ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν
Πανεπιστήμιον Αθηνών

Δίκτυα Επικοινωνιών

Ενότητα 3: Επίπεδο Μεταφοράς

Διδάσκων: Λάζαρος Μεράκος

Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Εθνικό & Καποδιστριακό Πανεπιστήμιο Αθηνών

Δίκτυα Επικοινωνιών

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Εθνικό & Καποδιστριακό Πανεπιστήμιο Αθηνών

Συνιστώμενο Βιβλίο:
Computer Networking: A Top-Down Approach, by Kurose & Ross, Addison-Wesley

Ελληνική Μετάφραση:
Εκδόσεις : Μ. Γκιούρδας

Θεματικές Ενότητες (ΘΕ) μαθήματος:

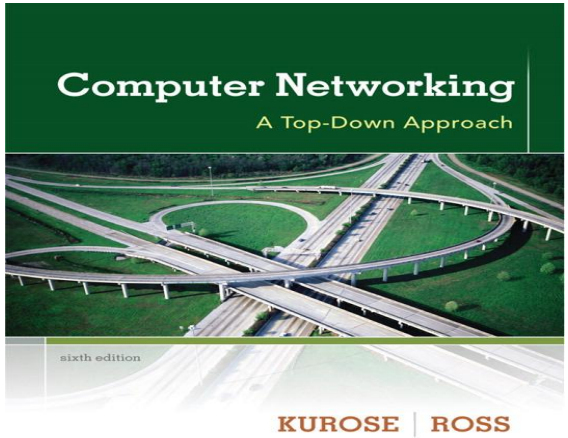
ΘΕ1: Εισαγωγή
(Κεφ. 1 του βιβλίου)

ΘΕ2: Επίπεδο Εφαρμογής
(Κεφ. 2 του βιβλίου)

ΘΕ3: Επίπεδο Μεταφοράς
(Κεφ. 3 του βιβλίου)

ΘΕ4: Επίπεδο Δικτύου
(Κεφ. 4 του βιβλίου)

ΘΕ5: Επίπεδο Ζεύξης: Ζεύξεις, Δίκτυα Πρόσβασης, Δίκτυα Τοπικής Περιοχής
(Κεφ. 5 του βιβλίου)



Οι περισσότερες από τις διαφάνειες αυτής της ενότητας αποτελούν προσαρμογή και απόδοση στα ελληνικά των διαφανειών που συνοδεύουν το βιβλίο Computer Networking : A Top-Down Approach, J.F Kurose and K.W. Ross, 6/E, Addison-Wesley.

Προσαρμογή και επιμέλεια της απόδοσης των πρωτότυπων διαφανειών στα ελληνικά : Λάζαρος Μεράκος

Κεφάλαιο 3: Επίπεδο Μεταφοράς

Οι στόχοι μας:

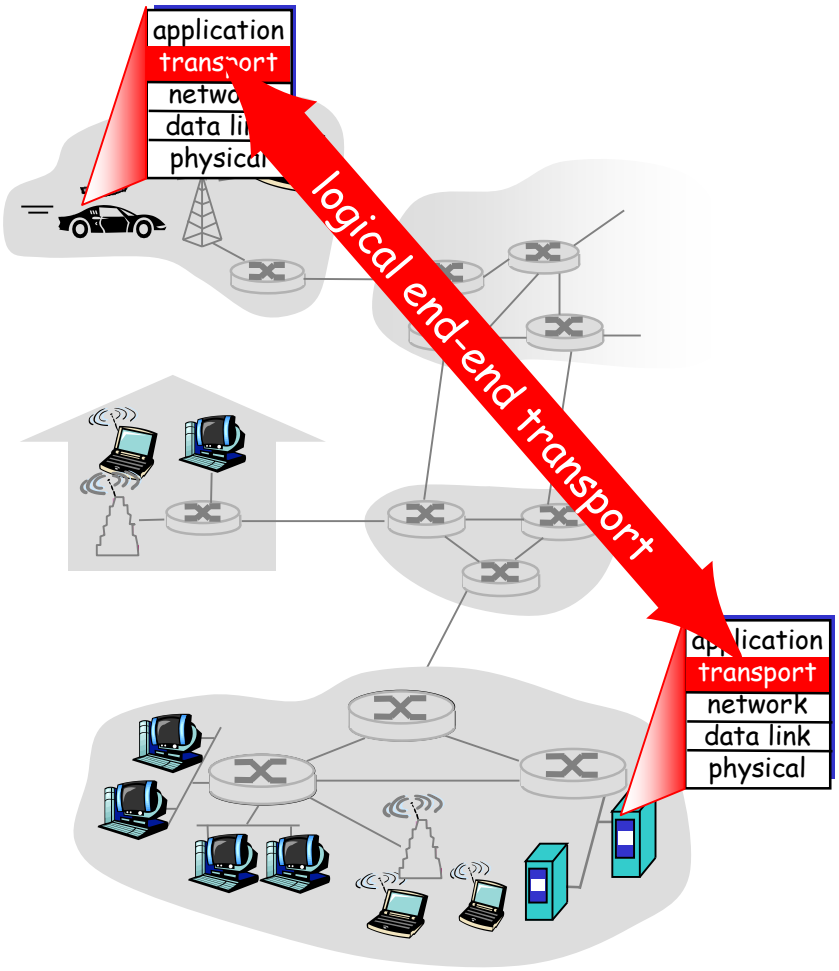
- Κατανόηση των αρχών πίσω από τις υπηρεσίες του επιπέδου μεταφοράς:
  - Πολύπλεξη/αποπολύπλεξη(multiplexing/demultiplexing)
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής (flow control)
  - Έλεγχος συμφόρησης (congestion control)
- Να μάθουμε για τα πρωτόκολλα επιπέδου μεταφοράς του Διαδικτύου:
  - UDP: ασυνδεσμική μεταφορά
  - TCP: συνδεσμική μεταφορά
  - Έλεγχος συμφόρησης τουTCP

Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

Υπηρεσίες και πρωτόκολλα μεταφοράς

- παρέχουν λογική επικοινωνία μεταξύ διεργασιών εφαρμογών που τρέχουν σε διαφορετικά τερματικά συστήματα
- τα πρωτόκολλα μεταφοράς τρέχουν στα τερματικά συστήματα
  - πλευρά αποστολής: σπάει τα μηνύματα της εφαρμογής σε τμήματα, τα περνάει στο επίπεδο δικτύου
  - πλευρά λήψης: ανασύνθεση των τμημάτων σε μηνύματα, τα περνάει στο επίπεδο εφαρμογής
- Περισσότερα από ένα πρωτόκολλα μεταφοράς διαθέσιμα στις εφαρμογές
  - Διαδίκτυο: TCP και UDP



Επίπεδο Μεταφοράς / Επίπεδο Δικτύου

- επίπεδο δικτύου: λογική επικοινωνία μεταξύ υπολογιστών
- επίπεδο μεταφοράς: λογική επικοινωνία μεταξύ διεργασιών
  - στηρίζεται, συμπληρώνει τις υπηρεσίες του επιπέδου δικτύου

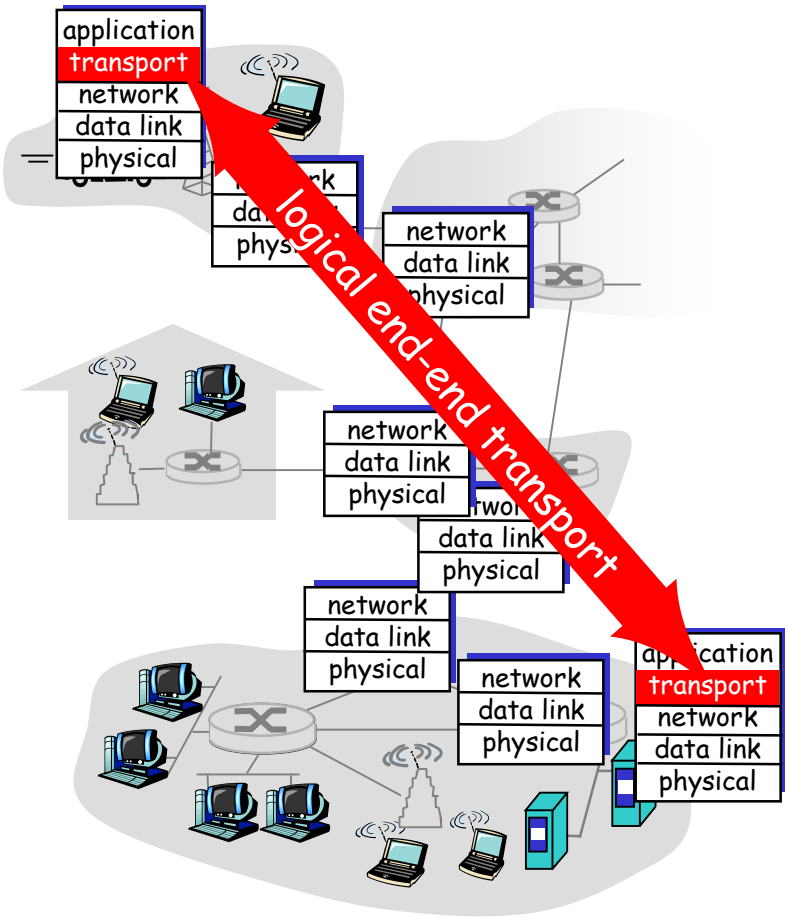
Αναλογία :

12 παιδιά στο σπίτι της Άννας στέλνουν γράμματα σε 12 παιδιά στο σπίτι του Βασίλη

- διεργασίες = παιδιά
- μηνύματα εφαρμογών = γράμματα σε φακέλους
- υπολογιστές = σπίτια
- πρωτόκολλο μεταφοράς = Άννα και Βασίλης (παιδί που αναλαμβάνει το μοίρασμα της αλληλογραφίας σε κάθε σπίτι)
- πρωτόκολλο επιπέδου δικτύου = ταχυδρομική υπηρεσία

Πρωτόκολλα Επιπέδου Μεταφοράς στο Διαδίκτυο

- αξιόπιστη, σε ορθή σειρά μεταφορά (TCP)
  - έλεγχος συμφόρησης (congestion control)
  - έλεγχος ροής (flow control)
  - εγκαθίδρυση σύνδεσης
- μη αξιόπιστη, εκτός σειράς παράδοση: UDP
  - καμία επέκταση της "βέλτιστης προσπάθειας" (best effort) του IP
- υπηρεσίες που δεν είναι διαθέσιμες:
  - εγγυήσεις ως προς την καθυστέρηση
  - εγγυήσεις ως προς το εύρος ζώνης



Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

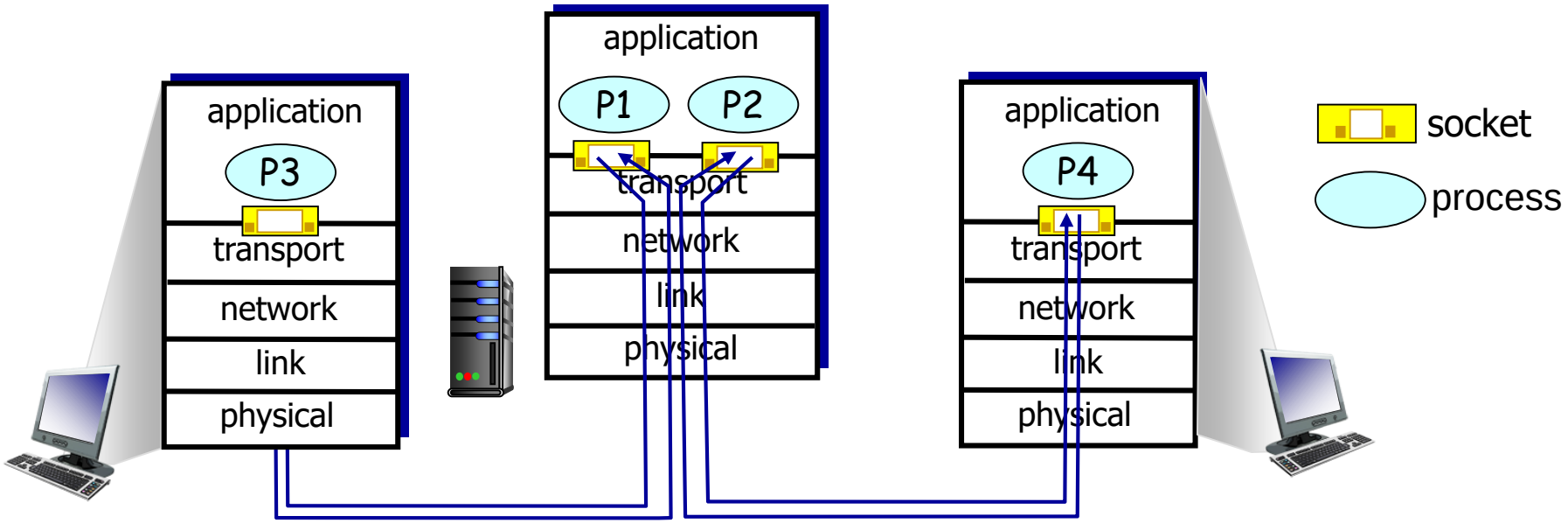
Πολύπλεξη/Αποπολύπλεξη

Αποπολύπλεξη στον υπολογιστή λήψης:

Χρήση πληροφοριών κεφαλίδας για παράδοση των τμημάτων που λαμβάνονται στη σωστή socket

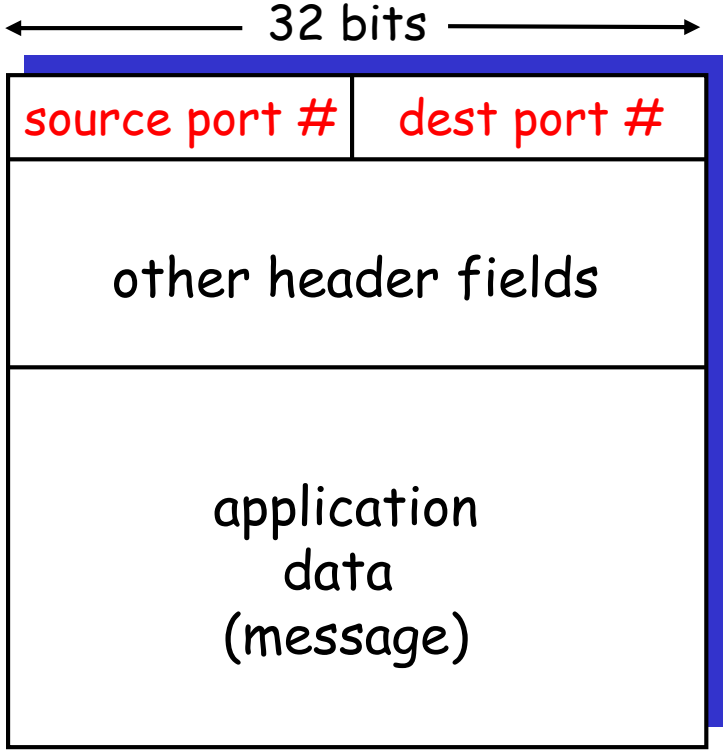
Πολύπλεξη στον υπολογιστή αποστολής:

Διαχείριση δεδομένων από πολλαπλές sockets, προσθήκη κεφαλίδας μεταφοράς (που αργότερα χρησιμοποιείται για αποπολύπλεξη)



Πώς δουλεύει η αποπολύπλεξη

- Ο υπολογιστής λαμβάνει IP datagrams (δεδομενογράμματα)
  - κάθε datagram έχει διεύθυνση IP προέλευσης, διεύθυνση IP προορισμού
  - κάθε datagram μεταφέρει ένα τμήμα επιπέδου μεταφοράς
  - κάθε τμήμα έχει αριθμό θύρας προέλευσης, προορισμού
- Ο υπολογιστής χρησιμοποιεί διευθύνσεις IP & αριθμούς θύρας για να κατευθύνει το τμήμα στην κατάλληλη socket



TCP/UDP segment format

Ασυνδεσμική αποπολύπλεξη (UDP)

- Το δημιουργημένο socket έχει αριθμό θύρας τοπικά στον υπολογιστή:

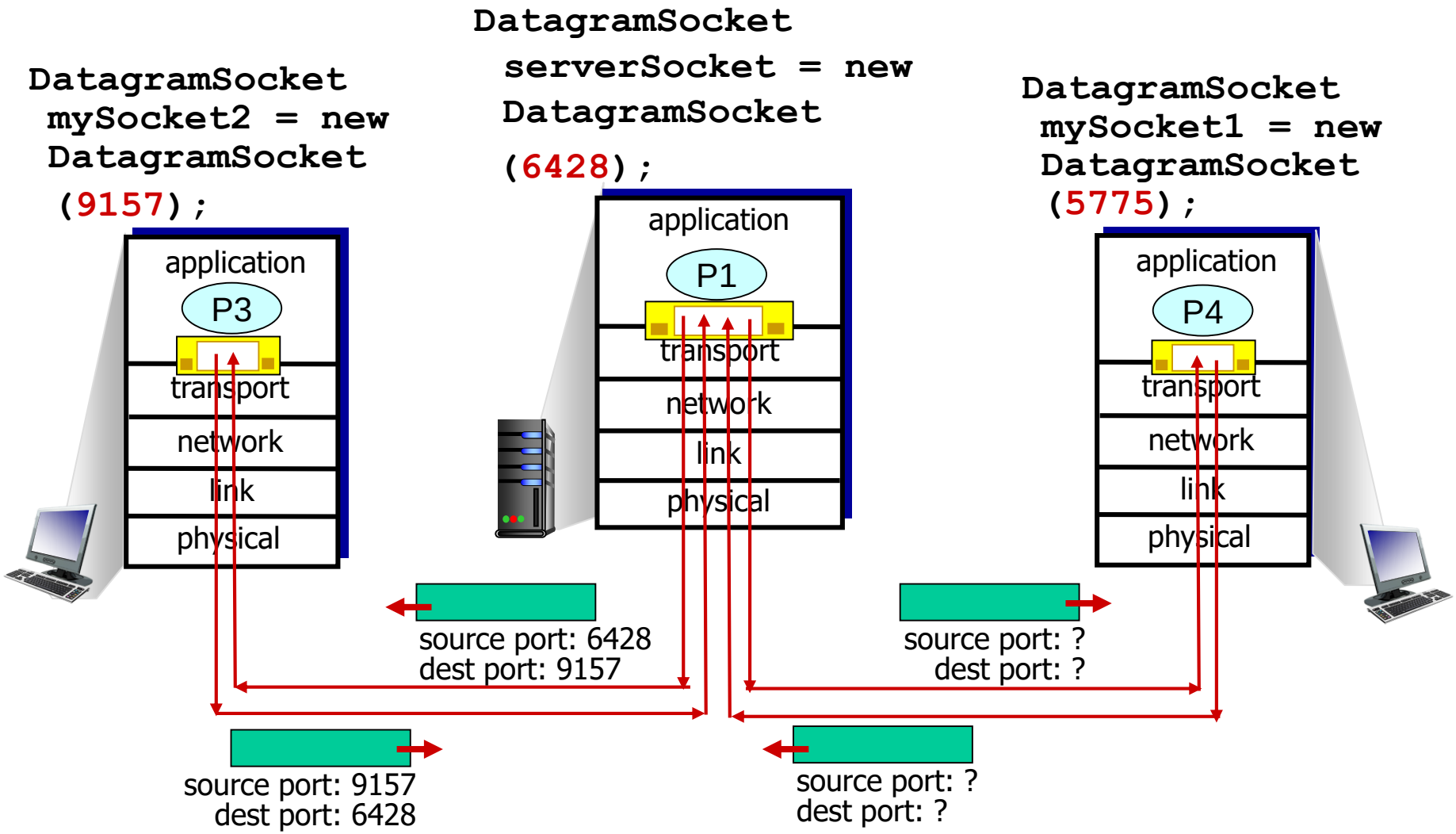
```
DatagramSocket mySocket1
= new DatagramSocket(12534);
```

- Όταν ο υπολογιστής λαμβάνει τμήμα UDP:
  - ελέγχει τον αριθμό θύρας προορισμού στο τμήμα
  - κατευθύνει το τμήμα UDP στη socket με αυτό τον αριθμό θύρας

- Όταν δημιουργείται το datagram για να σταλεί στη UDP socket, πρέπει να καθοριστεί
  - IP διεύθυνση προορισμού
  - Αριθμός θύρας προορισμού

- Τα IP datagrams με ίδιο αριθμό θύρας προορισμού, αλλά διαφορετικές IP διευθύνσεις προέλευσης ή/και αριθμούς θύρας προέλευσης θα κατευθυνθούν προς την ίδια socket προορισμού

Ασυνδεσμική αποπολύπλεξη (παράδειγμα)

