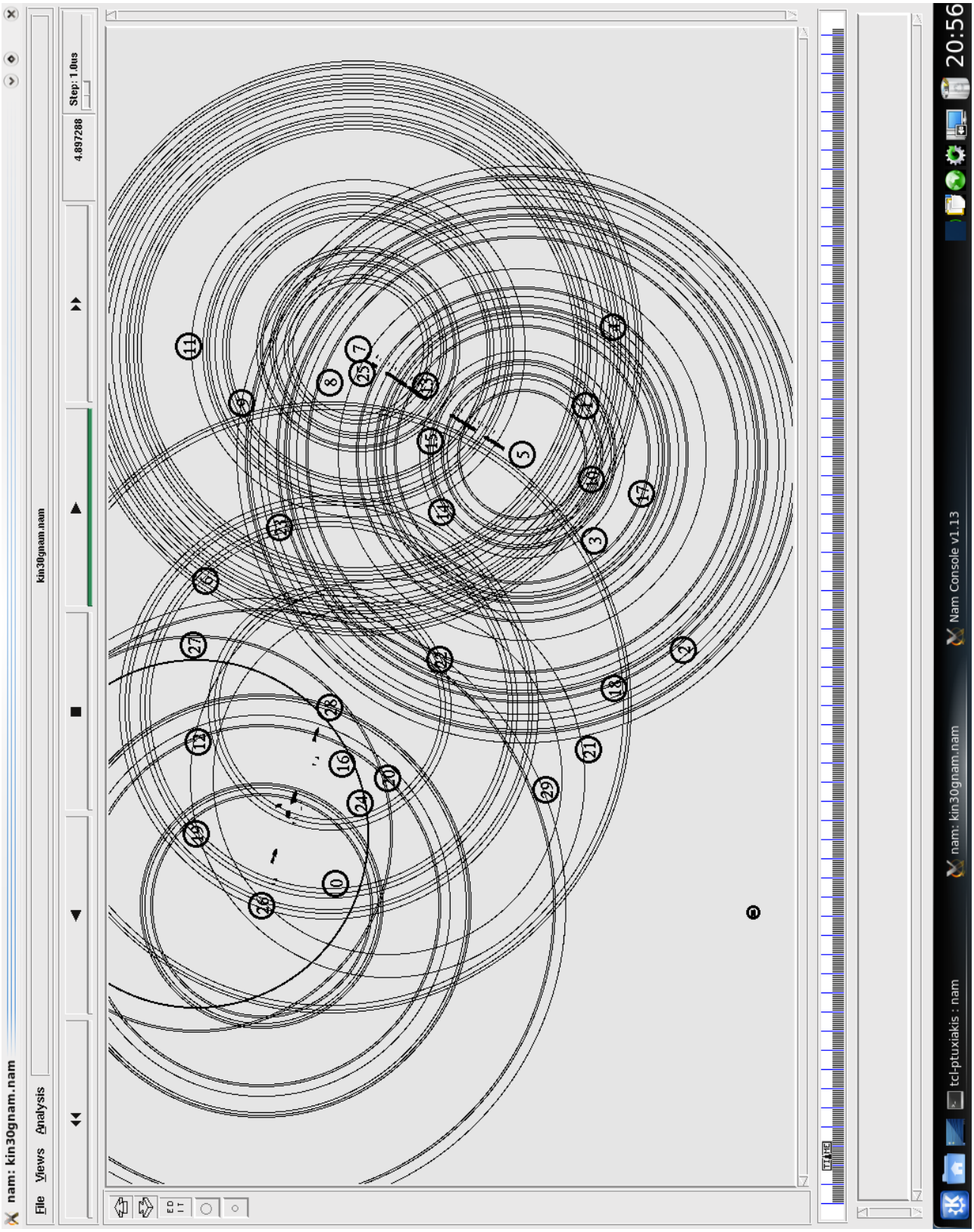
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 30 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp) "/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis30komvon"       ;
set opt(sc) "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi30komvoi"
          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              30   ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH     ;# routing protocol
set opt(rp)              DSDV     ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace               OFF
set MacTrace             OFF

LL set mindelay_         50us
```

```
LL set delay_            25us
LL set bandwidth_        0    ;# not used

Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460

Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1


# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
```

```
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin30gtr.tr w]
set namtrace [open kin30gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
           -macType $opt(mac) \
           -ifqType $opt(ifq) \
           -ifqLen $opt(ifqlen) \
           -antType $opt(ant) \
           -propType $opt(prop) \
           -phyType $opt(netif) \
           -topoInstance $topo \
           -agentTrace ON \
           -routerTrace ON \
           -macTrace ON \
           -movementTrace OFF \
           -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
```