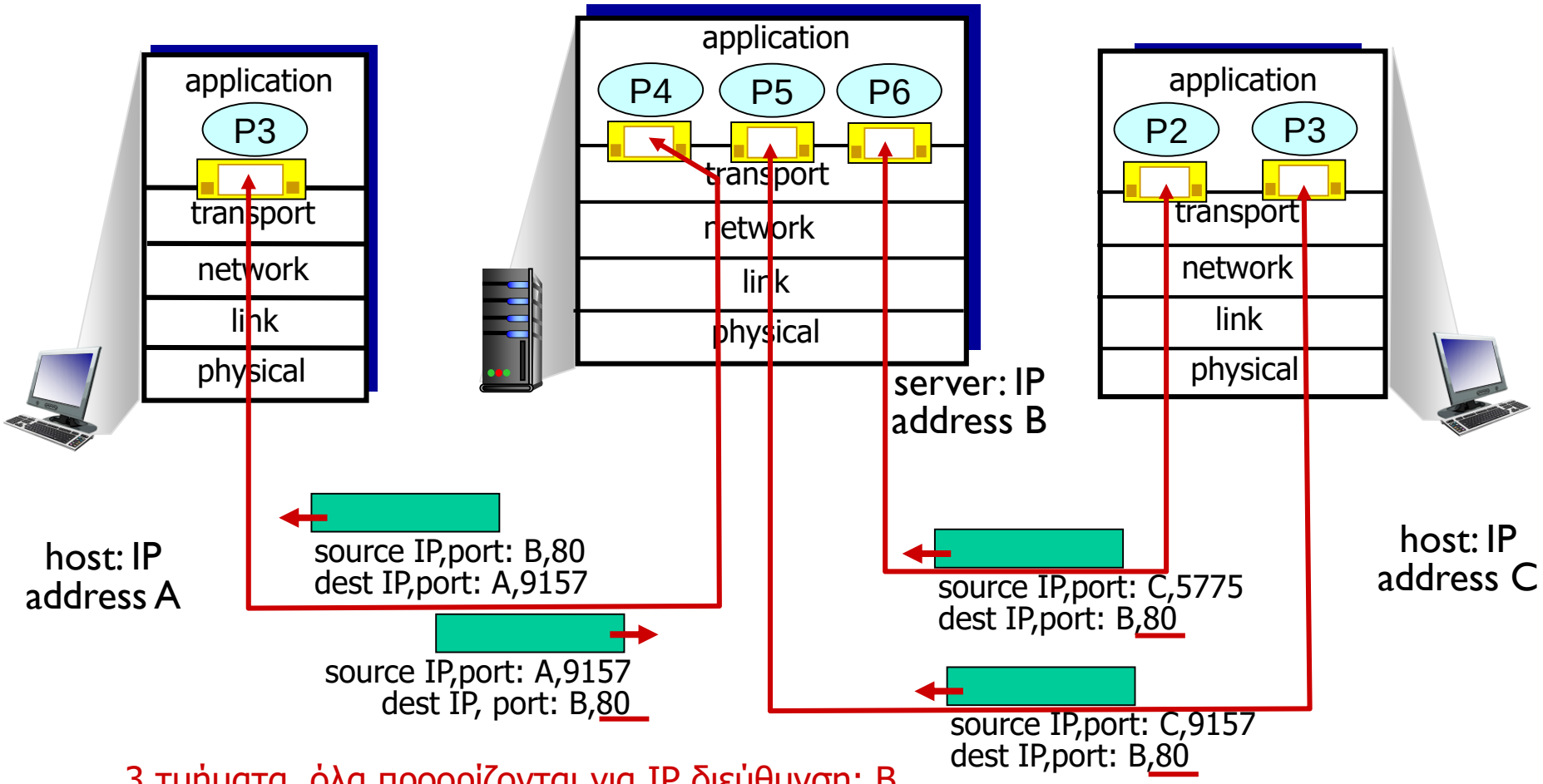


Το SP δίνει τη "διεύθυνση επιστροφής"

Συνδεσμική αποπολύπλεξη (TCP)

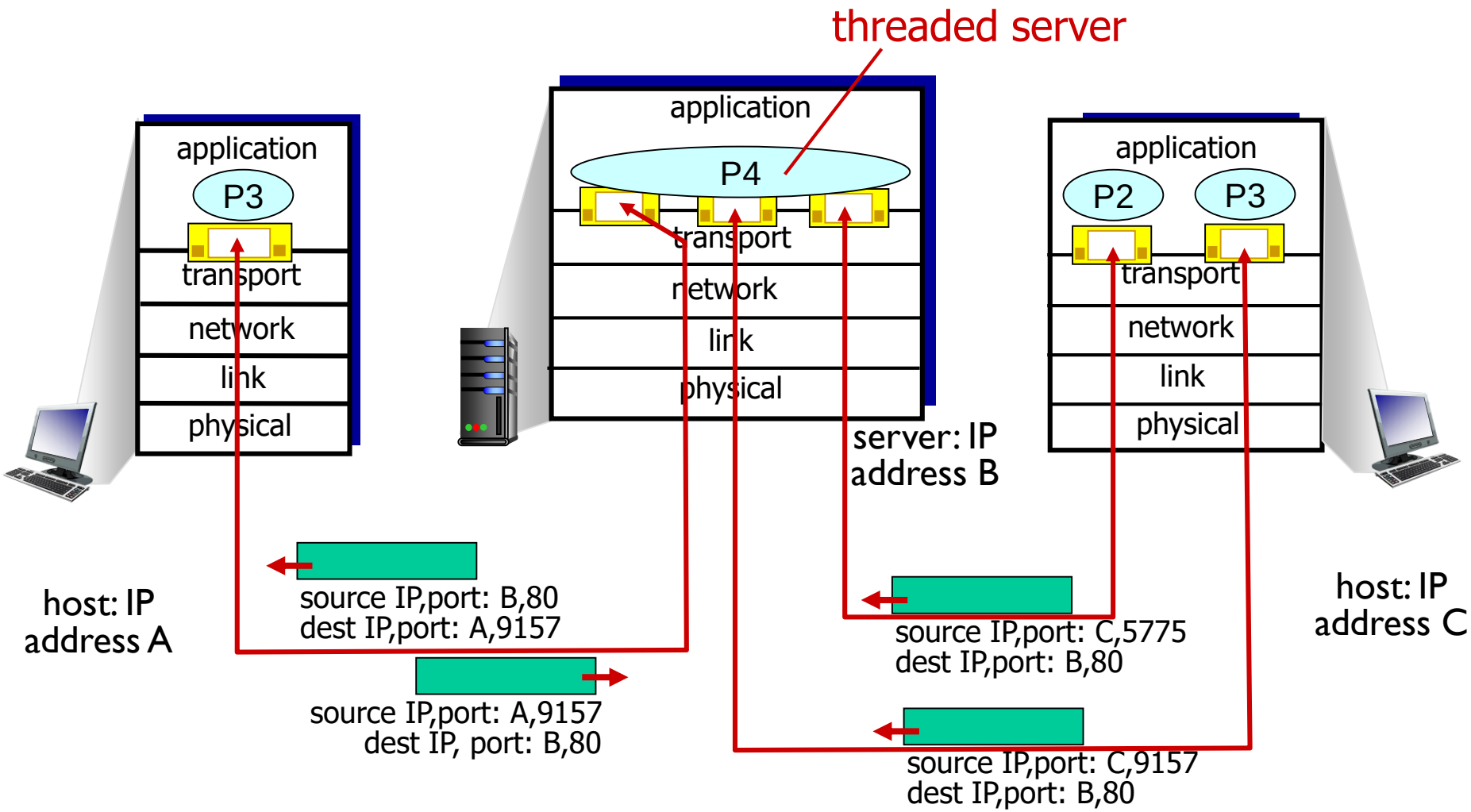
- Η TCP socket ταυτοποιείται από την τετράδα:
  - διεύθυνση IP προέλευσης
  - αριθμός θύρας προέλευσης
  - διεύθυνση IP προορισμού
  - αριθμός θύρας προορισμού
- *Αποπολύπλεξη*: ο δέκτης χρησιμοποιεί και τις τέσσερις τιμές για να κατευθύνει το τμήμα στην κατάλληλη socket
- Ένας εξυπηρέτης μπορεί να υποστηρίξει πολλές ταυτόχρονες TCP sockets:
  - κάθε socket αναγνωρίζεται από τη δική της τετράδα
- Οι εξυπηρέτες web έχουν διαφορετικές sockets για κάθε συνδεδεμένο πελάτη
  - το μη-παραμένον HTTP έχει διαφορετική socket για κάθε αίτηση HTTP

Συνδεσμική αποπολύπλεξη (παράδειγμα)



3 τμήματα, όλα προορίζονται για IP διεύθυνση: Β, θύρα προορισμού: 80 αποπολυπλέκονται σε διαφορετικές sockets

Συνδεσμική αποπολύπλεξη : Web Server με νήματα



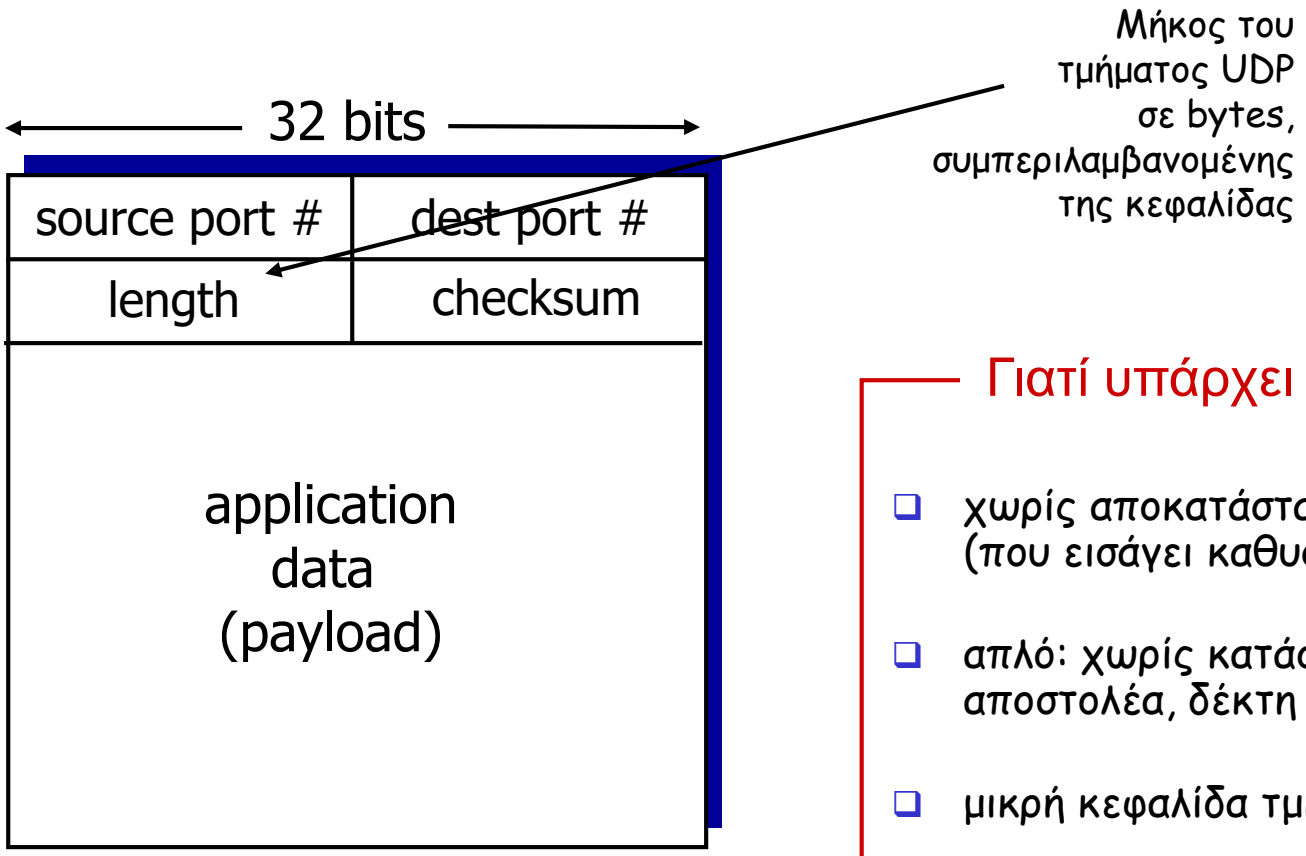
Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

UDP: User Datagram Protocol [RFC 768]

- Το απλούστερο πρωτόκολλο μεταφοράς του Διαδικτύου
- Υπηρεσία παράδοσης βέλτιστης προσπάθειας ("best effort"):
  τα τμήματα UDP μπορεί να
  - χαθούν
  - παραδοθούν εκτός σειράς στις εφαρμογές
- *Ασυνδεσμική:*
  - δεν γίνεται χειραψία μεταξύ του UDP αποστολέα και δέκτη
  - Η διαχείριση κάθε UDP τμήματος γίνεται ξεχωριστά
- Το UDP συχνά χρησιμοποιείται για εφαρμογές πολυμέσων συνεχούς ροής (streaming)
  - ανοχή ως προς τις απώλειες
  - ευαισθησία ως προς το ρυθμό
- άλλες χρήσεις του UDP π.χ.
  - DNS
  - SNMP
- αξιόπιστη μεταφορά πάνω από το UDP: προσθήκη αξιοπιστίας στο επίπεδο εφαρμογής
  - ανάνηψη από λάθη ανά εφαρμογή

UDP: κεφαλίδα τμήματος



Δομή τμήματος UDP

Γιατί υπάρχει το UDP;

- χωρίς αποκατάσταση σύνδεσης (που εισάγει καθυστέρηση)
- απλό: χωρίς κατάσταση σύνδεσης στον αποστολέα, δέκτη
- μικρή κεφαλίδα τμήματος
- χωρίς έλεγχο συμφόρησης: το UDP μπορεί να "εκραγεί" όσο γρήγορα θέλουμε

UDP checksum (άθροισμα ελέγχου)

Σκοπός: ανίχνευση "σφαλμάτων" (π.χ. ανεστραμμένων bits) στο μεταδιδόμενο τμήμα

Αποστολέας:

- χειρίζεται το περιεχόμενο του τμήματος, συμπεριλαμβανομένων πεδίων της κεφαλίδας, ως ακολουθία ακεραίων των 16 bits
- checksum: συμπλήρωμα ως προς το 1 του αθροίσματος του περιεχομένου (εκφρασμένου σε 16-bit λέξεις)
- ο αποστολέας τοποθετεί την τιμή του checksum στο πεδίο checksum του τμήματος UDP

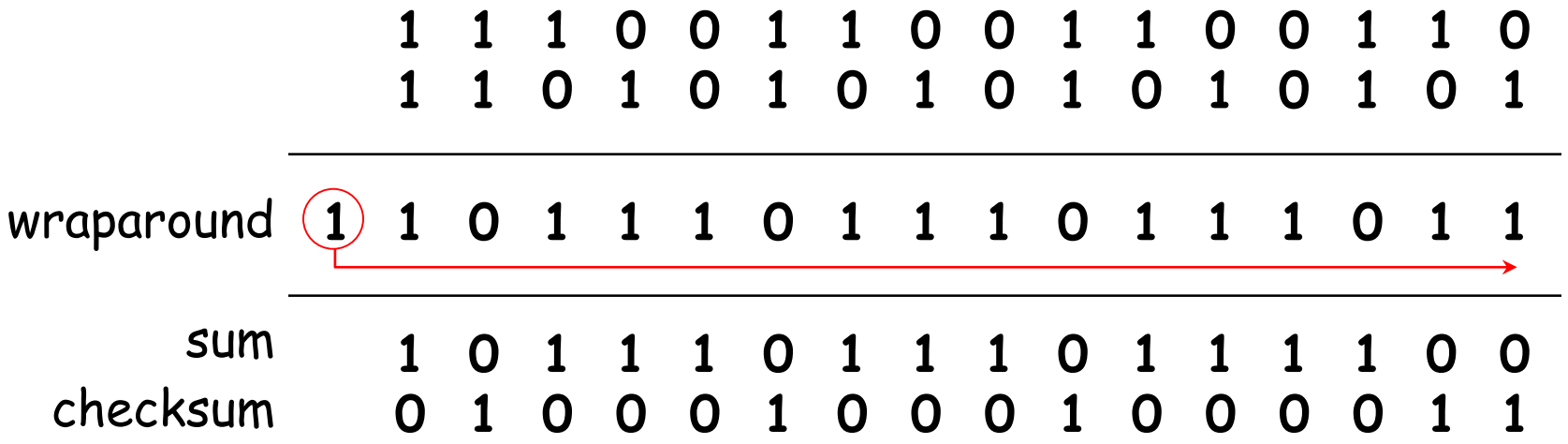
Δέκτης:

- υπολογισμός του checksum του λαμβανομένου μηνύματος
- έλεγχος αν το υπολογισμένο checksum ισούται με την τιμή του πεδίου checksum:
  - Όχι – ανίχνευση σφάλματος
  - Ναι – μη ανίχνευση σφάλματος

  Αλλά παρόλα αυτά ενδέχεται να γίνουν λάθη; (περισσότερα στη συνέχεια)

Παράδειγμα Checksum Διαδικτύου

- Σημείωση
  - Κατά την πρόσθεση των αριθμών, το κρατούμενο από την πιο σημαντική θέση πρέπει να προστεθεί στο αποτέλεσμα
- Παράδειγμα: πρόσθεση δύο ακεραίων 16-bit

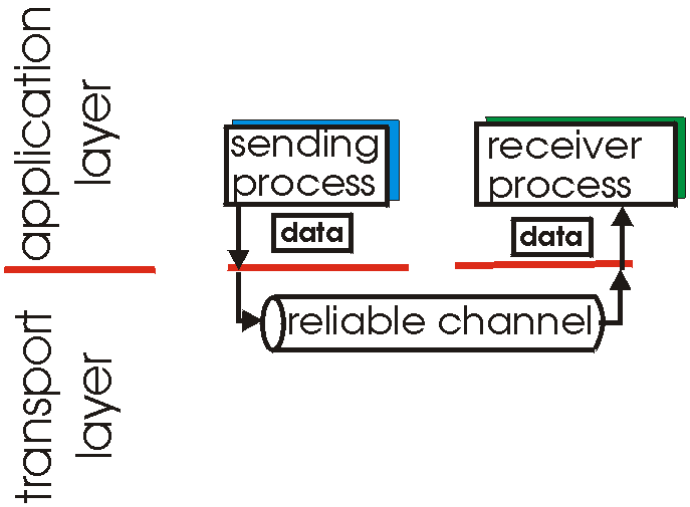


Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

Αρχές αξιόπιστης μεταφοράς δεδομένων

- Σημαντική στα επίπεδα εφαρμογής, μεταφοράς, ζεύξης
- Στο top-10 των σημαντικών θεμάτων της δικτύωσης!

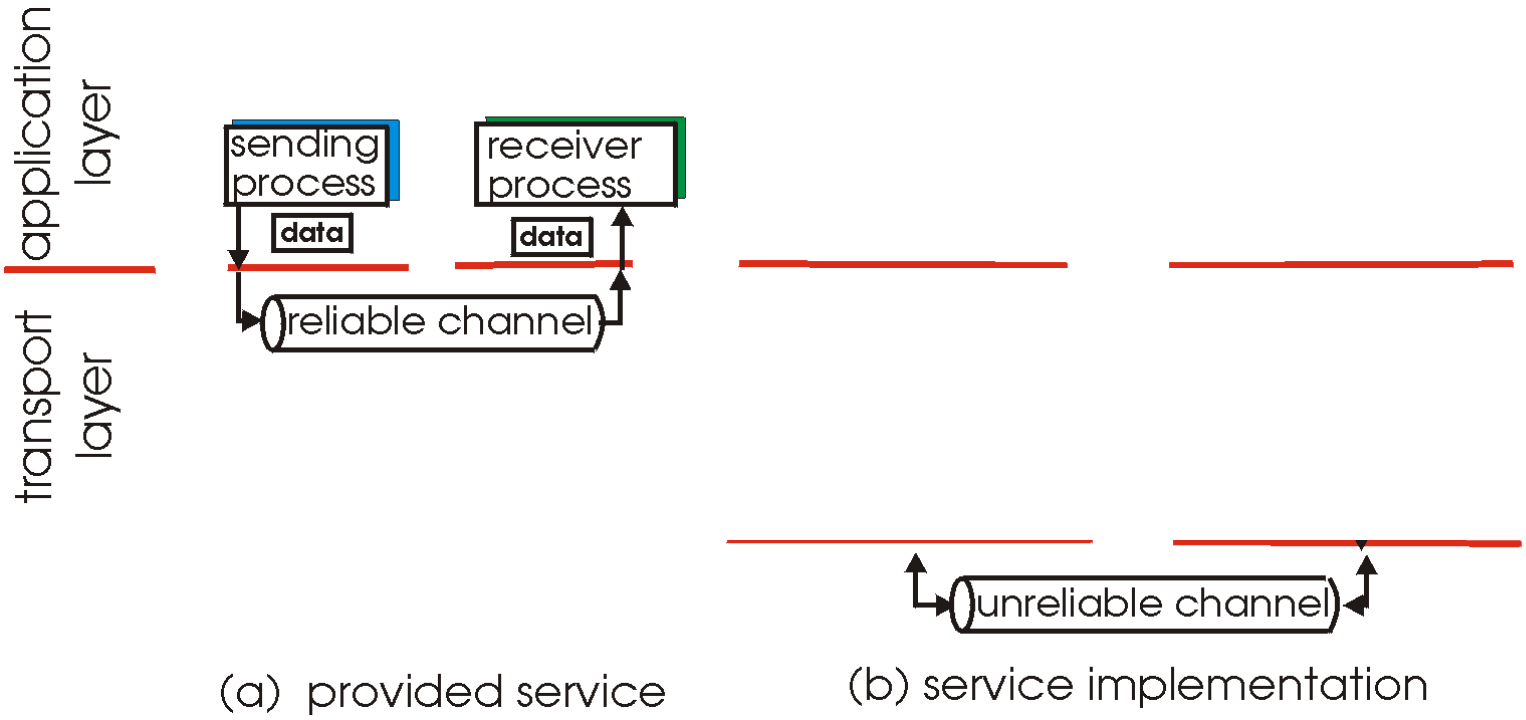


(a) provided service

- Τα χαρακτηριστικά του αναξιόπιστου καναλιού θα καθορίσουν την πολυπλοκότητα του πρωτοκόλλου αξιόπιστης μεταφοράς δεδομένων (reliable data transfer protocol- rdt)

Αρχές της αξιόπιστης μεταφοράς δεδομένων

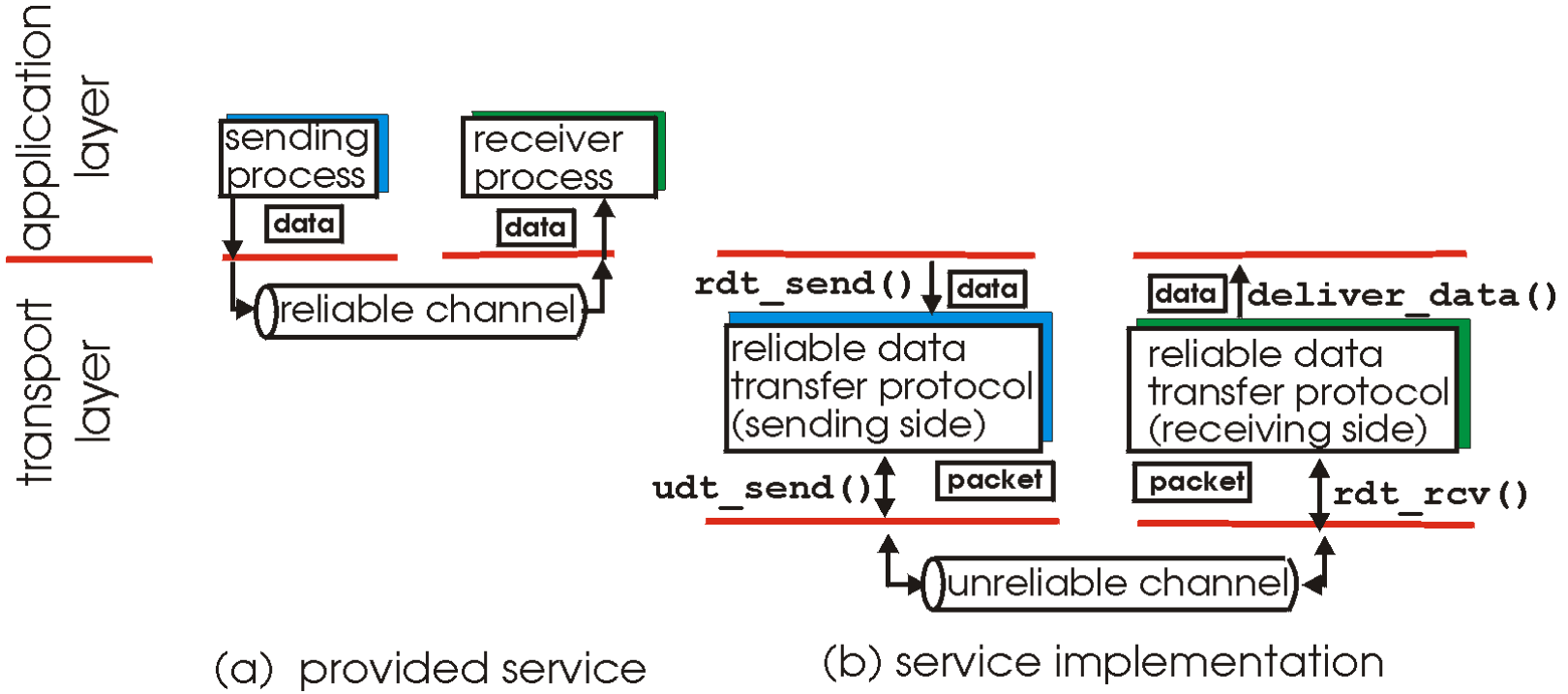
- Σημαντική στα επίπεδα εφαρμογής, μεταφοράς, ζεύξης
- Στο top-10 των σημαντικών θεμάτων της δικτύωσης!