```
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
 # 20 defines the node size in nam, must adjust it
according to your scenario
 # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.6, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 35 ΚΟΜΒΩΝ

Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 35 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)"/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis35komvon";
set opt(sc)"/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi35komvoi"
          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              35   ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH  ;# routing protocol
set opt(rp)              DSDV  ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF

LL set mindelay_         50us
LL set delay_            25us
LL set bandwidth_        0    ;# not used
```

```
Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460


Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
```

```
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin35gtr.tr w]
set namtrace [open kin35gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
```

```
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```

Εικόνα 10.7, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 40 ΚΟΜΒΩΝ

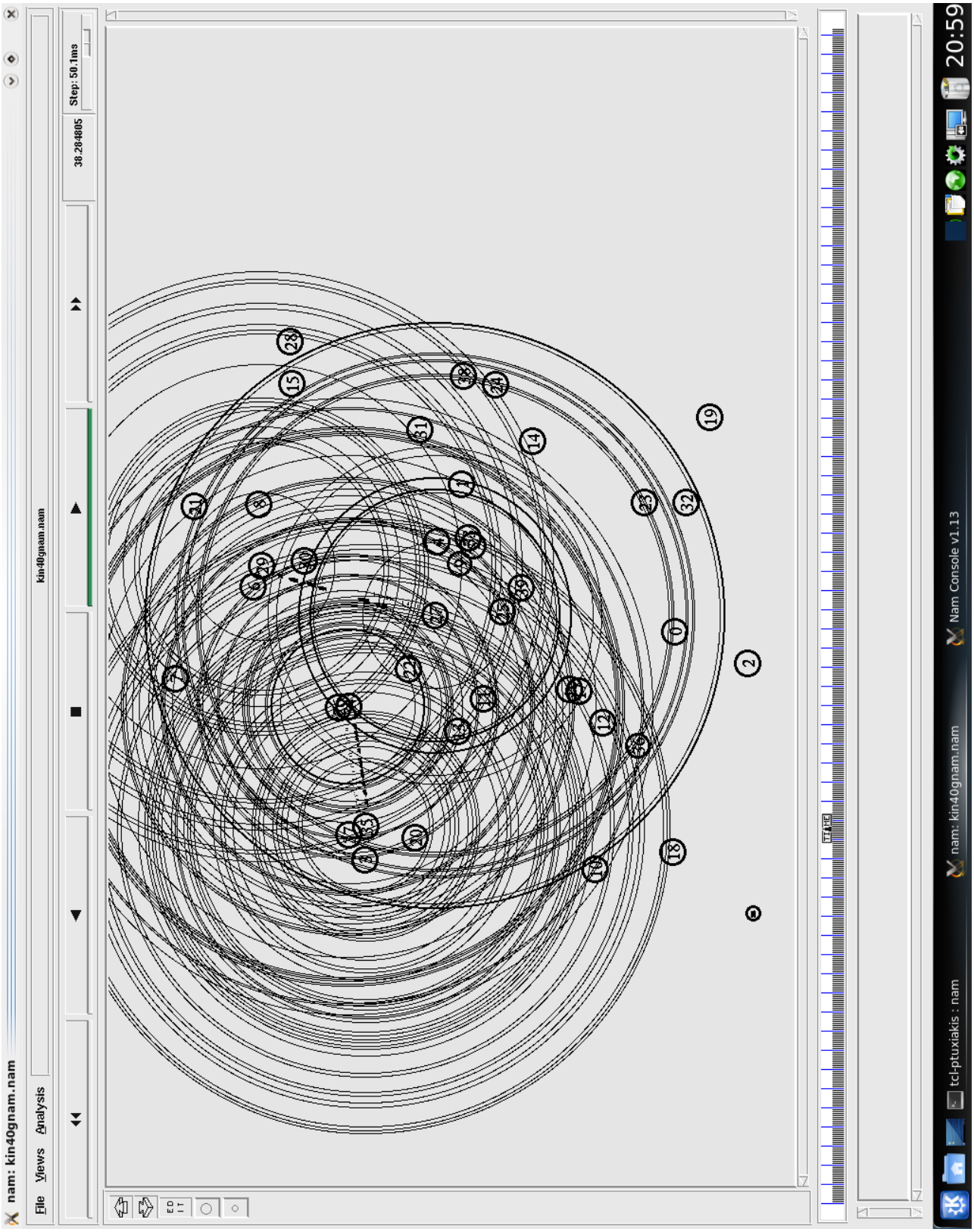
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 40 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)"/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis40komvon";
set opt(sc) "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi40komvoi"
          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              40   ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH  ;# routing protocol
set opt(rp)              DSDV  ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF

LL set mindelay_              50us
LL set delay_                 25us
LL set bandwidth_             0    ;# not used
```

```
Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460


Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_    54e6
     $opt(mac) set dataRate_       54e6
     $opt(mac) set basicRate_      6e6
     $opt(mac) set PLCPDataRate_   6e6
     $opt(mac) set PreambleLength_      0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
```

```
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin40gtr.tr w]
set namtrace [open kin40gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
```

```
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.8, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 45 ΚΟΜΒΩΝ

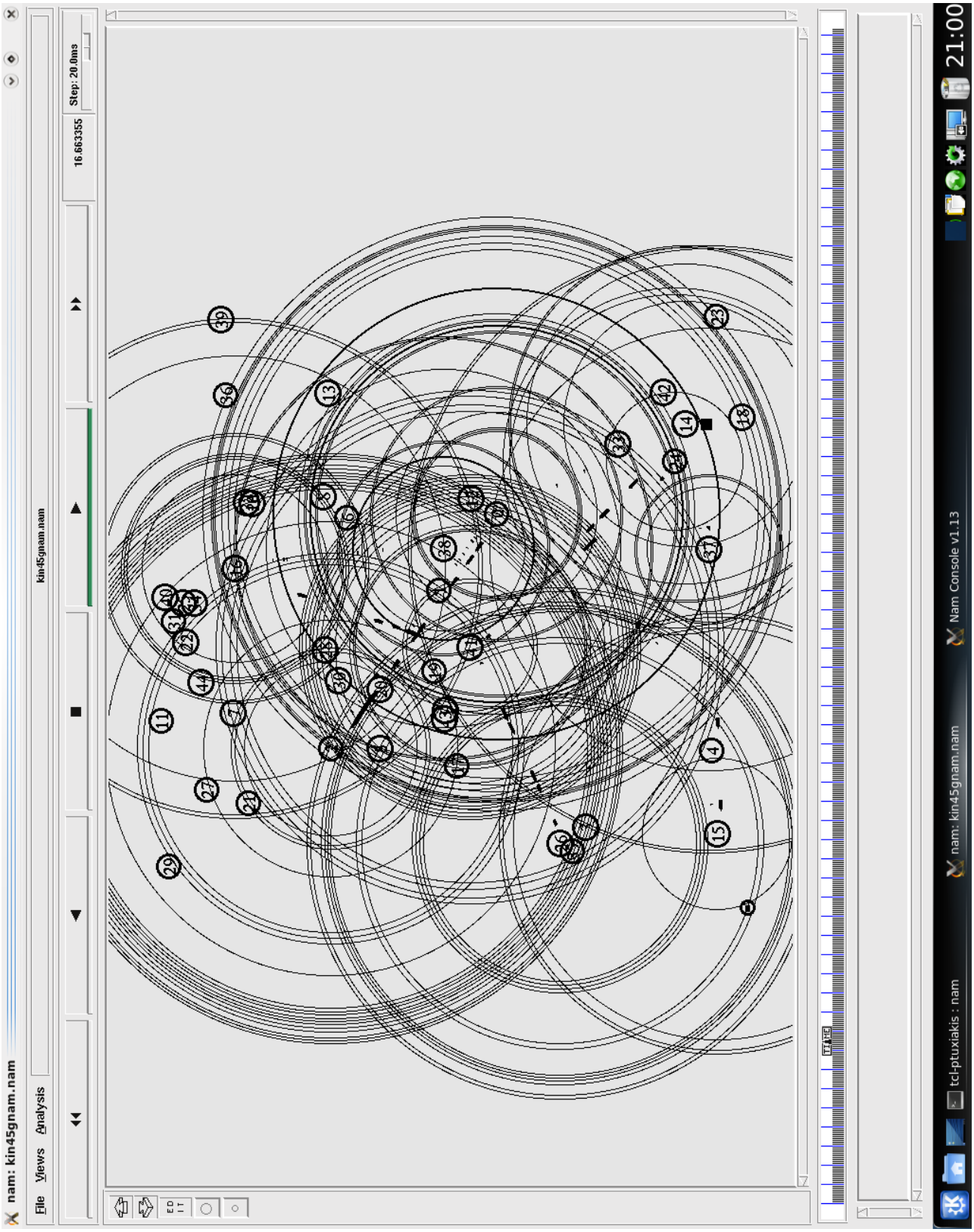
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 45 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)        500              ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)"/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis45komvon";
set opt(sc)"/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi45komvoi"
     ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              45   ;# number of mobilenodes
set opt(stop)            120.0 ;# simulation time