- Τα χαρακτηριστικά του αναξιόπιστου καναλιού θα καθορίσουν την πολυπλοκότητα του πρωτοκόλλου αξιόπιστης μεταφοράς δεδομένων (reliable data transfer protocol- rdt)

Αρχές της αξιόπιστης μεταφοράς δεδομένων

- Σημαντική στα επίπεδα εφαρμογής, μεταφοράς, ζεύξης
- Στο top-10 των σημαντικών θεμάτων της δικτύωσης!

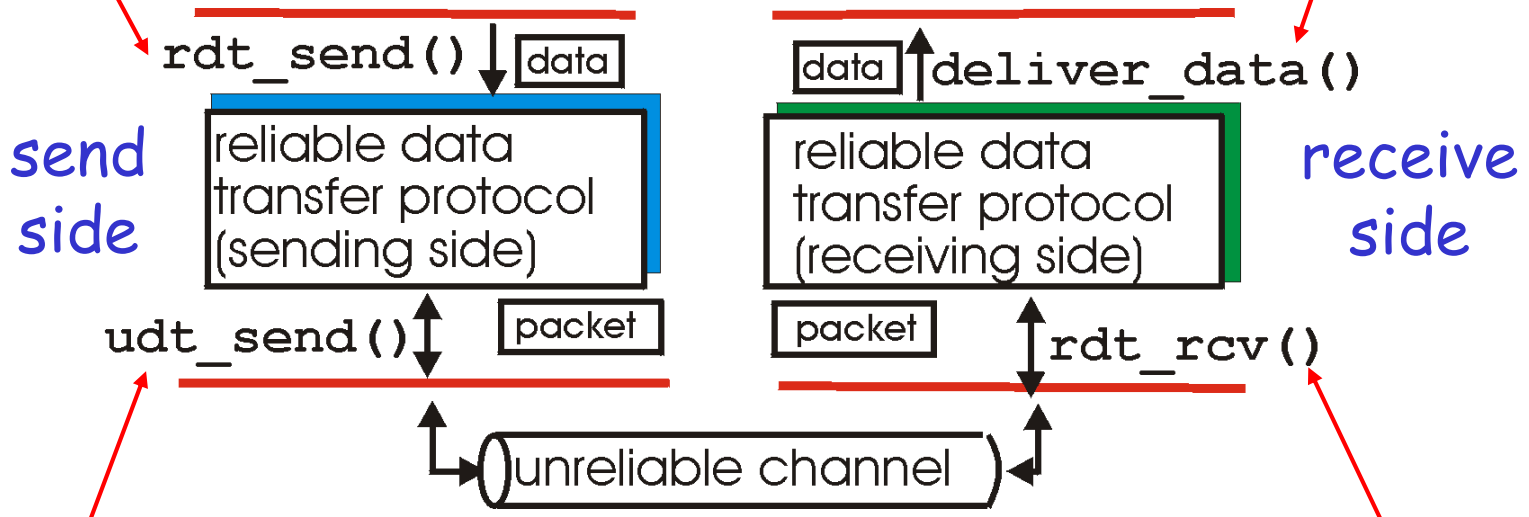


- Τα χαρακτηριστικά του αναξιόπιστου καναλιού θα καθορίσουν την πολυπλοκότητα του πρωτοκόλλου αξιόπιστης μεταφοράς δεδομένων (reliable data transfer protocol- rdt)

Αξιόπιστη μεταφορά δεδομένων: ξεκινώντας

rdt_send(): καλείται από πάνω, (π.χ., από την εφαρμογή). Δίνονται δεδομένα για παράδοση στο ανώτερο επίπεδο του δέκτη

deliver_data(): καλείται από το rdt για να παραδώσει δεδομένα στο ανώτερο επίπεδο



udt_send(): καλείται από το rdt, για τη μεταφορά πακέτου πάνω από αναξιόπιστο κανάλι στον δέκτη

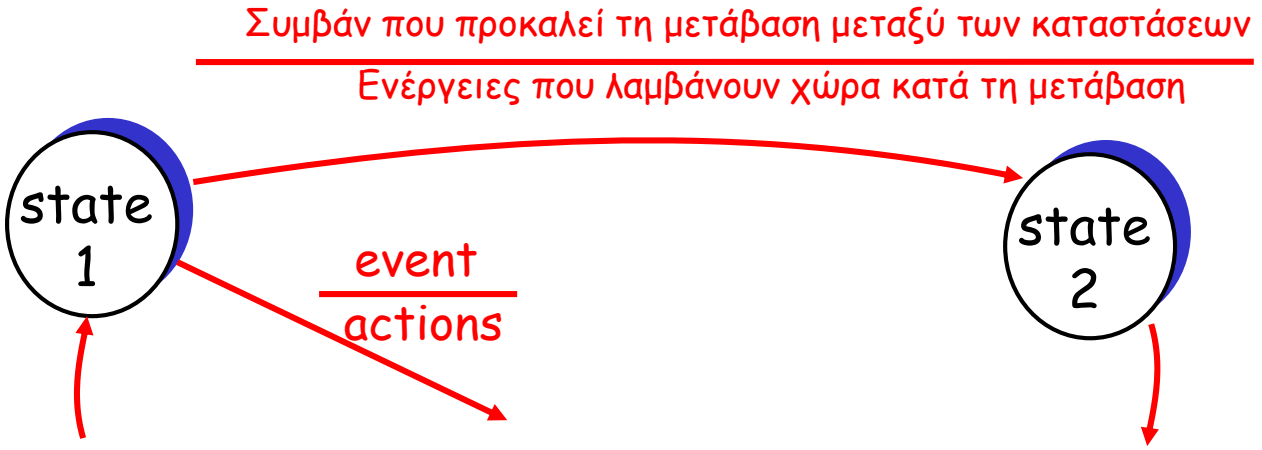
rdt_rcv(): καλείται όταν το πακέτο φτάσει στην πλευρά του καναλιού του δέκτη

Αξιόπιστη μεταφορά δεδομένων: ξεκινώντας

Θα:

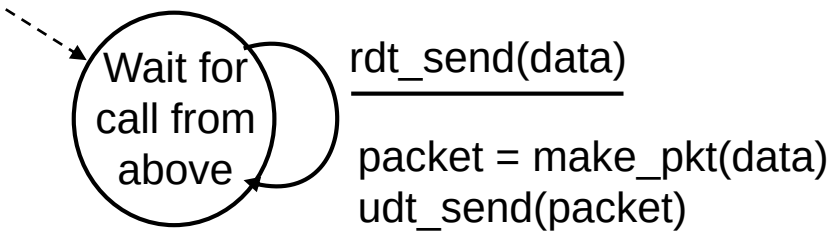
- αναπτύξουμε σταδιακά τις πλευρές του αποστολέα και του δέκτη ενός πρωτοκόλλου αξιόπιστης μεταφοράς δεδομένων [reliable data transfer **(rdt)** protocol]
- Θεωρήσουμε μόνο μονόδρομη μεταφορά δεδομένων
  - Αλλά η κίνηση ελέγχου θα ρέει και προς τις δύο κατευθύνσεις !
- Χρήση μηχανών πεπερασμένων καταστάσεων [finite state machines (FSM)] για την περιγραφή του αποστολέα και του δέκτη

Κατάσταση (state): όντας στην «κατάσταση» η επόμενη κατάσταση καθορίζεται μοναδικά από το επόμενο συμβάν

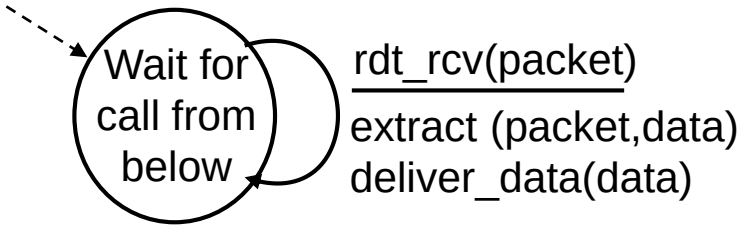


Rdt1.0: αξιόπιστη μεταφορά επάνω σε αξιόπιστο κανάλι

- Το υποκείμενο κανάλι είναι πλήρως αξιόπιστο
  - Χωρίς λάθη bit
  - Χωρίς απώλειες πακέτων
- Ξεχωριστές FSMs για τον αποστολέα, δέκτη:
  - Ο αποστολέας στέλνει δεδομένα στο υποκείμενο κανάλι
  - Ο δέκτης διαβάζει δεδομένα από το υποκείμενο κανάλι



Αποστολέας (sender)



Δέκτης (receiver)

Rdt2.0: κανάλι με σφάλματα bit

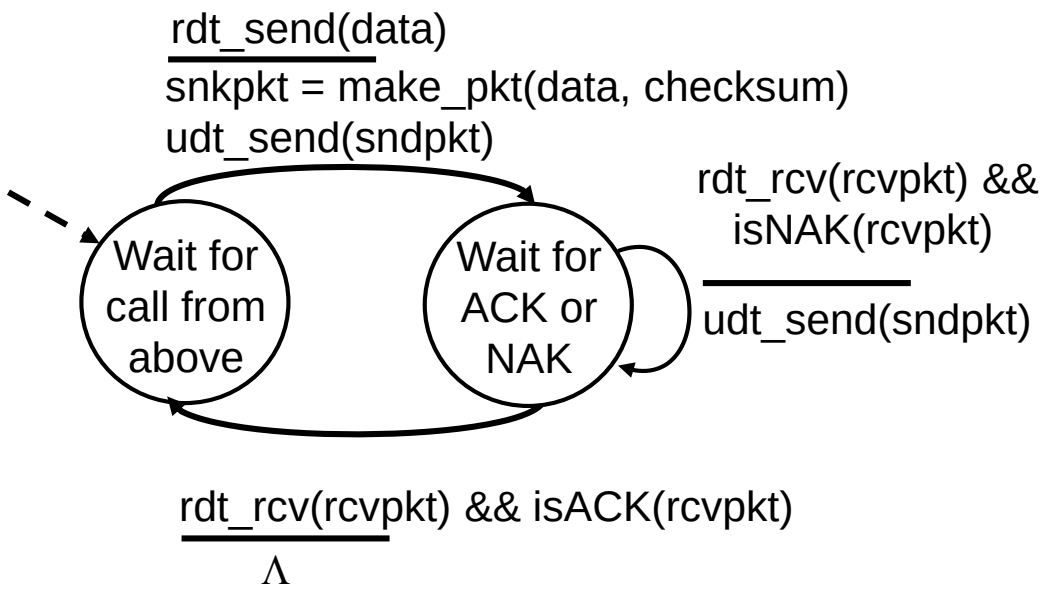
- Το υποκείμενο κανάλι ενδέχεται να αντιστρέψει bits στο πακέτο
  - Checksum για την ανίχνευση σφαλμάτων bit
- Ερώτημα: πώς να γίνεται η ανάνηψη από λάθη:

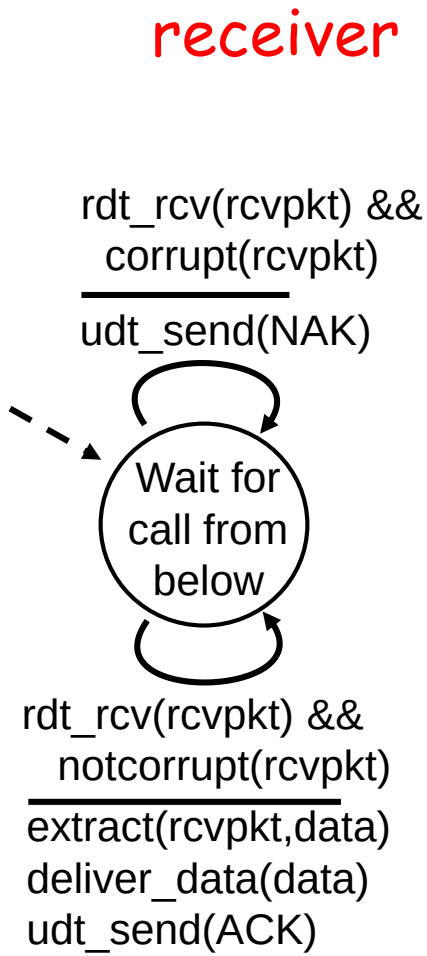
Πως ανακάμπτουν οι άνθρωποι μετά από "λάθη"
κατά τη διάρκεια συνομιλίας;

Rdt2.0: κανάλι με σφάλματα bit

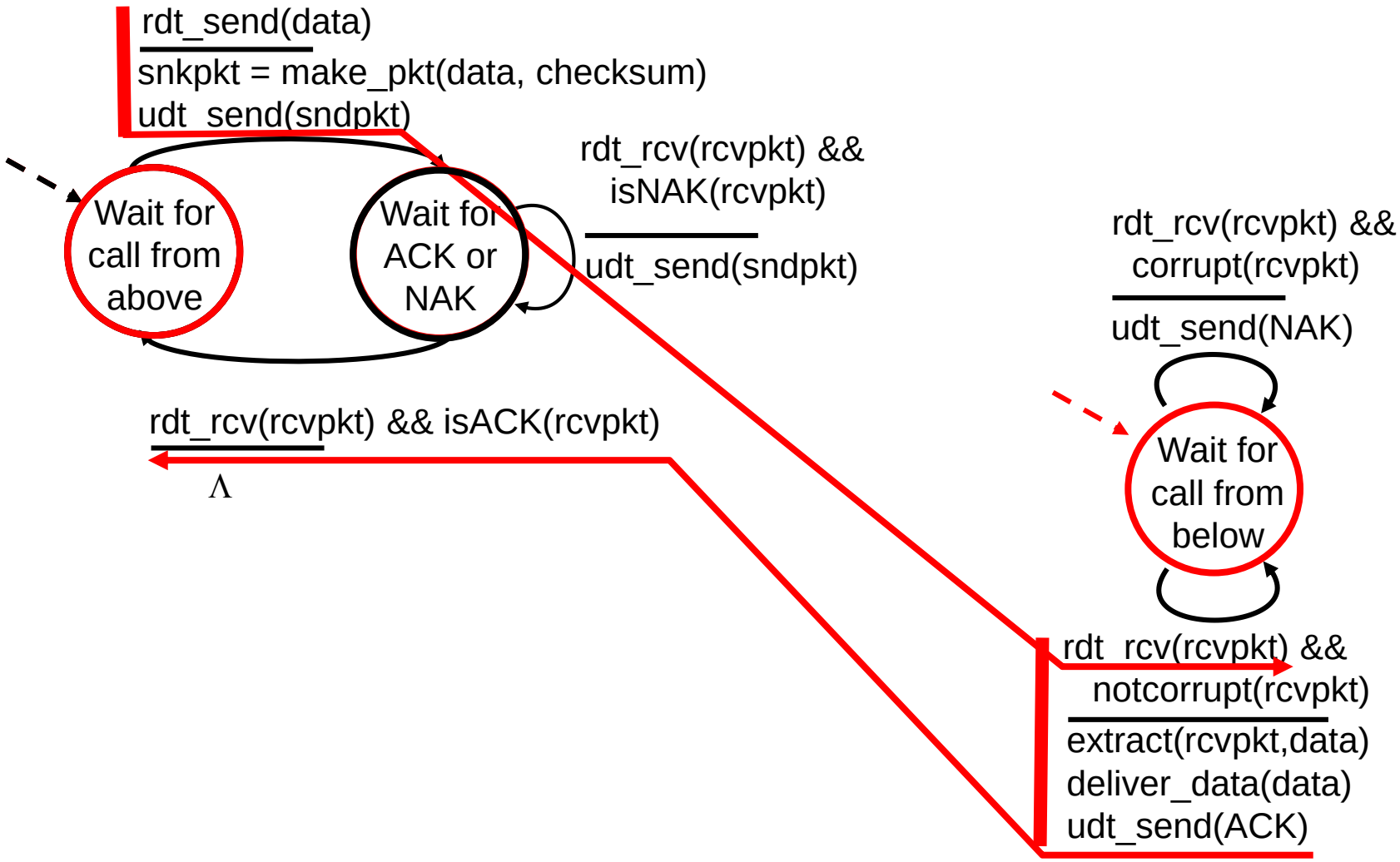
- Το υποκείμενο κανάλι δεν χάνει ποτέ πακέτα (δεδομένα ή ACKs)
- Το υποκείμενο κανάλι ενδέχεται να αναστρέψει bits στο πακέτο
  - Άθροισμα ελέγχου (Checksum) για την ανίχνευση σφαλμάτων bit
- *Ερώτημα*: πώς γίνεται η ανάνηψη από λάθη;
  - *Θετικές επιβεβαιώσεις [acknowledgements (ACKs)]*: ο δέκτης λέει ρητά στον αποστολέα ότι το πακέτο λήφθηκε σωστά
  - *Αρνητικές επιβεβαιώσεις [negative acknowledgements (NAKs)]*: ο δέκτης λέει ρητά στον αποστολέα ότι το πακέτο είχε λάθη
  - Ο δέκτης αναμεταδίδει το πακέτο μόλις λάβει NAK
- Νέοι μηχανισμοί στο `rdt2.0` (πέρα από το `rdt1.0`):
  - Ανίχνευση σφάλματος (error detection)
  - Ανάδραση δέκτη (receiver feedback): μηνύματα ελέγχου (ACK,NAK) δέκτης->αποστολέας

rdt2.0: περιγραφή FSM

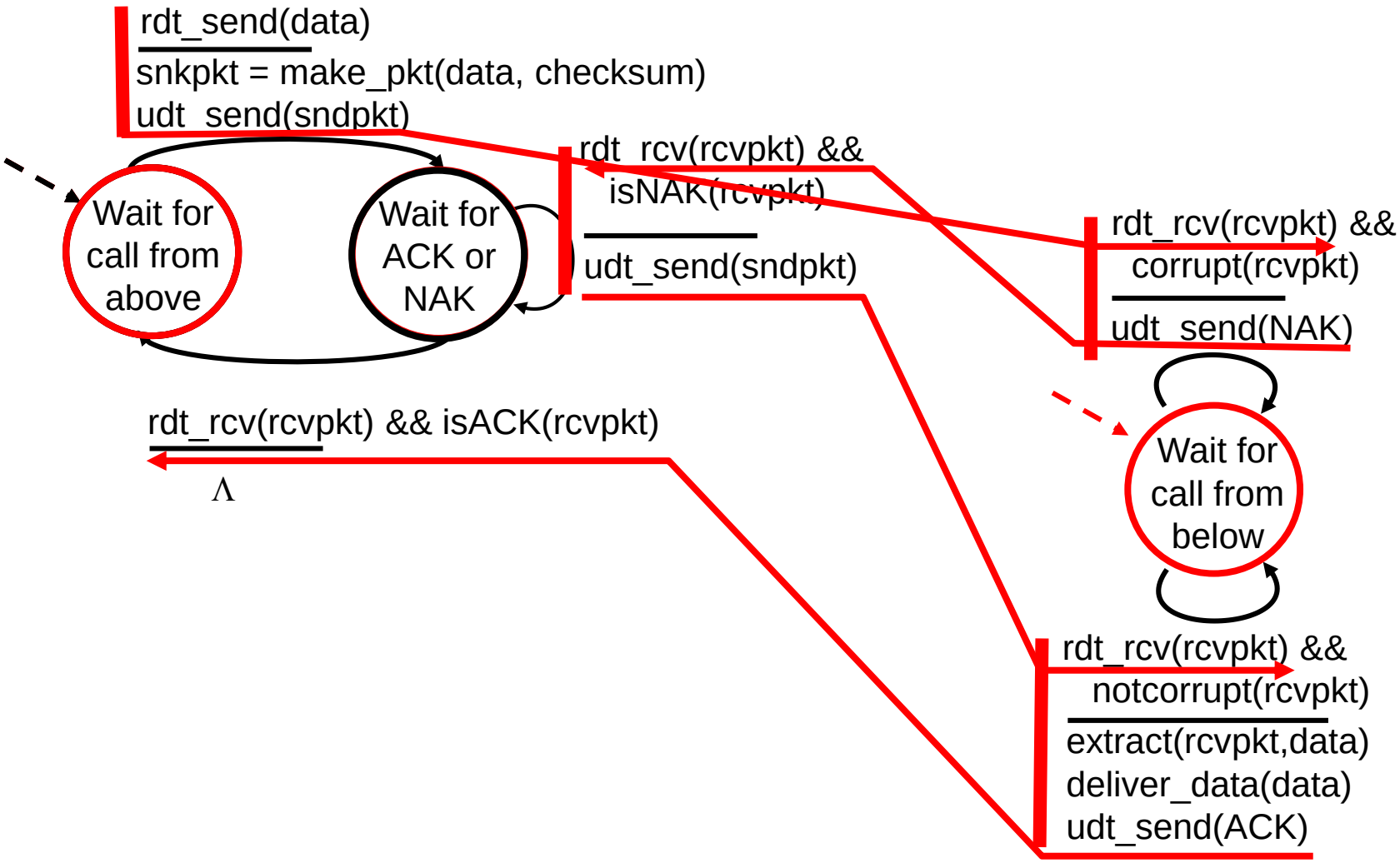




rdt2.0: λειτουργία χωρίς σφάλματα



rdt2.0: σενάριο σφάλματος



Το rdt2.0 έχει ένα μοιραίο σφάλμα!