set opt(adhocRouting)    NOAH  ;# routing protocol
set opt(rp)              DSDV  ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace               OFF
set MacTrace             OFF

LL set mindelay_              50us
LL set delay_                 25us
LL set bandwidth_             0    ;# not used
```

```
Agent/Null set sport_          0
Agent/Null set dport_          0

Agent/CBR set sport_           0
Agent/CBR set dport_           0

Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0

Agent/TCP set sport_           0
Agent/TCP set dport_           0
Agent/TCP set packetSize_      1460



Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_          54e6
     $opt(mac) set dataRate_        54e6
     $opt(mac) set basicRate_       6e6
     $opt(mac) set PLCPDataRate_    6e6
     $opt(mac) set PreambleLength_       0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
}
if {$opt(seed) > 0} {
```

```
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]

# Create topography object
set topo   [new Topography]

set tracefd  [open kin45gtr.tr w]
set namtrace [open kin45gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
```

```
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.9, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

ΣΕΝΑΡΙΟ 50 ΚΟΜΒΩΝ

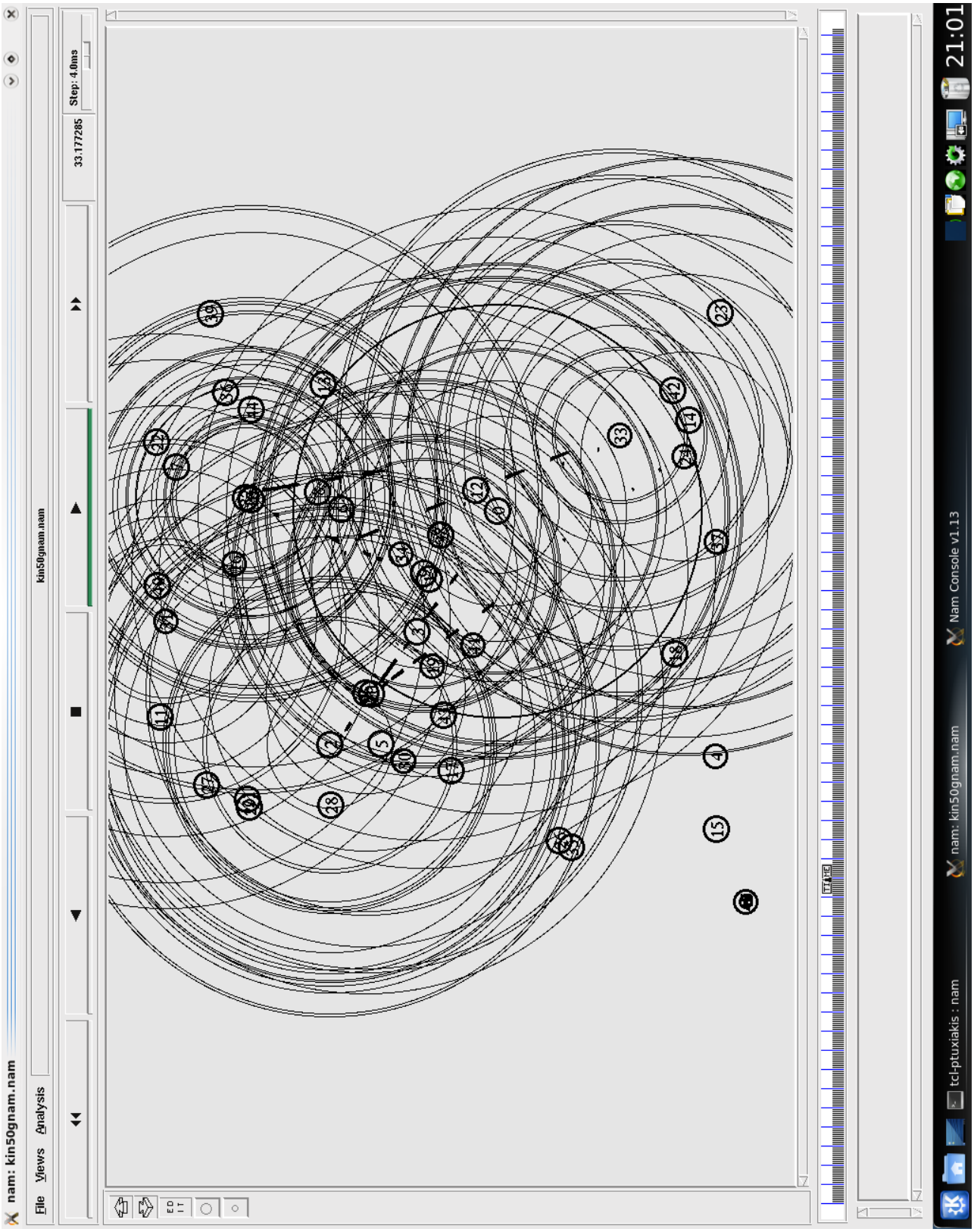
Το ακόλουθο σενάριο προσομοιώνει ένα ασύρματο δίκτυο που κάνει χρήση του πρωτοκόλλου 802.11g, αφορά 50 ασύρματους κινητούς κόμβους η κίνηση των οποίων και η μεταξύ τους συνδέσεις είναι τυχαίες. Το πρωτόκολλο δρομολόγησης είναι το DSDV, η διάρκεια της προσομοίωσης είναι 120 δευτερόλεπτα. Οι κόμβοι κινούνται σε μια επιφάνεια 500m*500m.

```
set opt(chan)  Channel/WirelessChannel ;# channel type
set opt(prop)  Propagation/TwoRayGround;# radio-
propagation model
set opt(netif) Phy/WirelessPhy         ;# network
interface type
set opt(mac)   Mac/802_11              ;# MAC type
set opt(ifq)   Queue/DropTail/PriQueue ;# interface queue
type
set opt(ll)    LL                      ;# link layer type
set opt(ant)   Antenna/OmniAntenna     ;# antenna model

set opt(x)       500               ;# X dimension of
topography
set opt(y)       500               ;# Y dimension of
topography

set opt(cp)"/home/kele/tcl-
files/arxeia_sundeseon_komvon/tuxaies_sundeseis45komvon"       ;
set opt(sc) "/home/kele/tcl-files/arxeia_kinisis_komvon/kinisi45komvoi"
          ;

set opt(ifqlen)          50   ;# max packet in ifq

set opt(seed)            0.0  ;# seed for random number
gen.

set opt(nn)              50   ;# number of mobilenodes
set opt(stop)            120.0

set opt(adhocRouting)    NOAH  ;# routing protocol
set opt(rp)              DSDV  ;# routing protocol

# ============ Other default settings
=======================

set AgentTrace           ON
set RouterTrace          OFF
set MacTrace             OFF

LL set mindelay_             50us
```

```
LL set delay_                   25us
LL set bandwidth_               0    ;# not used

Agent/Null set sport_           0
Agent/Null set dport_           0

Agent/CBR set sport_            0
Agent/CBR set dport_            0

Agent/TCPSink set sport_        0
Agent/TCPSink set dport_        0