Τι συμβαίνει αν καταστραφεί το ACK/NAK?

- Ο αποστολέας δεν γνωρίζει τι συνέβη στο δέκτη!
- Δεν μπορεί απλά να αναμεταδώσει: ενδεχομένως διπλά (duplicate) πακέτα

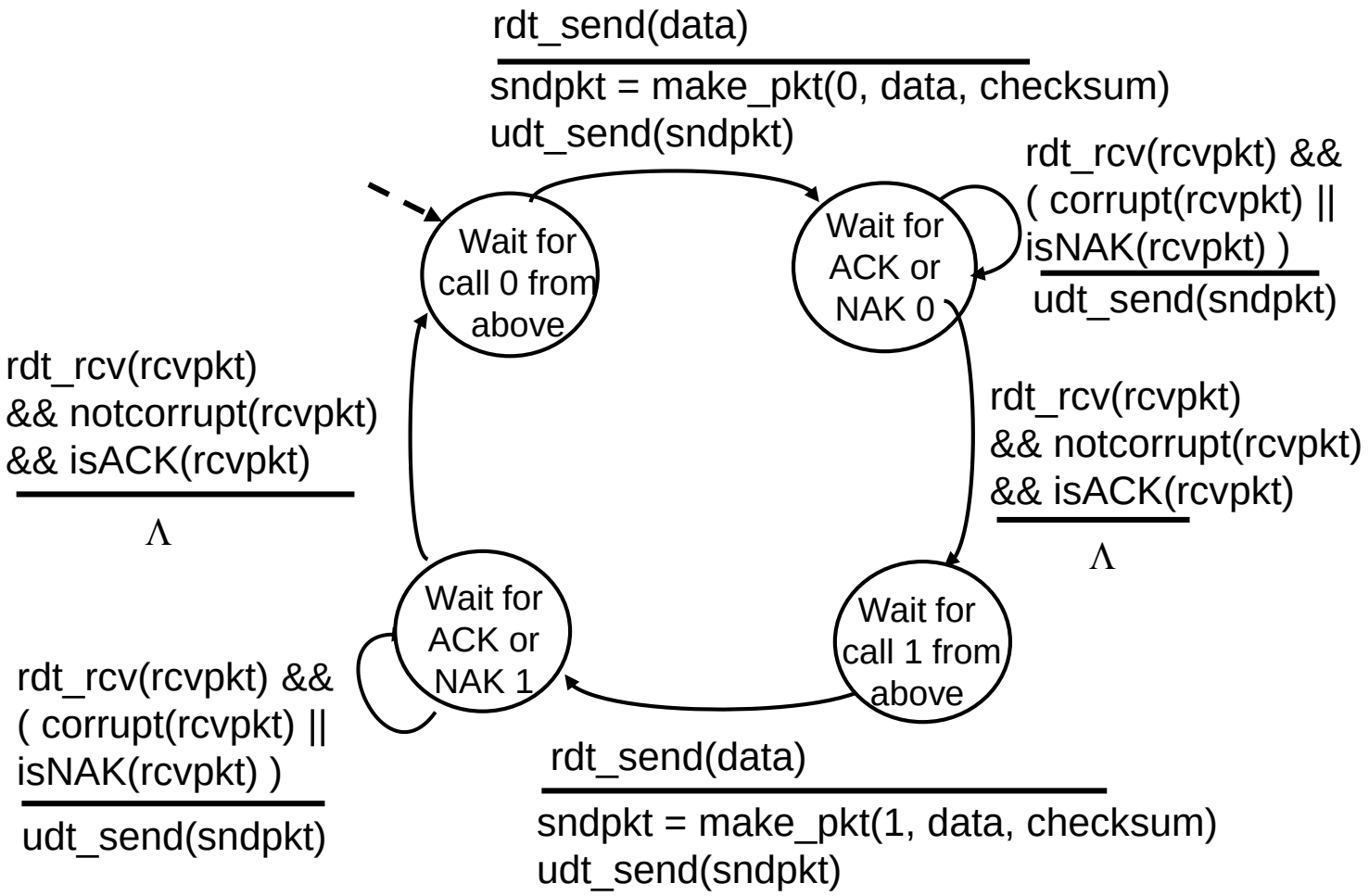
Διαχείριση διπλών πακέτων:

- Ο αποστολέας αναμεταδίδει το τρέχον πακέτο αν αλλοιωθεί το ACK/NAK
- Ο αποστολέας προσθέτει *αριθμό ακολουθίας (sequence number)* σε κάθε πακέτο
- Ο δέκτης απορρίπτει (δεν προωθεί προς τα πάνω) τα διπλά πακέτα

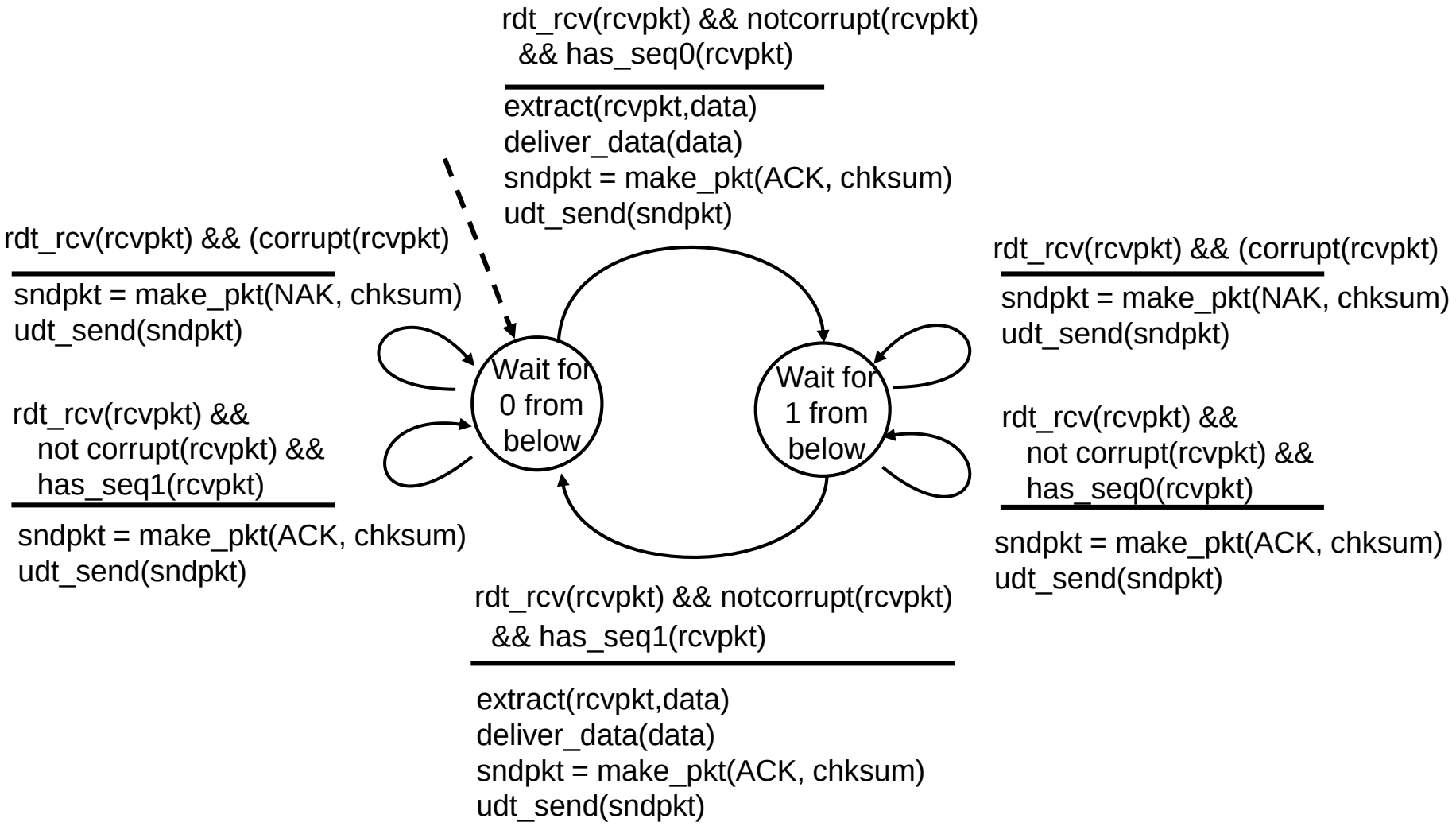
Διακοπή και αναμονή (stop and wait)

Ο αποστολέας στέλνει ένα πακέτο και μετά περιμένει για την απόκριση του δέκτη

rdt2.1: αποστολέας, διαχειρίζεται αλλοιωμένα ACK/NAKs



rdt2.1: δέκτης, διαχειρίζεται αλλοιωμένα ACK/NAKs



rdt2.1: Συζήτηση

Αποστολέας:

- Στο πακέτο προστίθεται # ακολουθίας
- Δύο # ακολουθίας (0,1) αρκούν. Γιατί;
- Πρέπει να ελέγξει αν το ACK/NAK που έλαβε είναι κατεστραμμένο
- Διπλάσιες καταστάσεις
  - Η κατάσταση πρέπει να «θυμάται» αν το «αναμενόμενο» πακέτο έχει # ακολουθίας 0 ή 1

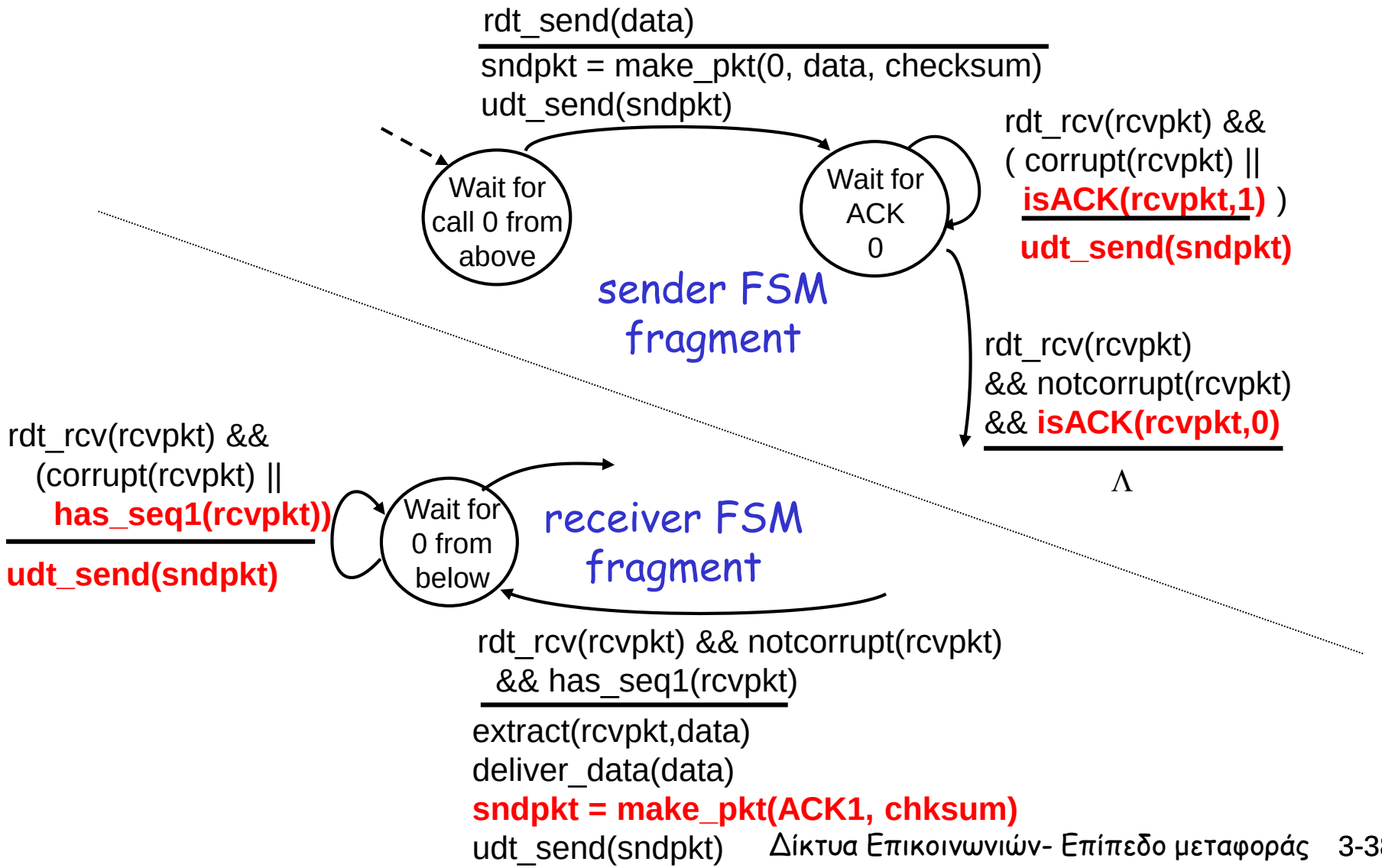
Δέκτης:

- Πρέπει να ελέγξει αν το λαμβανόμενο πακέτο είναι διπλό
  - Η κατάσταση υποδεικνύει αν αναμένεται 0 ή 1 ως # ακολουθίας πακέτου
- Σημείωση: ο δέκτης δεν μπορεί να γνωρίζει αν το τελευταίο του ACK/NAK ελήφθη σωστά στον αποστολέα

rdt2.2: Ένα πρωτόκολλο χωρίς NAK

- Ίδια λειτουργικότητα με το rdt2.1, χρησιμοποιώντας μόνο*ACKs*
- Αντί για NAK, ο δέκτης στέλνει ACK για το τελευταίο πακέτο που έλαβε σωστά
  - Ο δέκτης πρέπει ρητά να συμπεριλάβει τον # ακολουθίας του πακέτου για το οποίο γίνεται η θετική επιβεβαίωση (ACK)
- Διπλό (duplicate) ACK στον αποστολέα έχει σαν αποτέλεσμα την ίδια ενέργεια όπως το NAK: *αναμεταδίδει το τρέχον πακέτο*

rdt2.2: Αποσπάσματα αποστολέα, δέκτη



rdt3.0: κανάλια με λάθη και απώλειες

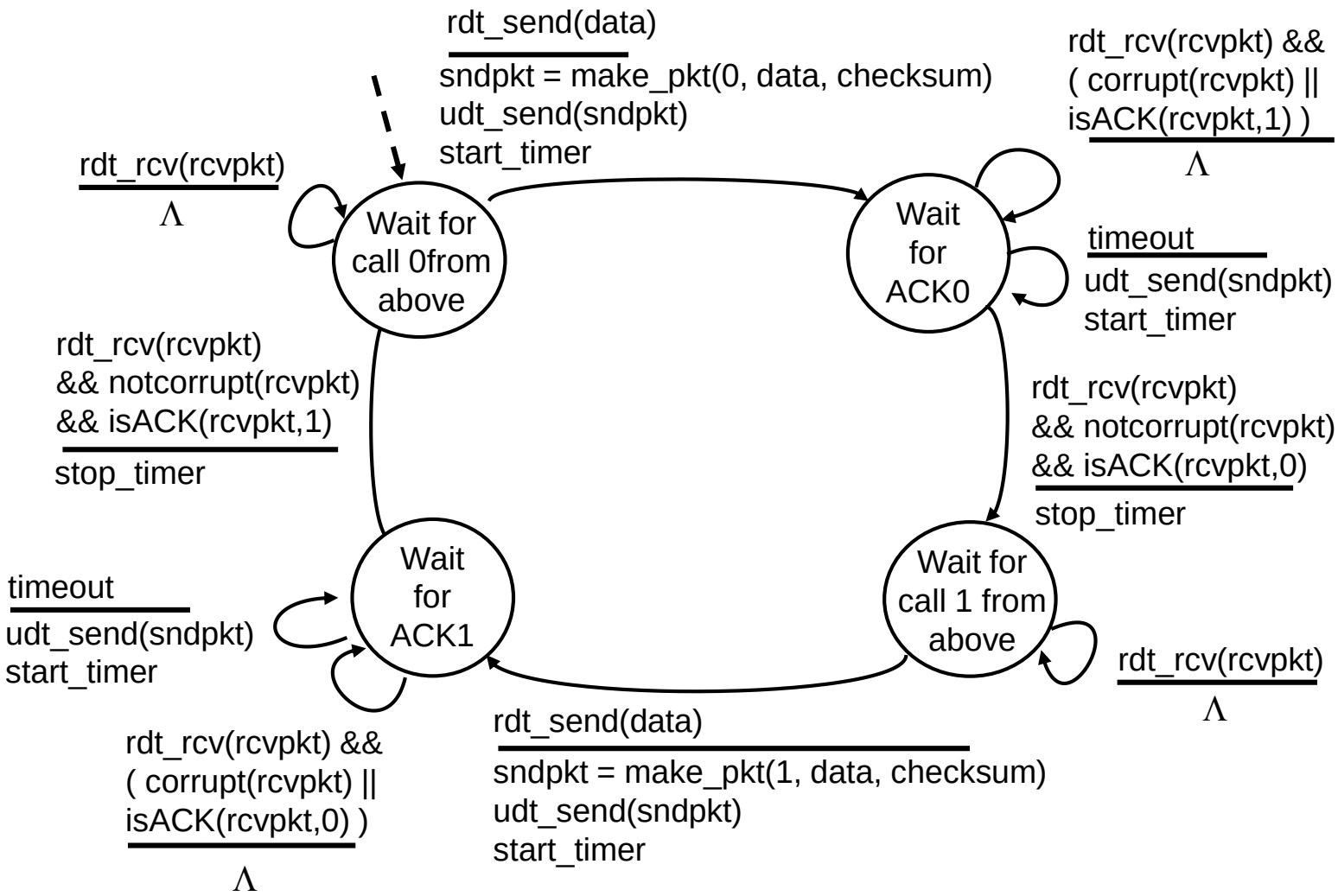
Νέα υπόθεση: το υποκείμενο κανάλι μπορεί επίσης να χάσει πακέτα (δεδομένα ή ACKs)

- Άθροισμα ελέγχου, # ακολουθίας, ACKs, αναμεταδόσεις βοηθούν αλλά δεν αρκούν

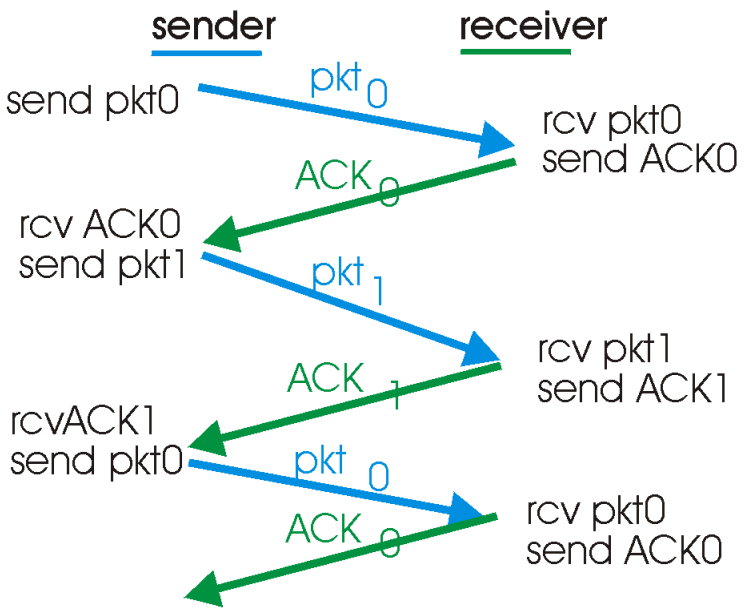
Προσέγγιση: ο αποστολέας περιμένει για «εύλογο» χρονικό διάστημα για το ACK

- Αναμεταδίδει αν δεν ληφθεί ACK σε αυτό το διάστημα
- Αν το πακέτο (ή το ACK) απλά καθυστέρησε (δεν χάθηκε):
  - Οι αναμεταδόσεις θα είναι διπλές (duplicate), αλλά η χρήση # ακολουθίας το αντιμετωπίζει ήδη αυτό
  - Ο δέκτης πρέπει να καθορίσει # ακολουθίας του πακέτου για το οποίο είναι το ACK
- Απαιτεί χρονομετρητή αντίστροφης μέτρησης