Agent/TCP set sport_            0
Agent/TCP set dport_            0
Agent/TCP set packetSize_       1460



Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5
meters #above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to
make
# it work like the 914MHz Lucent WaveLAN DSSS radio
interface
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e6
Phy/WirelessPhy set L_ 1.0

     $opt(mac) set CWMin_ 15  ;#IEEE 802.11g uses a lower
CWMin...
     $opt(netif) set bandwidth_    54e6
     $opt(mac) set dataRate_       54e6
     $opt(mac) set basicRate_      6e6
     $opt(mac) set PLCPDataRate_   6e6
     $opt(mac) set PreambleLength_      0.000016

# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
     puts "No X-Y boundary values given for wireless
topology\n"
```

```
}
if {$opt(seed) > 0} {
     puts "Seeding Random number generator with
$opt(seed)\n"
     ns-random $opt(seed)
}

# create simulator instance
set ns_   [new Simulator]
set chan [new $opt(chan)]


# Create topography object
set topo   [new Topography]

set tracefd  [open kin50gtr.tr w]
set namtrace [open kin50gnam.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$topo load_flatgrid $opt(x) $opt(y)


# create God
 set god_ [create-god $opt(nn)]

# configure node, please note the change below.
$ns_ node-config -adhocRouting $opt(rp) \
                 -llType $opt(ll) \
          -macType $opt(mac) \
          -ifqType $opt(ifq) \
          -ifqLen $opt(ifqlen) \
          -antType $opt(ant) \
          -propType $opt(prop) \
          -phyType $opt(netif) \
          -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace ON \
          -movementTrace OFF \
          -channel $chan

for {set i 0} {$i < $val(nn) } {incr i} {
     set node_($i) [$ns_ node]
     $node_($i) random-motion 0         ;# disable random
motion
}

for {set i 0} {$i < $opt(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 10
}
```

```
# source connection-pattern and node-movement scripts
if { $opt(cp) == "" } {
     puts "*** NOTE: no connection pattern specified."
        set opt(cp) "none"
} else {
     puts "Loading connection pattern..."
     source $opt(cp)
}
if { $opt(sc) == "" } {
     puts "*** NOTE: no scenario file specified."
        set opt(sc) "none"
} else {
     puts "Loading scenario file..."
     source $opt(sc)
     puts "Load complete..."
}

# Define initial node position in nam
for {set i 0} {$i < $opt(nn)} {incr i} {
    # 20 defines the node size in nam, must adjust it
according to your# scenario
    # The function must be called after mobility model is
defined
    $ns_ initial_node_pos $node_($i) 20
}
# Tell all nodes when the simulation ends
for {set i 0} {$i < $opt(nn) } {incr i} {
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
}
$ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ;
$ns_ halt"


proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}

# informative headers for CMUTracefile
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp
$opt(adhocRouting)"
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed
$opt(seed)"
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

puts "Starting Simulation..."
$ns_ run
```



Εικόνα 10.10, Τυχαίο στιγμιότυπο του αρχείου προσομοίωσης

Αρχείο τυχαίας κίνησης για το σενάριο των 5 κόμβων

Το αρχείο αυτό δημιουργείται αυτόματα. Το πώς γίνεται αυτό το αναφέρρεται στο κεφαλαιο 8 (8.2.2 Αρχεία Κίνησης - Συνδέσεων, σελ **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**)

```
#
# nodes: 5, pause: 2.00, max speed: 10.00, max x: 500.00, max y:
500.00
#
$node_(0) set X_ 37.410897035493
$node_(0) set Y_ 468.332352678555
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 483.666301263168
$node_(1) set Y_ 330.366702917096
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 125.055930463512
$node_(2) set Y_ 474.327833894684
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 294.593874778488
$node_(3) set Y_ 239.768452796116
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 451.981698864185
$node_(4) set Y_ 92.483266777480
$node_(4) set Z_ 0.000000000000
$god_ set-dist 0 1 16777215
$god_ set-dist 0 2 1
$god_ set-dist 0 3 16777215
$god_ set-dist 0 4 16777215
$god_ set-dist 1 2 16777215
$god_ set-dist 1 3 1
$god_ set-dist 1 4 1
$god_ set-dist 2 3 16777215
$god_ set-dist 2 4 16777215
$god_ set-dist 3 4 1
$ns_ at 2.000000000000 "$node_(0) setdest 426.789343071213
492.814174156019 9.564633015414"
$ns_ at 2.000000000000 "$node_(1) setdest 456.164154895149
393.257293686624 2.673068269156"
$ns_ at 2.000000000000 "$node_(2) setdest 366.462119441508
359.531844799086 9.633415132703"
$ns_ at 2.000000000000 "$node_(3) setdest 337.978617241502
409.888659663694 3.432128649178"
$ns_ at 2.000000000000 "$node_(4) setdest 92.960430503769
222.653989869177 1.956051538433"
$ns_ at 5.712022973303 "$god_ set-dist 0 1 3"
$ns_ at 5.712022973303 "$god_ set-dist 0 3 2"
$ns_ at 5.712022973303 "$god_ set-dist 0 4 3"
$ns_ at 5.712022973303 "$god_ set-dist 1 2 2"
$ns_ at 5.712022973303 "$god_ set-dist 2 3 1"
$ns_ at 5.712022973303 "$god_ set-dist 2 4 2"
$ns_ at 7.352763015916 "$god_ set-dist 1 4 2"
$ns_ at 13.789370169065 "$god_ set-dist 0 1 2"
$ns_ at 13.789370169065 "$god_ set-dist 0 3 1"
$ns_ at 13.789370169065 "$god_ set-dist 0 4 2"
$ns_ at 13.994644624946 "$god_ set-dist 1 2 1"
```

```