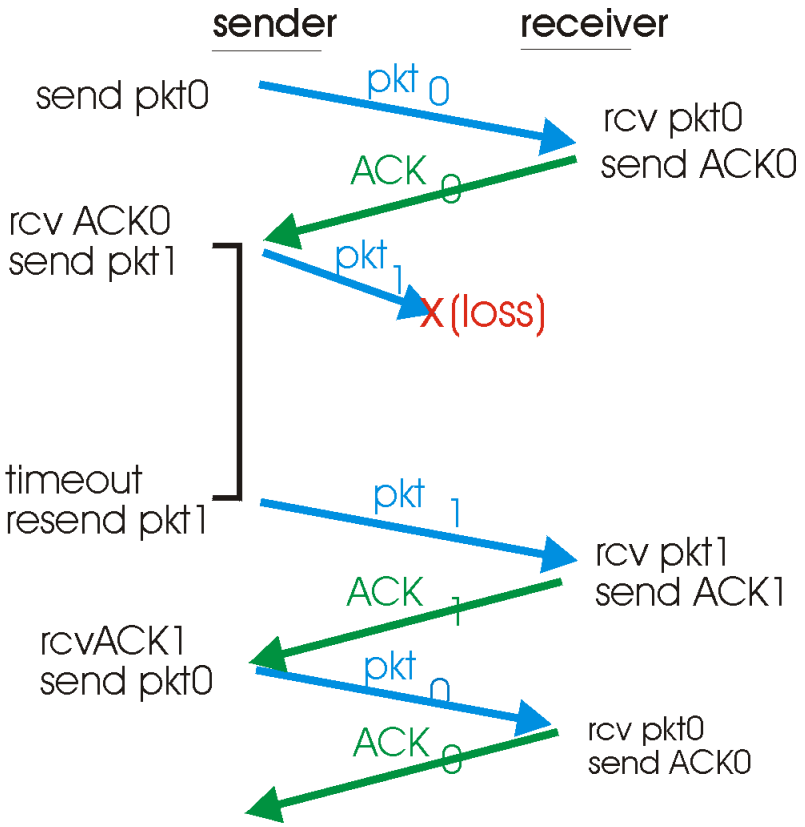
rdt3.0 αποστολέας



rdt3.0 εν δράσει

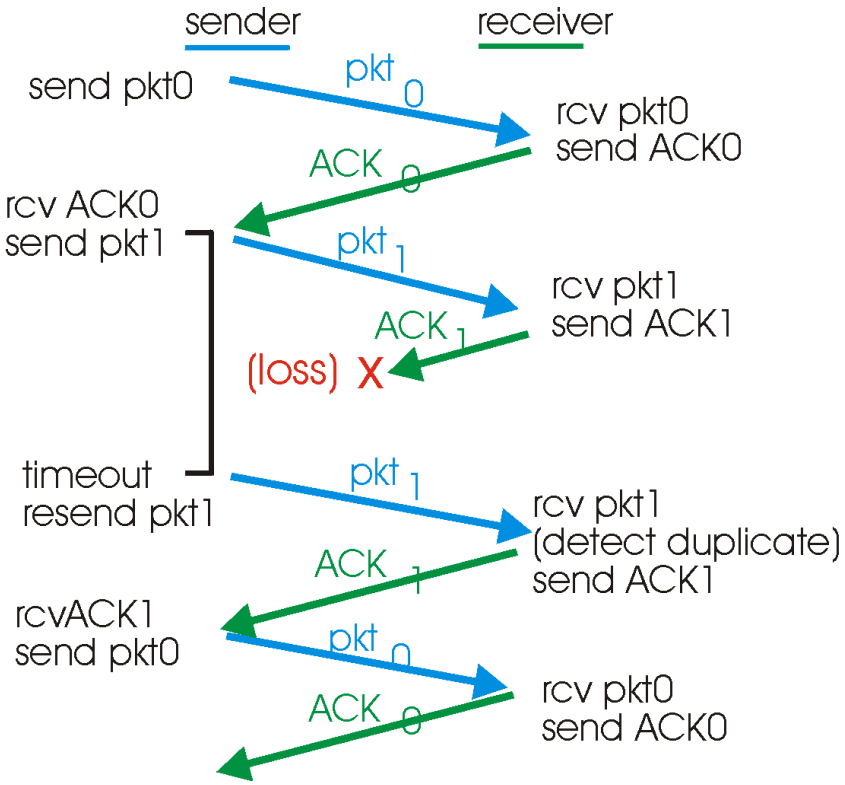


(a) operation with no loss

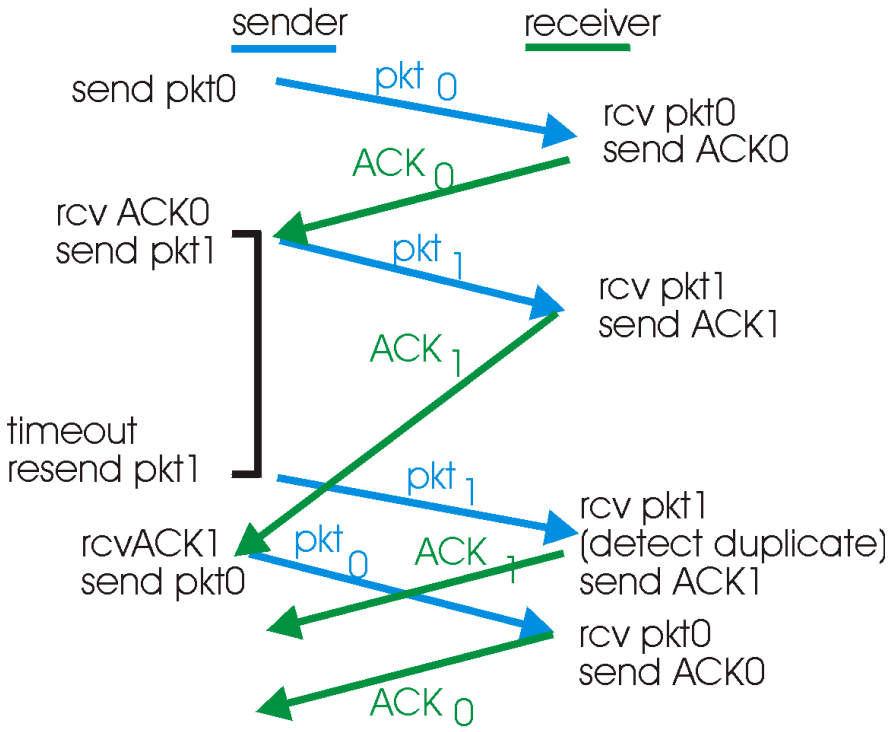


(b) lost packet

rdt3.0 εν δράσει



(c) lost ACK



(d) premature timeout

Απόδοση του rdt3.0

- Το rdt3.0 δουλεύει, αλλά η απόδοσή του είναι χαμηλή
- Π.χ.: ζεύξη 1 Gbps, καθυστ. διαδ. 15 ms, πακ. 8000 bit:

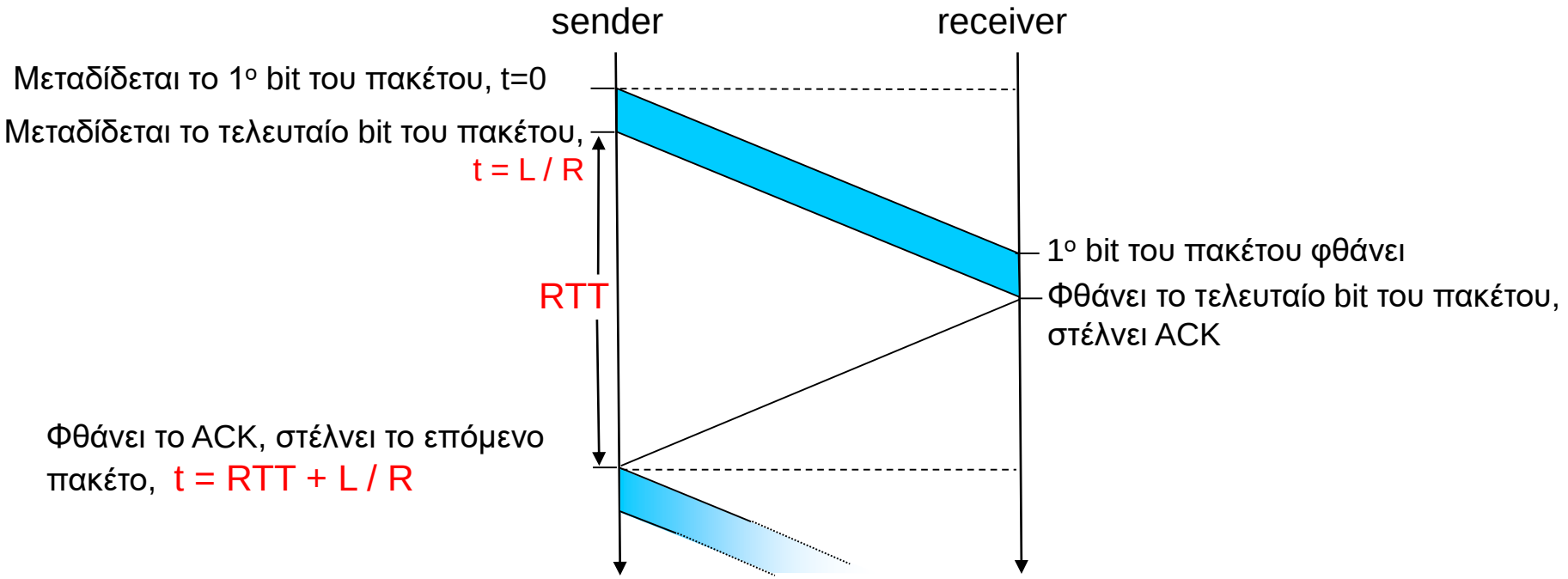
$$d_{trans} = \frac{L}{R} = \frac{8000 \text{ bits}}{10^9 \text{ bps}} = 8 \mu\text{sec}$$

- U_{sender}: βαθμός χρήσης (utilization) – ποσοστό του χρόνου που ο αποστολέας είναι απασχολημένος στέλνοντας

$$U_{sender} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

- Αν το RTT=30 msec, πακέτο 1KB κάθε 30 msec -> 33kB/sec =264 kbps throughput σε ζεύξη 1 Gbps
  - Το δικτυακό πρωτόκολλο περιορίζει τη χρήση των φυσικών πόρων!

rdt3.0: Λειτουργία διακοπής και αναμονής (stop-and-wait)

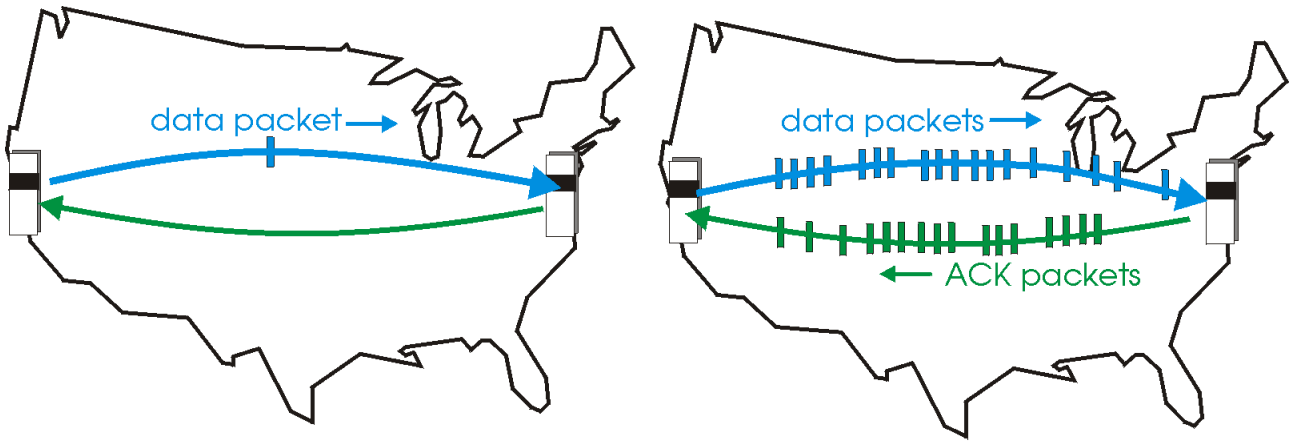


$$U_{sender} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

Πρωτόκολλα με διοχέτευση

Διοχέτευση (Pipelining): ο αποστολέας επιτρέπει πολλαπλά, "εν πτήσει", προς επιβεβαίωση πακέτα

- Το εύρος του αριθμού ακολουθίας πρέπει να αυξηθεί
- Ενταμίευση (buffering) στον αποστολέα ή/και στο δέκτη

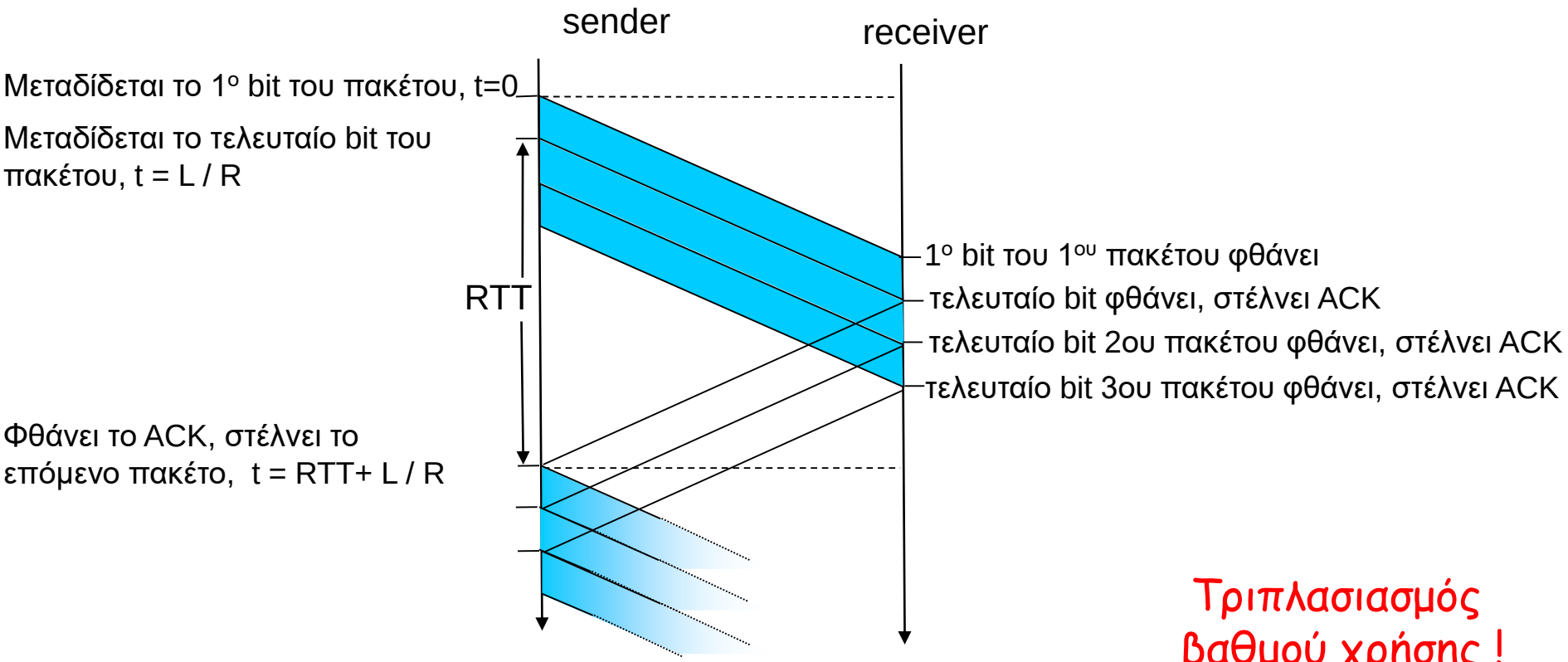


(a) a stop-and-wait protocol in operation

(b) a pipelined protocol in operation

- Δύο γενικές μορφές των πρωτοκόλλων διοχέτευσης: *go-Back-N, επιλεκτική επανάληψη (selective repeat)*

Διοχέτευση (pipelining): αύξηση βαθμού χρήσης (utilization)



Τριπλασιασμός βαθμού χρήσης !

$$U_{sender} = \frac{3 * L / R}{RTT + L / R} = \frac{.0024}{30.008} = 0.00081$$

Πρωτόκολλα διοχέτευσης

Go-back-N: γενική εικόνα

- Ο αποστολέας μπορεί να έχει έως και N μη επιβεβαιωμένα πακέτα στη διοχέτευση
- Ο δέκτης μόνο στέλνει σωρευτικά acks
  - Δεν επιβεβαιώνει πακέτο αν υπάρχει κάποιο κενό
- Ο αποστολέας έχει χρονομετρητή για το πιο παλιό μη επιβεβαιωμένο πακέτο
  - Αν ο χρονομετρητής λήξει, αναμεταδίδει όλα τα μη επιβεβαιωμένα πακέτα

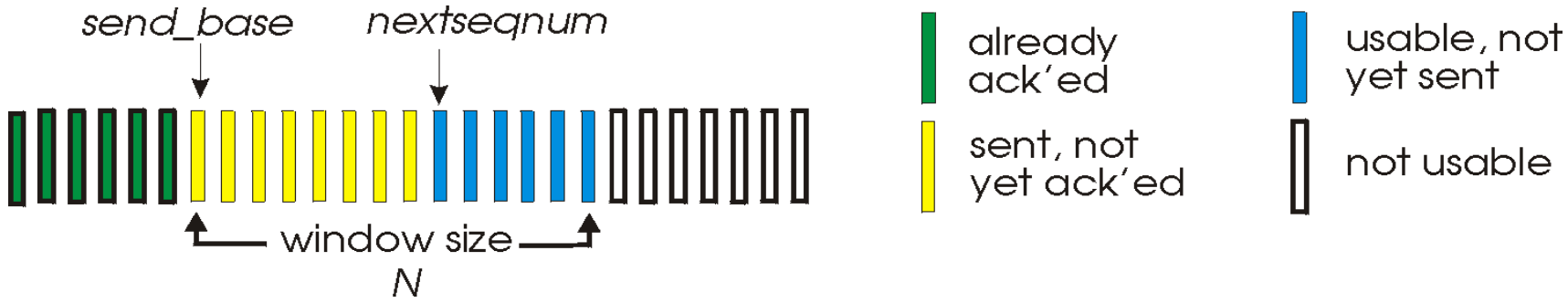
Επιλεκτική επανάληψη (Selective Repeat): γενική εικόνα

- Ο αποστολέας μπορεί να έχει έως και N μη επιβεβαιωμένα πακέτα στη διοχέτευση
- Ο δέκτης επιβεβαιώνει μεμονωμένα πακέτα
- Ο αποστολέας διατηρεί χρονομετρητή για κάθε μη επιβεβαιωμένο πακέτο
  - Όταν ο χρονομετρητής λήγει, αναμεταδίδει μόνο το μη επιβεβαιωμένο πακέτο

Go-Back-N

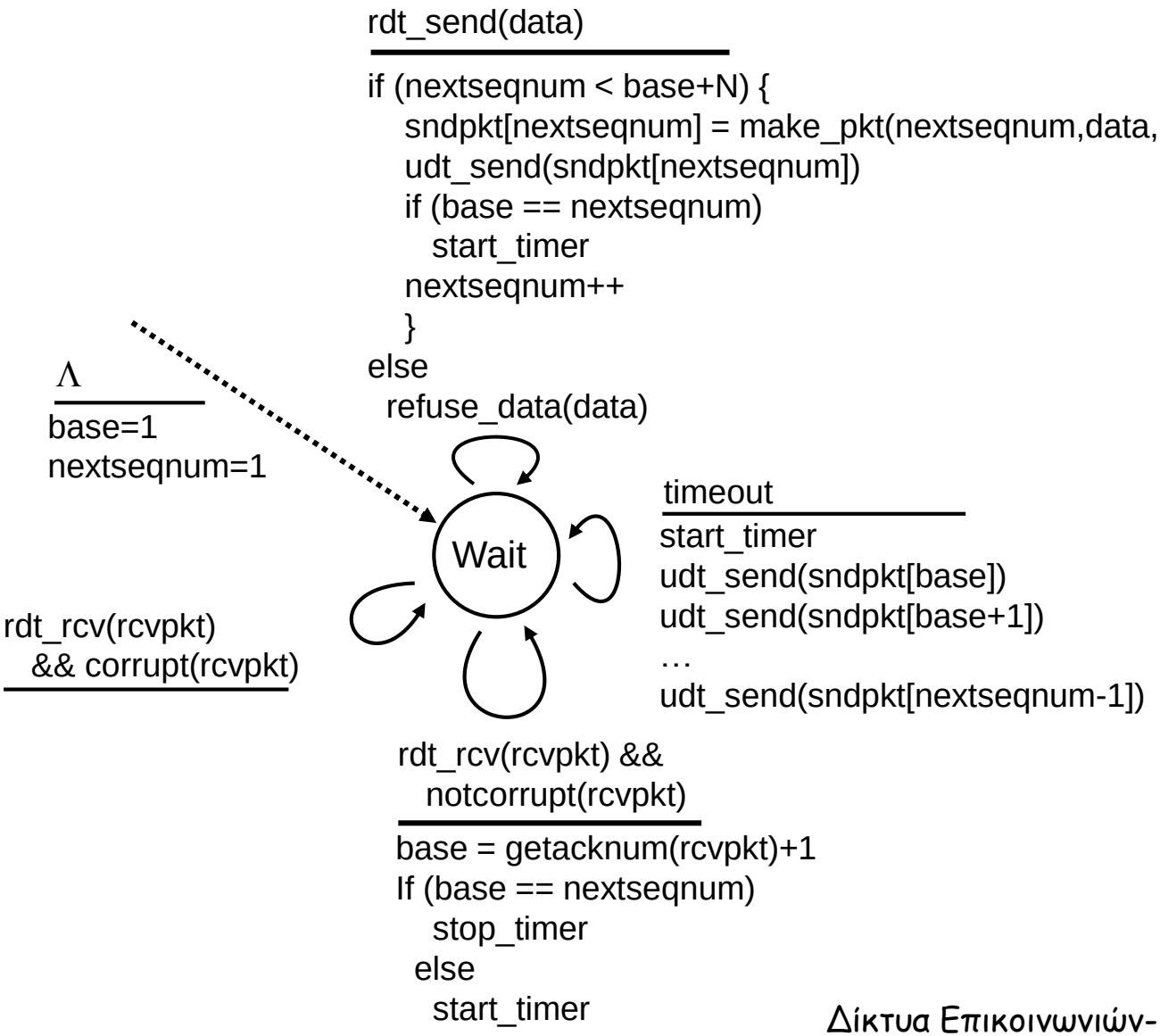
Αποστολέας:

- k-bit # ακολουθίας στην κεφαλίδα του πακέτου
- Επιτρέπεται «παράθυρο» ("window") έως και N, συνεχόμενων μη επιβεβαιωμένων πακέτων

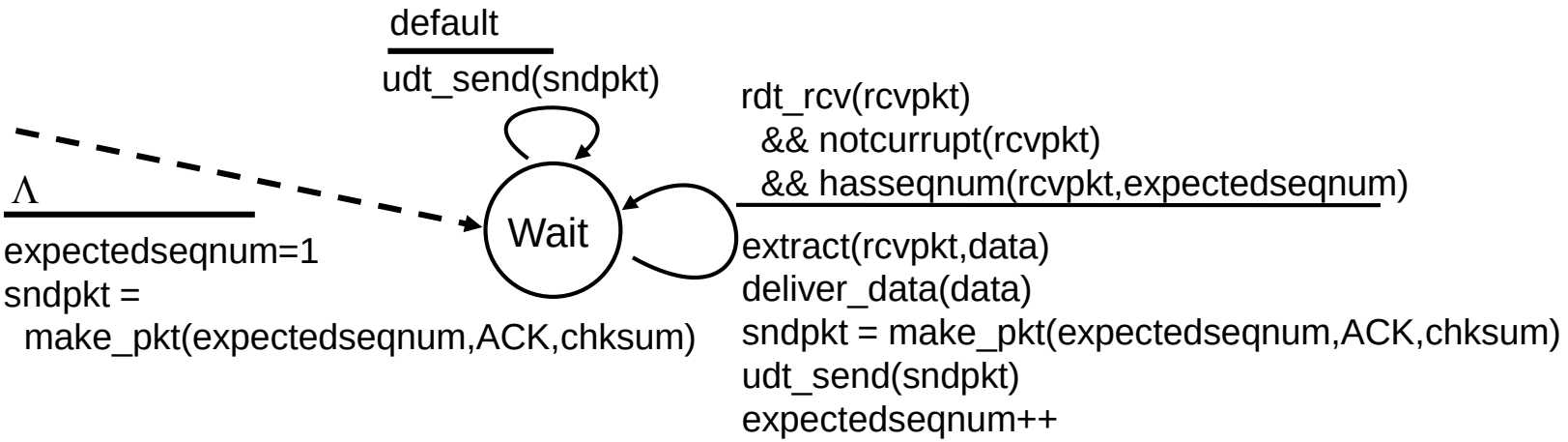


- ACK(n): επιβεβαιώνει όλα τα πακέτα έως και αυτό με # ακολουθίας n – «συσσωρευτικό ACK» ("cumulative ACK")
  - ενδέχεται να λάβει διπλά ACKs (βλέπε δέκτη)
- χρονομετρητής για το αρχαιότερο «εν πτήσει» πακέτο
- *timeout(n) (Λήξη χρόνου(n)):* αναμεταδίδει το πακέτο n και όλα τα πακέτα με υψηλότερο # ακολουθίας στο παράθυρο

GBN: Επεκτεταμένη FSM αποστολέα



GBN: Επεκτεταμένη FSM δέκτη

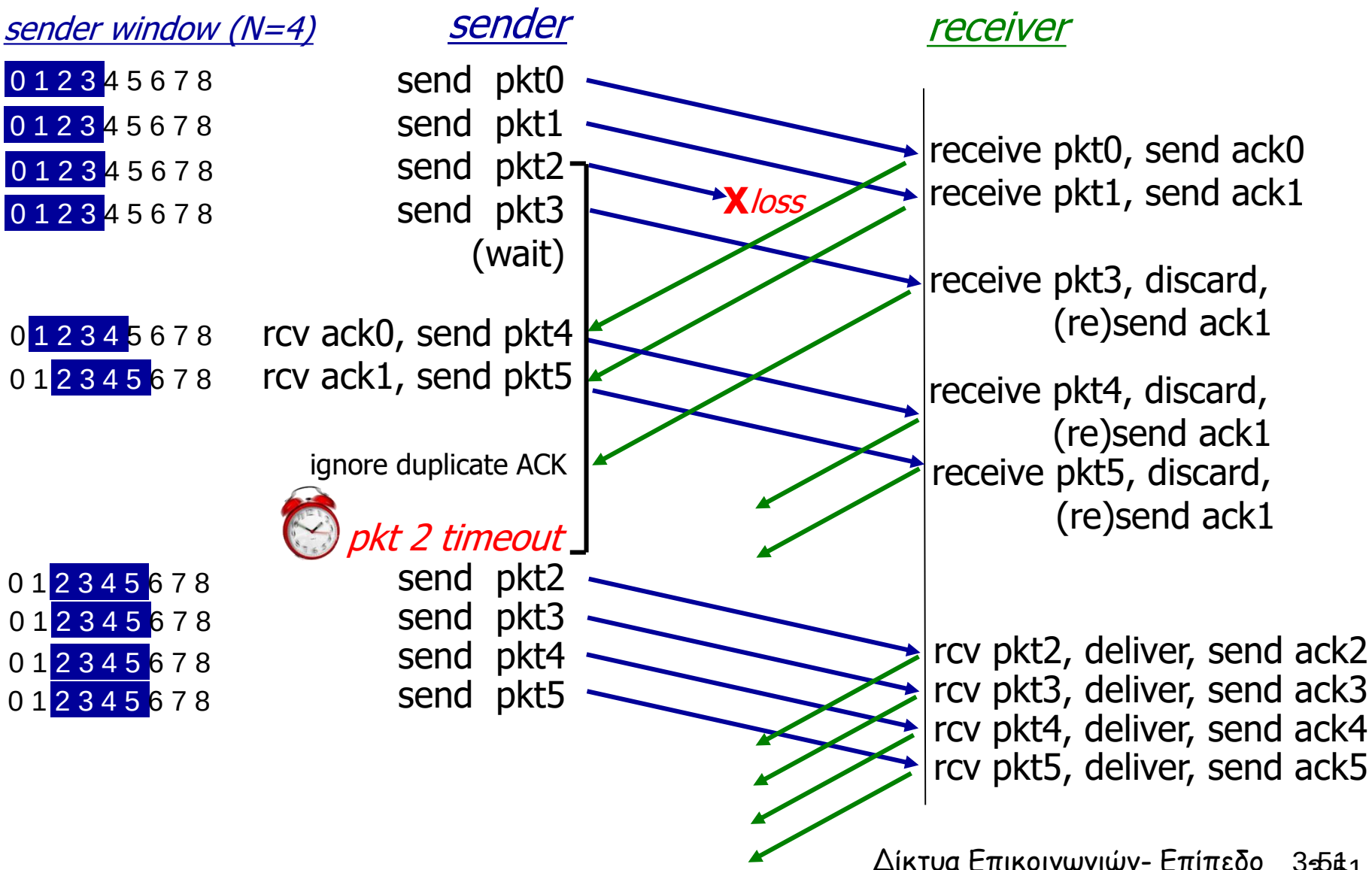


Μόνο ACK: πάντα στέλνει ACK για το πακέτο με το μέγιστο σε (ορθή) σειρά # ακολουθίας που έχει ληφθεί σωστά

- Ενδεχομένως να δημιουργήσει διπλά ACKs
- Χρειάζεται να θυμάται μόνο το **`expectedseqnum`**

- Εκτός σειράς πακέτα:
  - Απόρριψε (μην ενταμιεύεις) -> χωρίς ενταμίευση δέκτη!
  - Επανα-επιβεβαίωσε το πακέτο με το μέγιστο σε σειρά # ακολουθίας

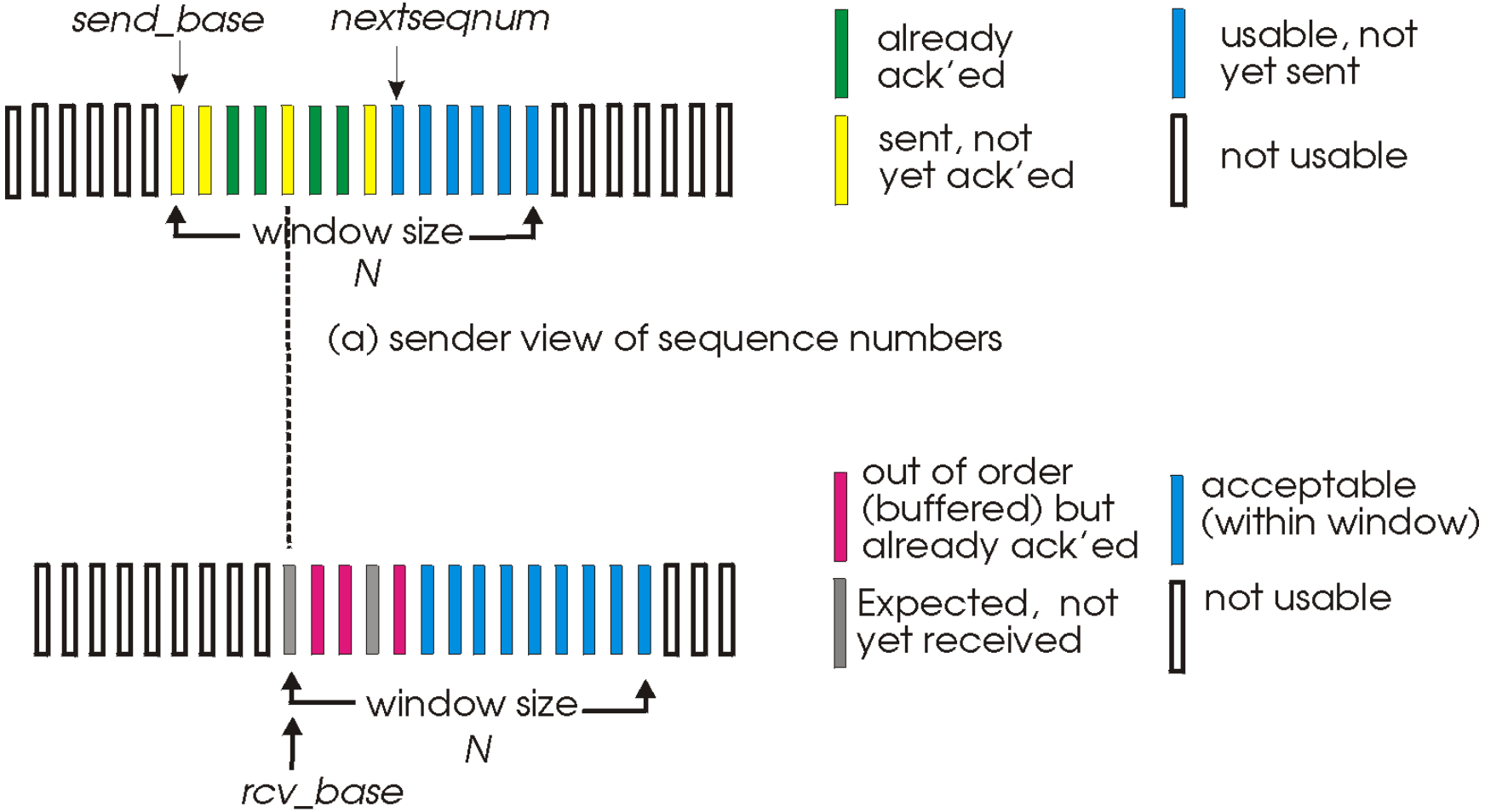
GBN «εν δράσει»



Επιλεκτική Επανάληψη (Selective Repeat)

- Ο δέκτης επιβεβαιώνει μεμονωμένα όλα τα σωστά ληφθέντα πακέτα
  - Ενταμιεύει πακέτα, αν χρειάζεται, ώστε τελικά να παραδώσει τα πακέτα σε σωστή σειρά στο ανώτερο επίπεδο
- Ο αποστολέας ξαναστέλνει μόνο τα πακέτα για τα οποία δεν έχει ληφθεί ACK
  - Χρονομετρητής στον αποστολέα για κάθε μη επιβεβαιωμένο πακέτο
- Παράθυρο αποστολέα
  - N συνεχόμενοι # ακολουθίας
  - Περιορίζει τους # ακολουθίας των σταλμένων, μη επιβεβαιωμένων πακέτων

Επιλεκτική επανάληψη: παράθυρα αποστολέα και δέκτη



Επιλεκτική επανάληψη

αποστολέας

Δεδομένα από πάνω:

- Αν είναι διαθέσιμος ο επόμενος #ακολουθίας στο παράθυρο, στείλε πακέτο

timeout(n)(Λήξη χρόνου (n)):

- Ξαναστείλε το πακέτο n, επανεκκίνησε το χρονομετρητή

ACK(n) σε [sendbase,sendbase+N]:

- Σημείωσε το πακέτο n ως ληφθέν
- Αν το n είναι το μικρότερο μη επιβεβαιωμένο πακέτο, μετακίνησε τη βάση του παραθύρου στον επόμενο μη επιβεβαιωμένο # ακολουθίας

δέκτης

πακέτο n στο [rcvbase, rcvbase+N-1]

- στείλε ACK(n)
- Εκτός σειράς: ενταμίευσε
- Σε σειρά: παράδωσε (επίσης παράδωσε τα ενταμιευμένα, σε σειρά πακέτα), μετακίνησε το παράθυρο στο επόμενο πακέτο που δεν έχει ληφθεί ακόμα

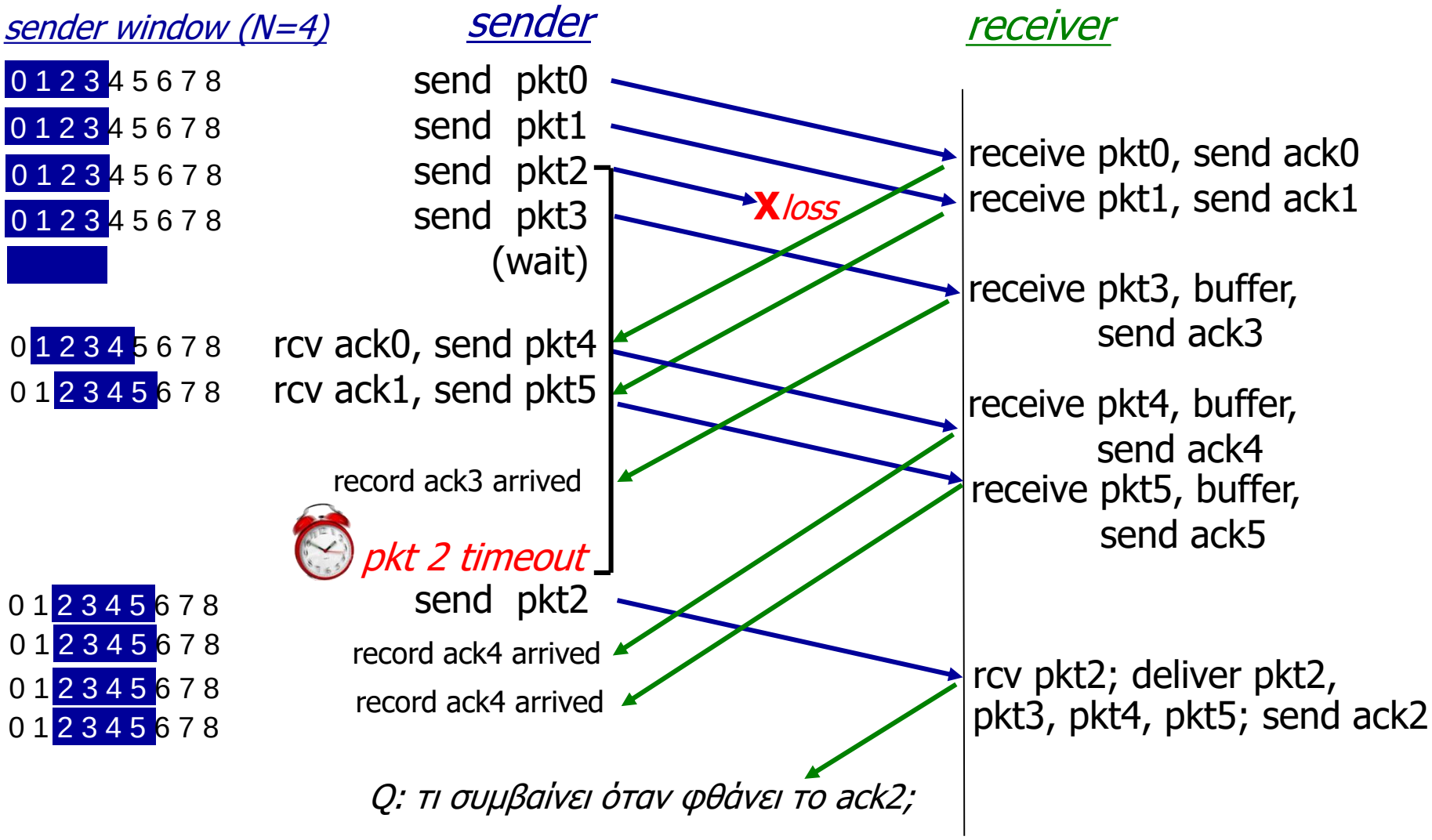
πακέτο n στο [rcvbase-N,rcvbase-1]

- ACK(n)

διαφορετικά:

- αγνόησε

Επιλεκτική επανάληψη «εν δράσει»



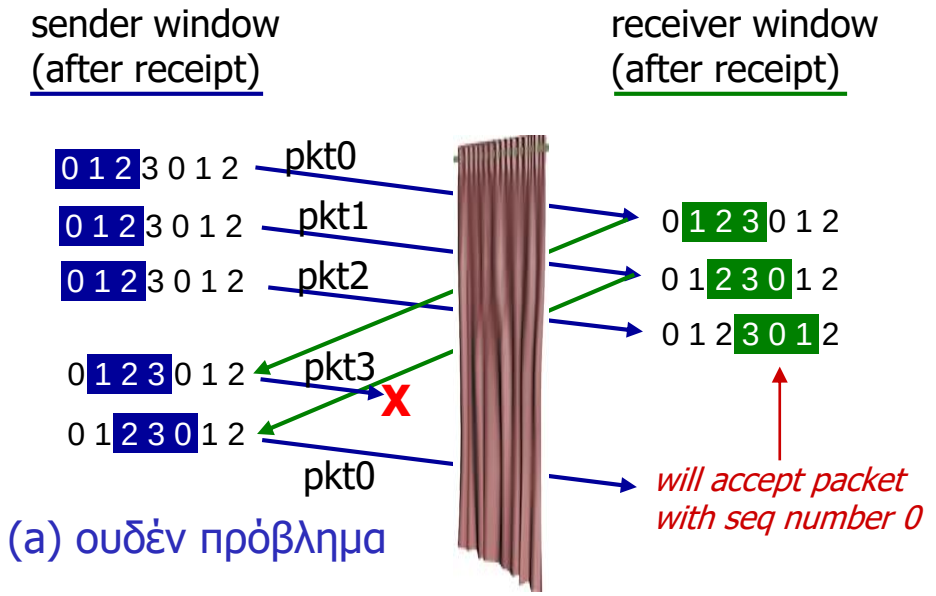
Επιλεκτική επανάληψη: δίλημμα

Παράδειγμα:

- # ακολουθίας: 0, 1, 2, 3
- Μέγεθος παραθύρου=3

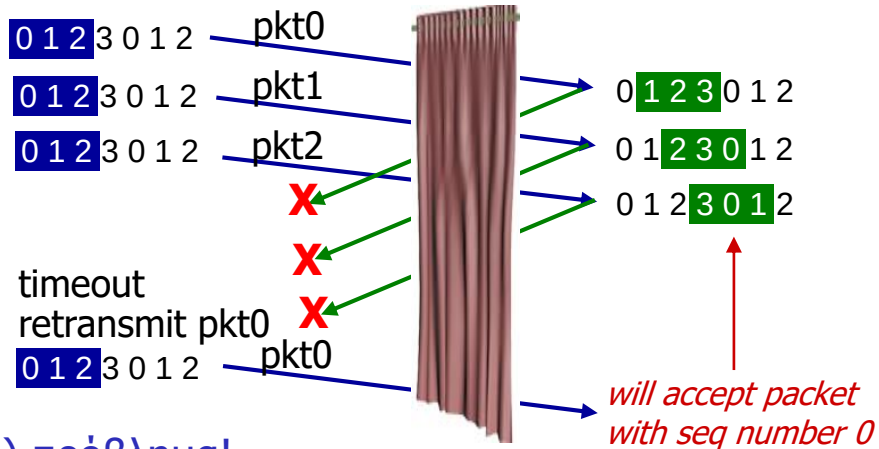
- Ο δέκτης δε βλέπει διαφορά στα δύο σενάρια!
- Τα διπλά (duplicate) δεδομένα γίνονται δεκτά σαν νέα στο (b)

Ε: Ποιά η σχέση μεταξύ μεγέθους #ακολουθίας και μεγέθους παραθύρου;



(a) ουδέν πρόβλημα

Ο δέκτης δεν μπορεί να δει την πλευρά του αποστολέα.
Η συμπεριφορά του δέκτη ίδια και στις 2 περιπτώσεις!
κάτι πάει (πολύ) στραβά!



(b) πρόβλημα!