$ns_ at 22.457072362865 "$god_ set-dist 0 1 1"
$ns_ at 27.678751274516 "$node_(1) setdest 456.164154895149
393.257293686624 0.000000000000"
$ns_ at 29.678751274516 "$node_(1) setdest 217.202299315966
233.454313259222 8.595569342812"
$ns_ at 29.748290991685 "$node_(2) setdest 366.462119441508
359.531844799086 0.000000000000"
$ns_ at 30.773357290496 "$god_ set-dist 2 4 1"
$ns_ at 31.748290991685 "$node_(2) setdest 54.721899652424
169.833059614821 8.551803116867"
$ns_ at 35.802264050550 "$god_ set-dist 1 4 1"
$ns_ at 37.759565217535 "$god_ set-dist 3 4 2"
$ns_ at 42.790621302124 "$node_(0) setdest 426.789343071213
492.814174156019 0.000000000000"
$ns_ at 44.790621302124 "$node_(0) setdest 246.949172663835
488.232698769402 1.015842976350"
$ns_ at 45.254436422887 "$god_ set-dist 0 2 2"
$ns_ at 53.153428218801 "$node_(3) setdest 337.978617241502
409.888659663694 0.000000000000"
$ns_ at 53.402386627738 "$god_ set-dist 0 1 2"
$ns_ at 53.402386627738 "$god_ set-dist 0 4 3"
$ns_ at 55.153428218801 "$node_(3) setdest 279.568570002509
366.021654599464 0.283758765790"
$ns_ at 60.142009527209 "$god_ set-dist 0 2 3"
$ns_ at 60.142009527209 "$god_ set-dist 2 3 2"
$ns_ at 63.122879352598 "$node_(1) setdest 217.202299315966
233.454313259222 0.000000000000"
$ns_ at 65.122879352598 "$node_(1) setdest 18.509546725542
116.450133435807 0.855785852299"
$ns_ at 71.053161524626 "$god_ set-dist 2 4 2"
$ns_ at 74.420151863611 "$node_(2) setdest 54.721899652424
169.833059614821 0.000000000000"
$ns_ at 76.420151863611 "$node_(2) setdest 121.144546707642
368.505535099924 4.567312467844"
$ns_ at 80.752320112249 "$god_ set-dist 2 4 1"
$ns_ at 101.716443075732 "$god_ set-dist 0 4 2"
$ns_ at 101.716443075732 "$god_ set-dist 3 4 1"
$ns_ at 106.325684354666 "$god_ set-dist 0 2 2"
$ns_ at 106.325684354666 "$god_ set-dist 2 3 1"
#
# Destination Unreachables: 6
#
# Route Changes: 26
#
# Link Changes: 15
#
# Node | Route Changes | Link Changes
#    0 |            13 |            4
#    1 |             8 |            5
#    2 |            12 |            8
#    3 |             7 |            6
#    4 |            12 |            7

#
```