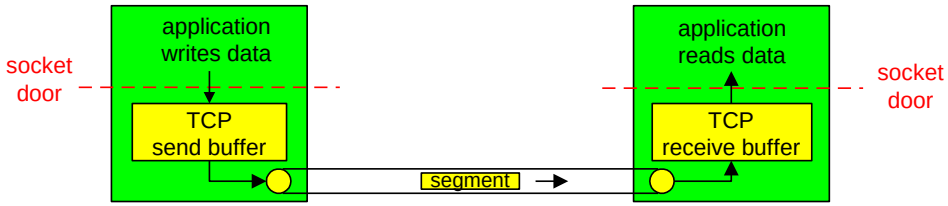
Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

TCP: Επισκόπηση

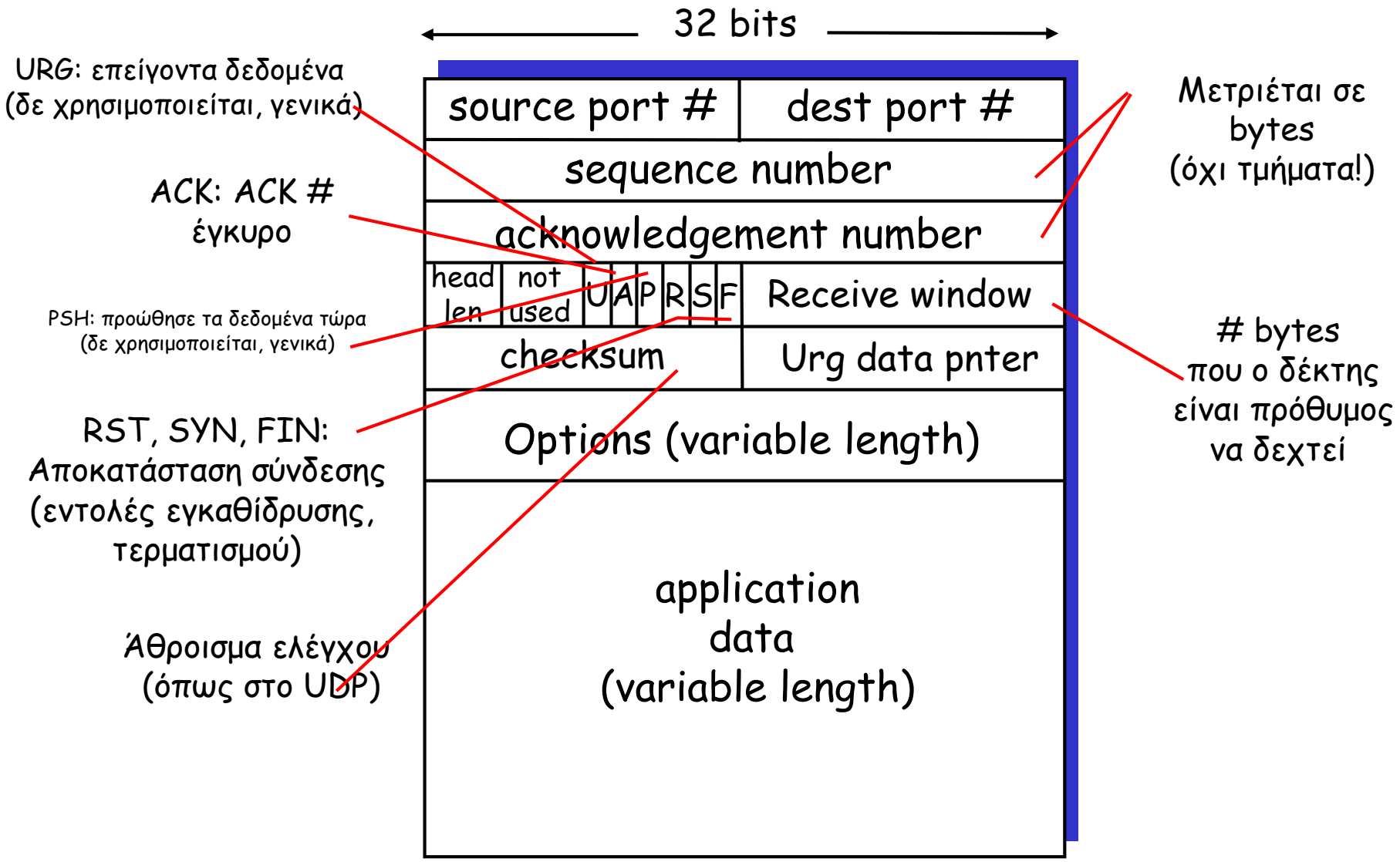
RFCs: 793, 1122, 1323, 2018, 2581

- Από σημείο προς σημείο:
  - Ένας αποστολέας, ένας δέκτης
- Αξιόπιστη, σε σειρά ροή από bytes:
  - Χωρίς "όρια μηνυμάτων"
- Με διοχέτευση:
  - Ο έλεγχοι συμφόρησης και ροής του TCP καθορίζουν το μέγεθος του παραθύρου
- Ενταμιευτές αποστολής και λήψης



- Πλήρως αμφίδρομα δεδομένα:
  - Δικατευθυντική ροή δεδομένων στην ίδια σύνδεση
  - MSS: maximum segment size (μέγιστο μέγεθος τμήματος)
- Συνδεσμική:
  - Η χειραψία (handshaking) (ανταλλαγή μηνυμάτων ελέγχου) αρχικοποιεί την κατάσταση του αποστολέα και του δέκτη πριν την ανταλλαγή δεδομένων
- Ροή υπό έλεγχο:
  - Ο αποστολέας δεν θα υπερφορτώσει το δέκτη

Δομή τμήματος TCP



TCP: αριθμοί ακολουθίας και ACKs

Αριθμοί ακολουθίας (Seq. #'s):

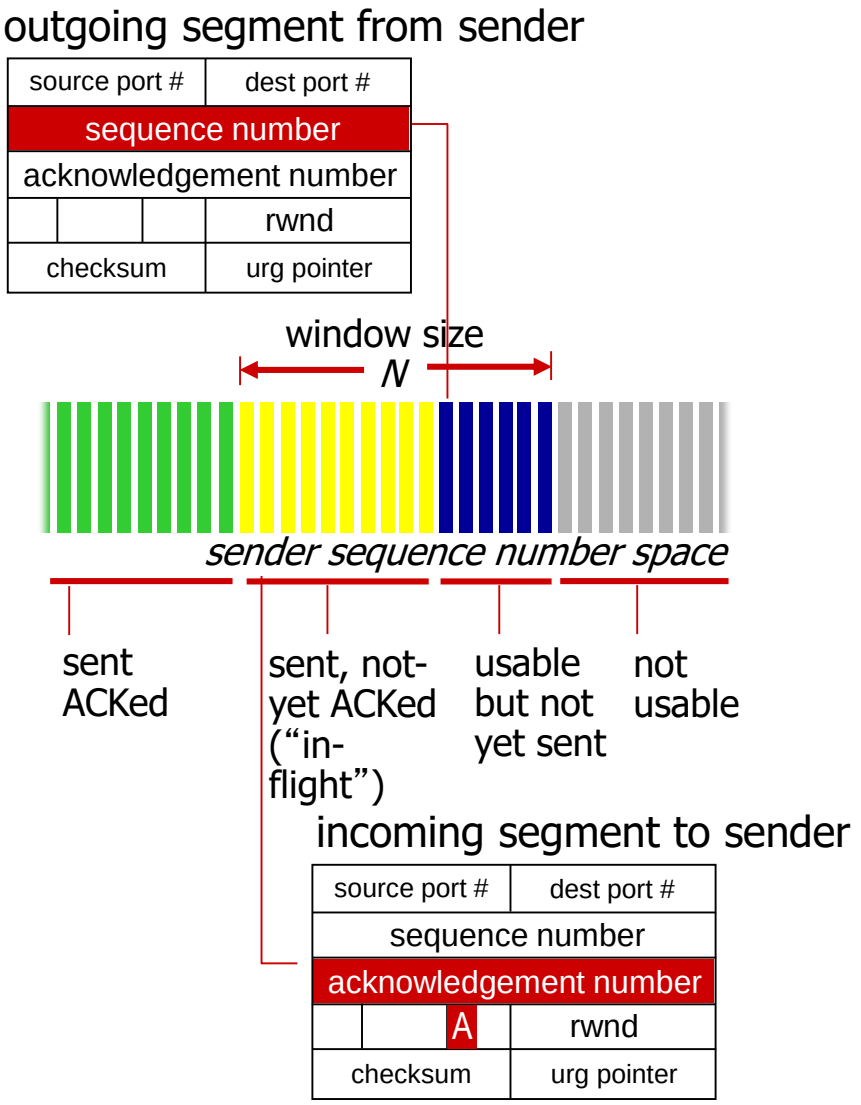
- Αριθμός του πρώτου byte των δεδομένων του τμήματος

ACKs:

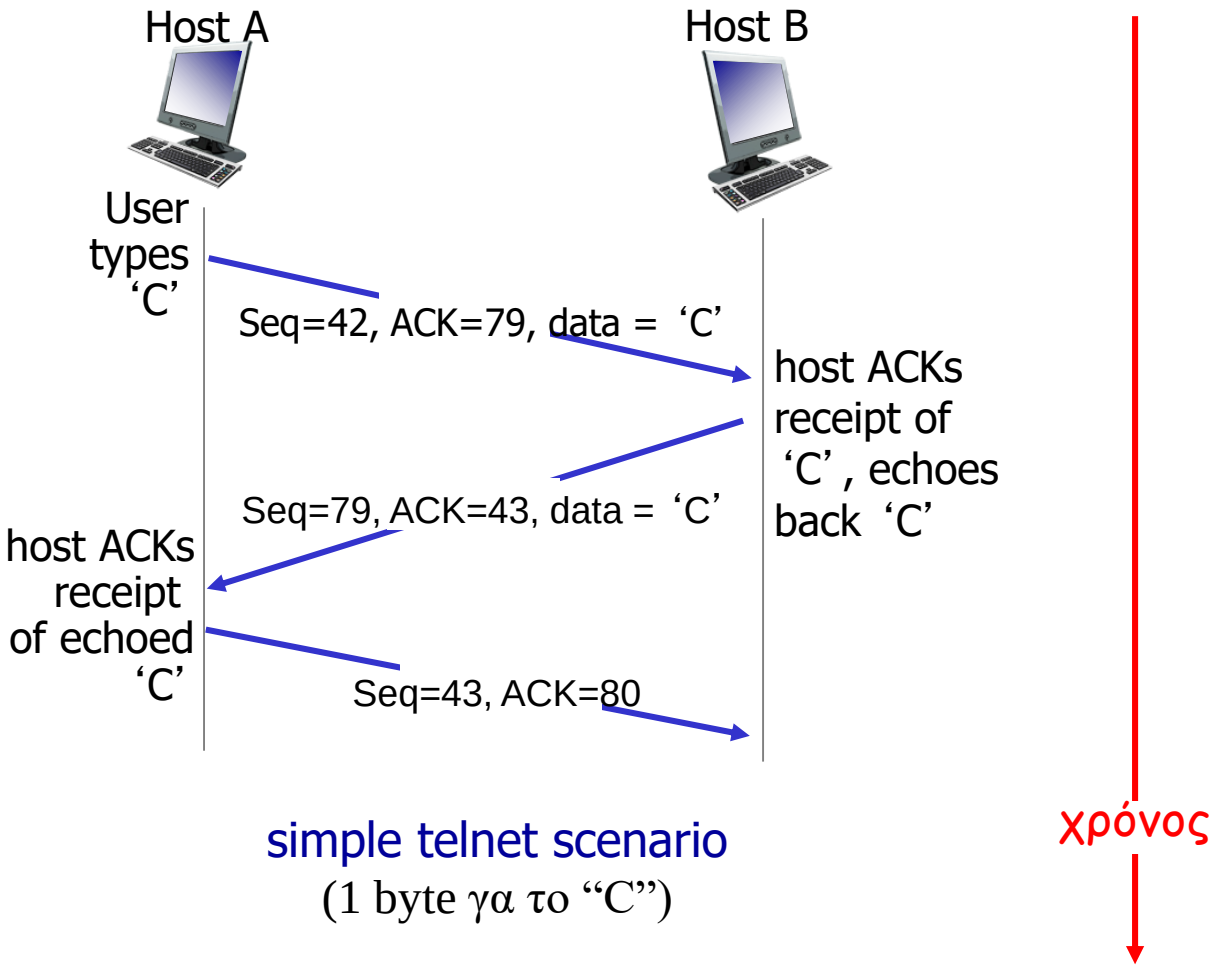
- seq # του επόμενου byte που αναμένεται από την άλλη πλευρά
- συσσωρευτικά ACK

Ε: πώς διαχειρίζεται ο δέκτης τα τμήματα εκτός σειράς;

- Α: η προδιαγραφή του TCP δεν καθορίζει (εναπόκειται στην υλοποίηση)



TCP αριθμοί ακολουθίας, ACKs



simple telnet scenario
(1 byte για το "C")

Χρόνος Διαδρομής μετ' επιστροφής (Round Trip Time) και Λήξη Χρόνου (Timeout) του TCP

- Ε: Πώς καθορίζεται η τιμή του timeout (λήξη χρόνου) του TPC;
- Μεγαλύτερο από RTT
  - αλλά το RTT μεταβάλλεται
- Πολύ σύντομο: πρώιμο timeout
  - μη απαραίτητες αναμεταδόσεις
- μεγάλης διάρκειας: αργή αντίδραση σε απώλεια τμήματος

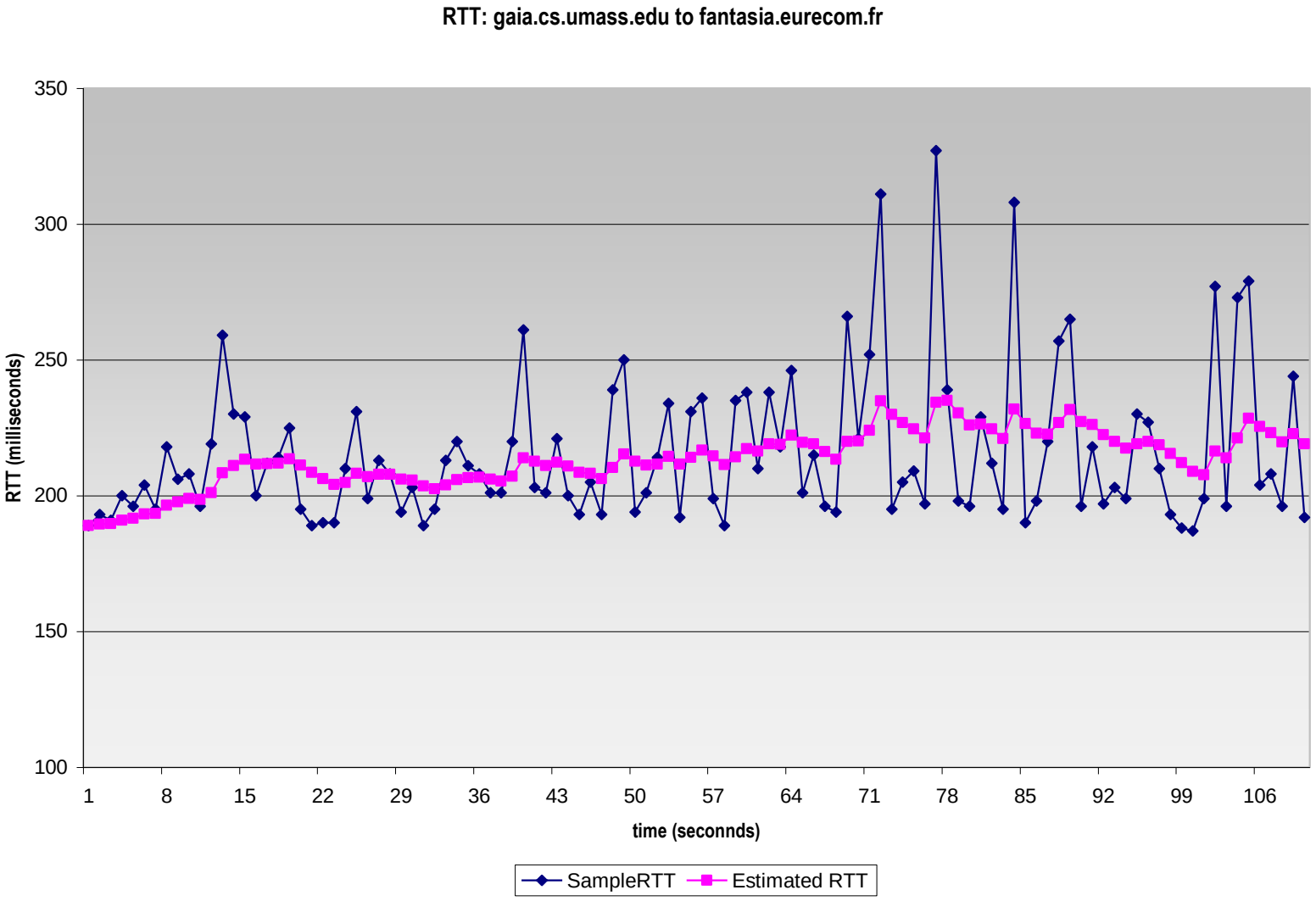
- Ε: Πώς εκτιμάται το RTT;
- **SampleRTT**: χρόνος που μετριέται από τη μετάδοση του τμήματος ως την παραλαβή του ACK
  - αγνοούνται οι αναμεταδόσεις
- **Το SampleRTT** θα μεταβάλλεται, θέλουμε το εκτιμώμενο RTT πιο "ομαλό"
  - μέσος όρος αρκετών πρόσφατων μετρήσεων, όχι μόνο του τρέχοντος SampleRTT

Χρόνος Διαδρομής Μετ' επιστροφής (Round Trip Time) και Λήξη Χρόνου (Timeout) του TCP

EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT

- Εκθετική σταθμισμένη κινητή μέση τιμή (Exponentially weighted moving average)
- η επίδραση των παλαιών δειγμάτων μειώνεται εκθετικά
- τυπική τιμή: α = 0.125

Παράδειγμα εκτίμησης του RTT:



Χρόνος Διαδρομής Μετ' επιστροφής (Round Trip Time) και Λήξη Χρόνου (Timeout) του TCP

Καθορισμός του timeout

EstimatedRTT συν "περιθώριο ασφαλείας"

- **μεγάλη μεταβολή στο EstimatedRTT -> μεγαλύτερο περιθώριο ασφαλείας**
- **πρώτα εκτιμάται πόσο αποκλίνει το SampleRTT από το EstimatedRTT:**

$$DevRTT = (1-\beta)*DevRTT + \beta*|SampleRTT-EstimatedRTT|$$

(τυπικά, β = 0.25)

Μετά καθορίζεται η τιμή του timeout:

$$TimeoutInterval = EstimatedRTT + 4*DevRTT$$

estimated RTT

"safety margin"

Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

Αξιόπιστη μεταφορά δεδομένων (αμδ) του TCP

- Το TCP δημιουργεί υπηρεσία «αμδ» πάνω από την αναξιόπιστη υπηρεσία του IP
  - Τμήματα σε διοχέτευση
  - Σωρευτικά acks
  - Το TCP χρησιμοποιεί ένα μόνο χρονομετρητή αναμεταδόσεων
- Αναμεταδόσεις προκαλούνται από
  - συμβάντα λήξης χρόνου (timeouts)
  - διπλές επιβεβαιώσεις (duplicate ACKs)
- Αρχικά θεωρούμε απλοποιημένο αποστολέα TCP:
  - αγνοούνται διπλά ACKS
  - αγνοείται έλεγχος ροής, έλεγχος συμφόρησης

Γεγονότα του αποστολέα TCP:

Λήψη δεδομένων από εφαρμογή:

- δημιουργία τμήματος με # ακολουθίας
- # ακολουθίας είναι ο αριθμός του πρώτου byte δεδομένων στο τμήμα
- εκκίνηση χρονομετρητή αν δεν τρέχει ήδη (ο χρονομετρητής είναι σαν το χρονομετρητή του πιο παλιού μη επιβεβαιωμένου τμήματος)
- διάστημα λήξης: **`TimeOutInterval`**

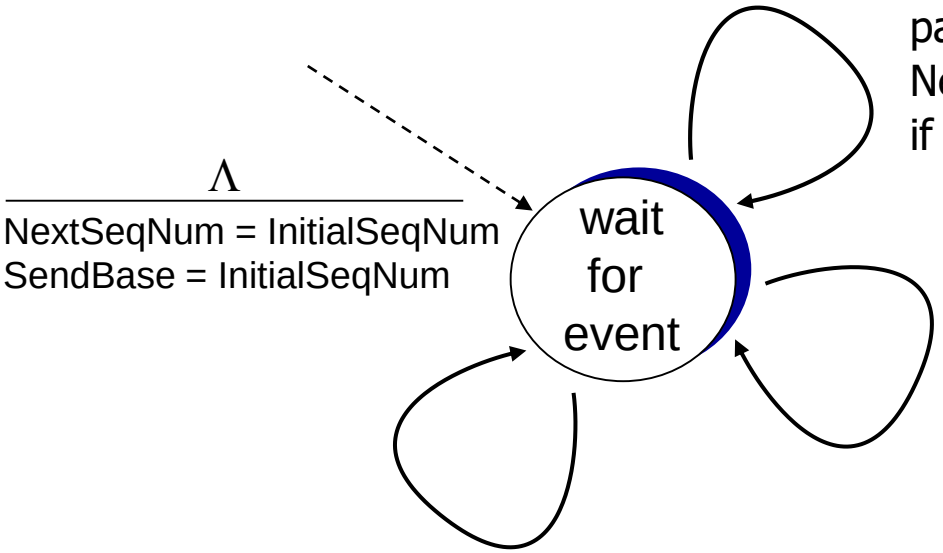
Λήξη χρόνου (timeout):

- επαναμετάδοση του τμήματος που προκάλεσε το timeout
- επανεκκίνηση χρονομετρητή

Λήψη ACK:

- αν επιβεβαιώνει τμήματα που δεν έχουν ήδη επιβεβαιωθεί
  - ανανέωση του τι είναι γνωστό ότι έχει επιβεβαιωθεί
  - εκκίνηση χρονομετρητή αν εξακολουθούν να υπάρχουν τμήματα

Αποστολέας TCP (απλοποιημένος)



data received from application above

create segment, seq. #: NextSeqNum
pass segment to IP (i.e., "send")
NextSeqNum = NextSeqNum + length(data)
if (timer currently not running)
    start timer