Αρχείο τυχαίων συνδέσεων για το σενάριο των 5 κόμβων

Το αρχείο αυτό δημιουργείται επίσης αυτόματα. Το πώς γίνεται αυτό το αναφέρεται στο κεφάλαιο στο κεφαλαιο 8 (8.2.2 Αρχεία Κίνησης-Συνδέσεων, σελ **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**)

```
#
# nodes: 5, max conn: 30, send rate: 0.0, seed: 0
#
#
# 0 connecting to 1 at time 138.93227706613592
#
set tcp_(0) [$ns_ create-connection  TCP $node_(0) TCPSink $node_(1)
0]
$tcp_(0) set window_ 32
$tcp_(0) set packetSize_ 512
set ftp_(0) [$tcp_(0) attach-source FTP]
$ns_ at 138.93227706613592 "$ftp_(0) start"
#
# 1 connecting to 2 at time 56.321628837064665
#
set tcp_(1) [$ns_ create-connection  TCP $node_(1) TCPSink $node_(2)
0]
$tcp_(1) set window_ 32
$tcp_(1) set packetSize_ 512
set ftp_(1) [$tcp_(1) attach-source FTP]
$ns_ at 56.321628837064665 "$ftp_(1) start"
#
# 1 connecting to 3 at time 152.72372065704491
#
set tcp_(2) [$ns_ create-connection  TCP $node_(1) TCPSink $node_(3)
0]
$tcp_(2) set window_ 32
$tcp_(2) set packetSize_ 512
set ftp_(2) [$tcp_(2) attach-source FTP]
$ns_ at 152.72372065704491 "$ftp_(2) start"
#
# 2 connecting to 3 at time 127.10943726222472
#
set tcp_(3) [$ns_ create-connection  TCP $node_(2) TCPSink $node_(3)
0]
$tcp_(3) set window_ 32
$tcp_(3) set packetSize_ 512
set ftp_(3) [$tcp_(3) attach-source FTP]
$ns_ at 127.10943726222472 "$ftp_(3) start"
#
# 3 connecting to 4 at time 76.627088681155385
#
set tcp_(4) [$ns_ create-connection  TCP $node_(3) TCPSink $node_(4)
0]
$tcp_(4) set window_ 32
$tcp_(4) set packetSize_ 512
set ftp_(4) [$tcp_(4) attach-source FTP]
$ns_ at 76.627088681155385 "$ftp_(4) start"
#
# 3 connecting to 0 at time 156.76439419238102
```

```
#
set tcp_(5) [$ns_ create-connection  TCP $node_(3) TCPSink $node_(0)
0]
$tcp_(5) set window_ 32
$tcp_(5) set packetSize_ 512
set ftp_(5) [$tcp_(5) attach-source FTP]
$ns_ at 156.76439419238102 "$ftp_(5) start"
#
# 4 connecting to 0 at time 42.007990079935631
#
set tcp_(6) [$ns_ create-connection  TCP $node_(4) TCPSink $node_(0)
0]
$tcp_(6) set window_ 32
$tcp_(6) set packetSize_ 512
set ftp_(6) [$tcp_(6) attach-source FTP]
$ns_ at 42.007990079935631 "$ftp_(6) start"
#
#Total sources/connections: 5/7
#
```

Βιβλιογραφία

Βιβλία

1. William Stallings, «Ασύρματες Επικοινωνίες & Δίκτυα», Εκδόσεις Τζιόλα, 2007.
2. Nikopolitidis, Obaidat, Papadimitriou and Pomportsis, «Ασύρματα Δίκτυα», Εκδόσεις Κλειδάριθμος, 2003.
3. Πανέτσος Σπυριδον, «Επικοινωνίες & Δίκτυα Υπολογιστών», Εκδόσεις Τζιόλα, 2007
4. Wayne Lewis, «Lan Switching and Wireless», Cisco Press 2008
5. Αλεξόπουλος Άρης, «Τηλεπικοινωνίες & Δίκτυα Υπολογιστών 6η Έκδοση», Αυτοεκδοση , 2003
6. Steve McQuerry, «CCNA Αυτοδιδασκαλία», Εκδόσεις Κλειδαριθμός, 2005
7. Πρεβές Ν., «**ΑΣΦΑΛΕΙΑ ΚΑΙ ΑΠΟΔΟΣΗ ΤΩΝ ΠΡΩΤΟΚΟΛΛΩΝ TCP/IP**», Εκδόσεις ΝΕΩΝ ΤΕΧΝΟΛΟΓΙΩΝ, 2009

Links

1. **http://www.isi.edu/nsnam/ns/tutorial/index.html**
2. **http://en.wikipedia.org/wiki/802.11**
3. **http://en.wikipedia.org/wiki/IEEE_802.11g**
4. **http://standards.ieee.org/getieee802/download/802.11g-2003.pdf**
5. **http://www.joshuarobinson.net/docs/802_11.html**
6. **http://nile.wpi.edu/NS/**
7. **ftp://gaia.cs.umass.edu/pub/ffm_in_ns.pdf**
8. **http://nsnam.isi.edu/nsnam/index.php/Installing_ns2.31_on_Ubuntu7.04**
9. **http://imtl.skku.ac.kr/~hjlim99/ns2/%5B1_%B8%DE%B4%BA%BE%F3%5D/%B3%BB%BA%CE%B5%BF%C0%DB/ns2-200407/ns2-200407-s3_appendix.pdf**
10. **http://www.ece.rice.edu/~jpr/ns/docs/ns-802_11b.html**
11. **http://www.winlab.rutgers.edu/~zhibinwu/html/network_simulator_2.html**