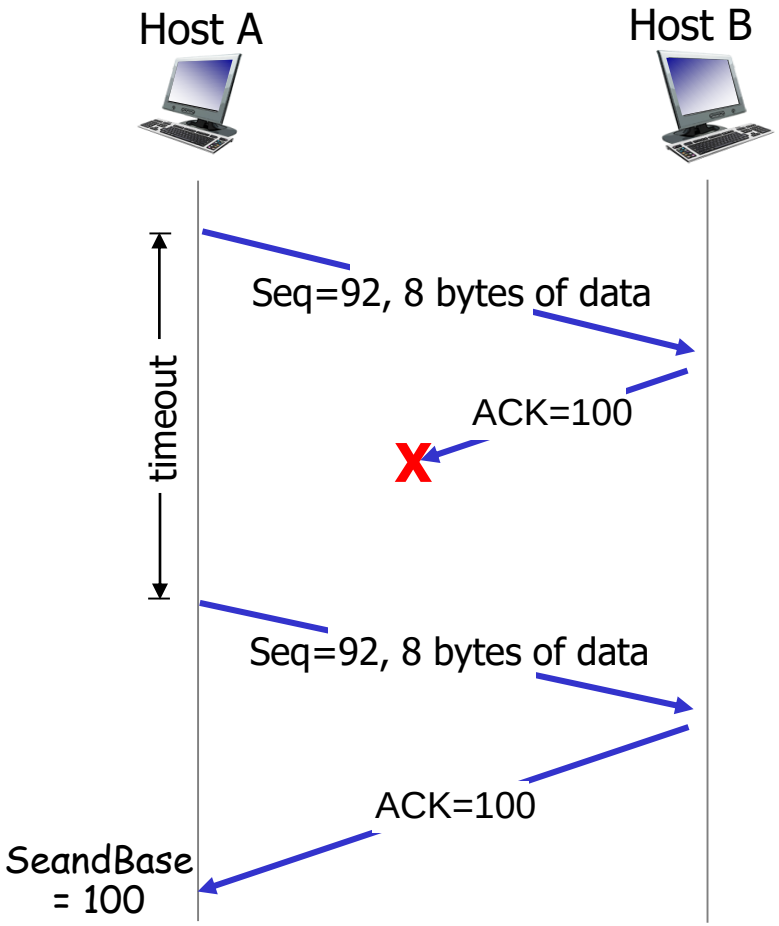
timeout

retransmit not-yet-acked segment
        with smallest seq. #
start timer

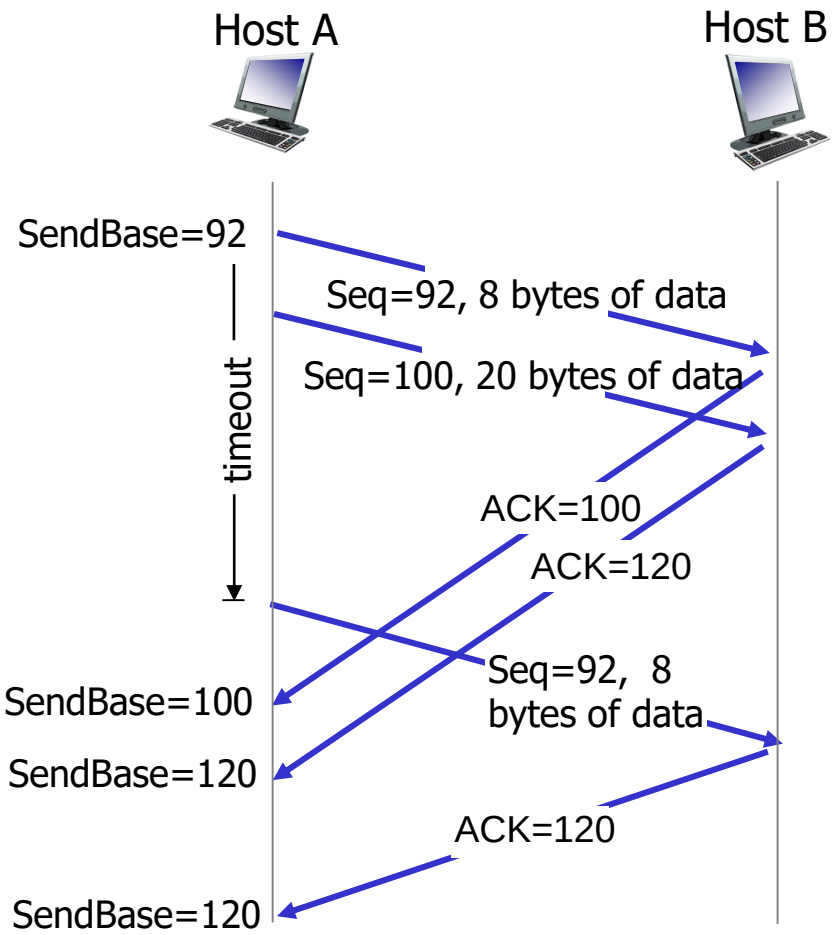
ACK received, with ACK field value y

```
if (y > SendBase) {
   SendBase = y
   /* SendBase–1: last cumulatively ACKed byte */
   if (there are currently not-yet-acked segments)
       start timer
     else stop timer
    }
```

Σενάρια αναμεταδόσεων TCP

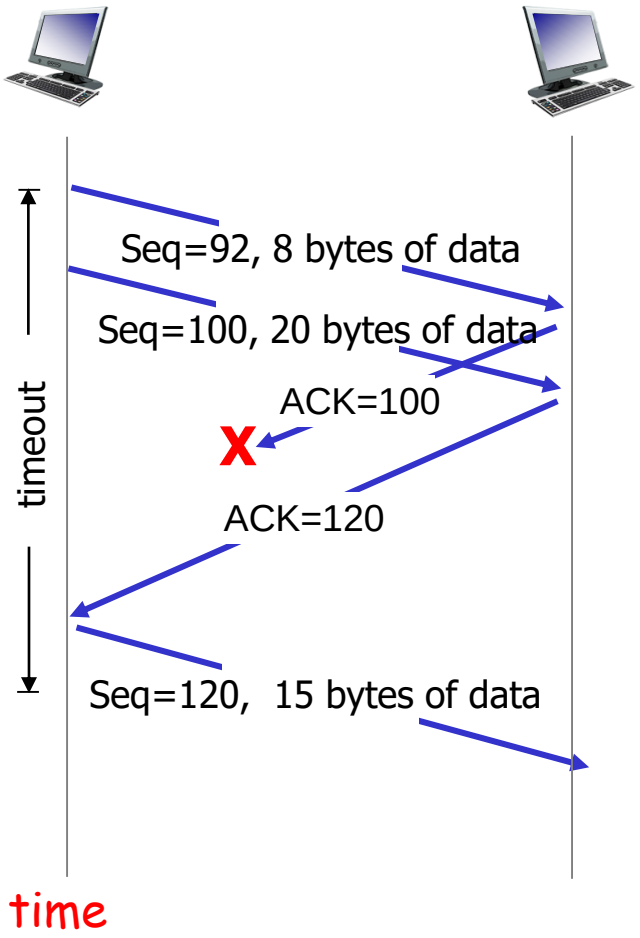


Σενάριο χαμένου ACK



Πρώιμη λήξη χρόνου
(premature timeout)

Σενάρια αναμεταδόσεων TCP (συν.)



Σενάριο συσσωρευτικού ACK

Παραγωγή TCP ACK [RFC 1122, RFC 2581]

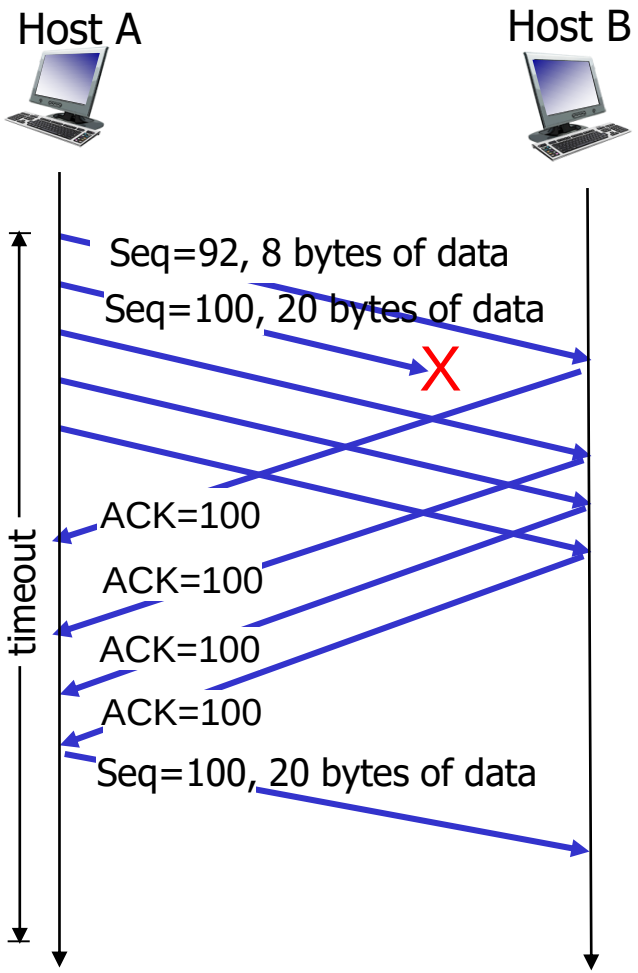
Συμβάν στο δέκτη	Ενέργεια δέκτη TCP
Άφιξη τμήματος σε σειρά με αναμενόμενο # ακολουθίας. Όλα τα δεδομένα μέχρι τον αναμενόμενο # ακολουθίας έχουν επιβεβαιωθεί	Καθυστερημένο ACK. Αναμονή 500ms για το επόμενο τμήμα. Αν όχι επόμενο τμήμα στείλε ACK
Άφιξη τμήματος σε σειρά με αναμενόμενο # ακολουθίας. Ένα άλλο τμήμα περιμένει για μετάδοση ACK	Άμεση αποστολή ενός συσσωρευτικού ACK που κάνει επιβεβαίωση και για τα δύο τμήματα που έφτασαν σε σειρά
Άφιξη τμήματος εκτός σειράς με μεγαλύτερο του αναμενόμενου # ακολουθίας. Ανίχνευση κενού	Άμεση αποστολή *duplicate ACK* που δηλώνει # ακολουθίας επόμενου αναμενόμενου byte
Άφιξη τμήματος που μερικώς ή πλήρως συμπληρώνει κενό στα ληφθέντα δεδομένα	Άμεση αποστολή ACK, αρκεί το τμήμα αυτό να αρχίζει στο κάτω άκρο του κενού

Ταχεία αναμετάδοση (Fast retransmit)

- Το διάστημα λήξης χρόνου είναι συχνά σχετικά μεγάλο:
  - μεγάλη καθυστέρηση πριν ξανασταλεί το χαμένο πακέτο
- Ανίχνευση χαμένων τμημάτων με duplicate ACKs
  - Ο αποστολέας συχνά στέλνει πολλά τμήματα το ένα πίσω από το άλλο
  - Αν ένα τμήμα χαθεί, θα υπάρξουν πιθανώς πολλά διπλά ACKs.

- TCP ταχεία αναμετάδοση:
  Εάν ο αποστολέας λάβει 3 duplicate ACK για τα ίδια δεδομένα, ξαναστέλνει το μη επιβεβαιωμένο τμήμα με το μικρότερο αριθμό ακολουθίας.
  - Πιθανότατα το μη επιβεβαιωμένο πακέτο έχει χαθεί, οπότε μην περιμένεις τη λήξη του χρονομετρητή

Ταχεία Αναμετάδοση TCP



Ταχεία αναμετάδοση μετά τη λήψη από
τον αποστολέα του τριπλού διπλότυπου ACK

Αλγόριθμος ταχείας αναμετάδοσης:

```
event: ACK received, with ACK field value of y
            if (y > SendBase) {
                SendBase = y
                if (there are currently not-yet-acknowledged segments)
                    start timer
              }
            else {
                increment count of dup ACKs received for y
                if (count of dup ACKs received for y = 3) {
                    resend segment with sequence number y
                 }
```

Ένα duplicate ACK για ένα ήδη επιβεβαιωμένο τμήμα

Ταχεία αναμετάδοση

Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

Έλεγχος ροής του TCP

- Η πλευρά του δέκτη της TCP σύνδεσης διαθέτει ενταμιευτή λήψης (receive buffer)

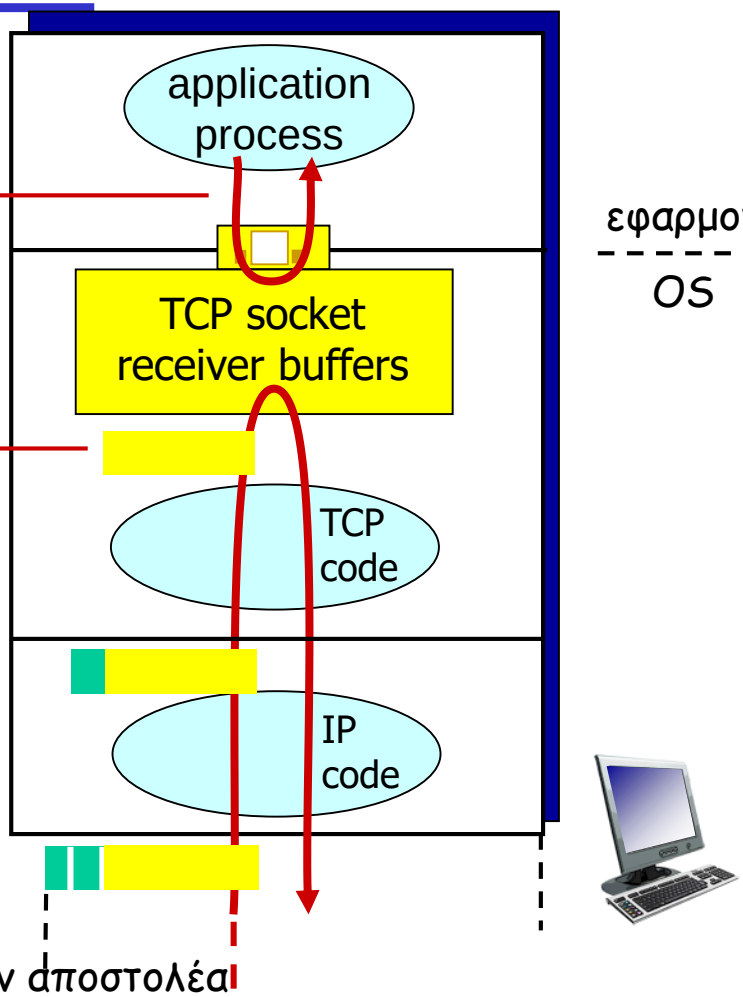
- Η διεργασία της εφαρμογής ενδέχεται να είναι αργή στην ανάγνωση

- Υπηρεσία ταιριάσματος ταχύτητας: ταίριασμα του ρυθμού αποστολής με το ρυθμό που η εφαρμογή αντλεί τα δεδομένα

Έλεγχος ροής του TCP

η εφαρμογή μπορεί να απομακρύνει δεδομένα από τους ενταμιευτές της TCP socket ….

ο αποστολέας στέλνει… πιο αργά απ' ότι παραδίδει ο TCP δέκτης

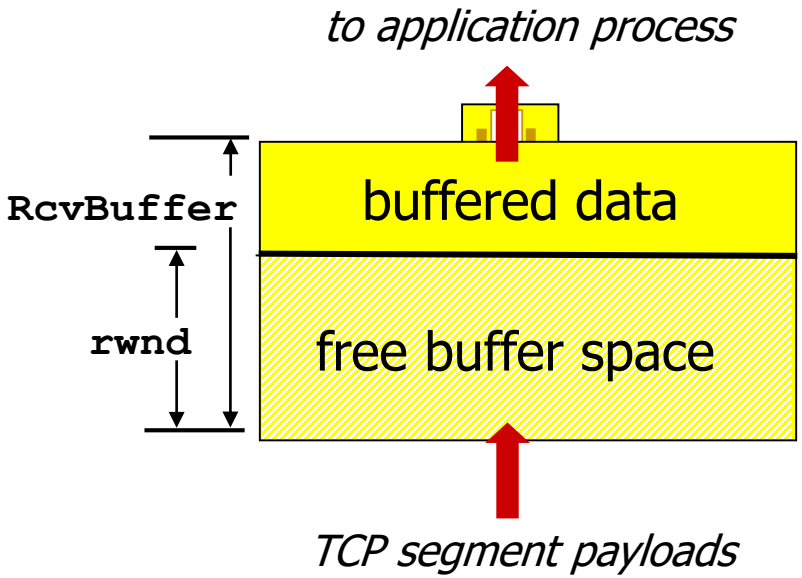


στοίβα πρωτοκόλλων δέκτη

Έλεγχος ροής

ο δέκτης ελέγχει τον αποστολέα, έτσι ο αποστολέας δεν υπερχειλίζει τον ενταμιευτή του δέκτη στέλνοντας με υπερβολικά υψηλό ρυθμό

Έλεγχος ροής του TCP



receiver-side buffering

- Ο δέκτης κοινοποιεί τον ελεύθερο χώρο του ενταμιευτή περιλαμβάνοντας την τιμή rwnd στην TCP κεφαλίδα των τμημάτων από το δέκτη προς στον αποστολέα
  - το μέγεθος του **RcvBuffer** ορίζεται μέσω των επιλογών του socket (προκαθορισμένη τιμή 4096 bytes)
  - πολλά λειτουργικά συστήματα ρυθμίζουν αυτόματα το RcvBuffer
- Ο αποστολέας περιορίζει τα μη επιβεβαιωμένα δεδομένα στην τιμή rwnd του δέκτη
- εγγυάται ότι ο ενταμιευτής στον δέκτη δεν υπερχειλίζει

Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

Διαχείριση σύνδεσης TCP

Θυμηθείτε: αποστολέας, δέκτης του TCP αποκαθιστούν «σύνδεση» πριν ανταλλάξουν τμήματα δεδομένων

- Αρχικοποίηση μεταβλητών TCP:
  - αριθμοί ακολουθίας
  - ενταμιευτές, πληροφορία ελέγχου ροής (π.χ. **RcvWindow**)
- *Πελάτης(client): αυτός που εκκινεί τη σύνδεση*

```
Socket clientSocket = new
 Socket("hostname","port
 number")
```

- *Εξυπηρέτης (server):* έρχεται σε επαφή μαζί του ο πελάτης

```
Socket connectionSocket =
 welcomeSocket.accept()
```

Τριμερής Χειραψία (Three way handshake):

Βήμα 1: Ο υπολογιστής πελάτης του TCP στέλνει τμήμα SYN στον εξυπηρέτη
- καθορίζει αρχικό # ακολουθίας
- χωρίς δεδομένα

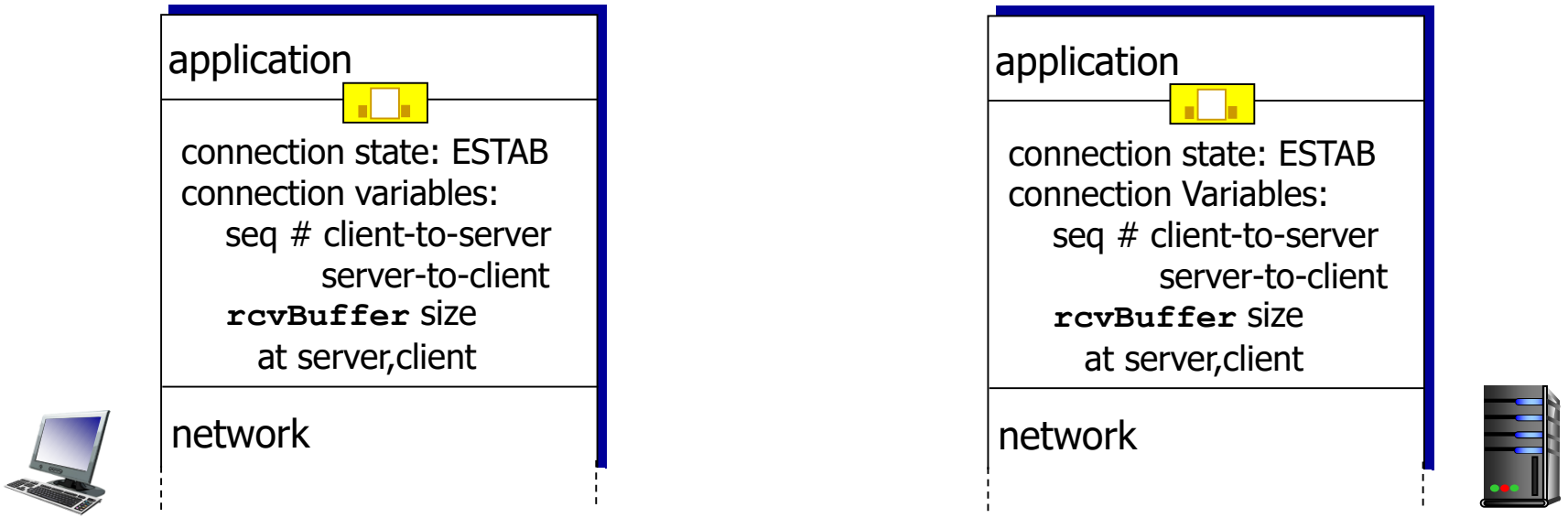
Βήμα 2: Ο υπολογιστής εξυπηρέτης λαμβάνει SYN, απαντά με τμήμα SYNACK
- ο εξυπηρέτης δεσμεύει ενταμιευτές
- καθορίζει αρχικό # ακολουθίας εξυπηρέτη

Βήμα 3: Ο πελάτης λαμβάνει SYNACK, απαντά με τμήμα ACK που μπορεί να περιέχει δεδομένα

Διαχείριση σύνδεσης TCP (συν.)

Πριν γίνει ανταλλαγή δεδομένων, ο αποστολέας κι ο δέκτης κάνουν "χειραψία":

- συμφωνούν για τη δημιουργία σύνδεσης (ο καθένας να γνωρίζει ότι ο άλλος είναι πρόθυμος να δημιουργήσει τη σύνδεση)
- συμφωνούν στις παραμέτρους της σύνδεσης

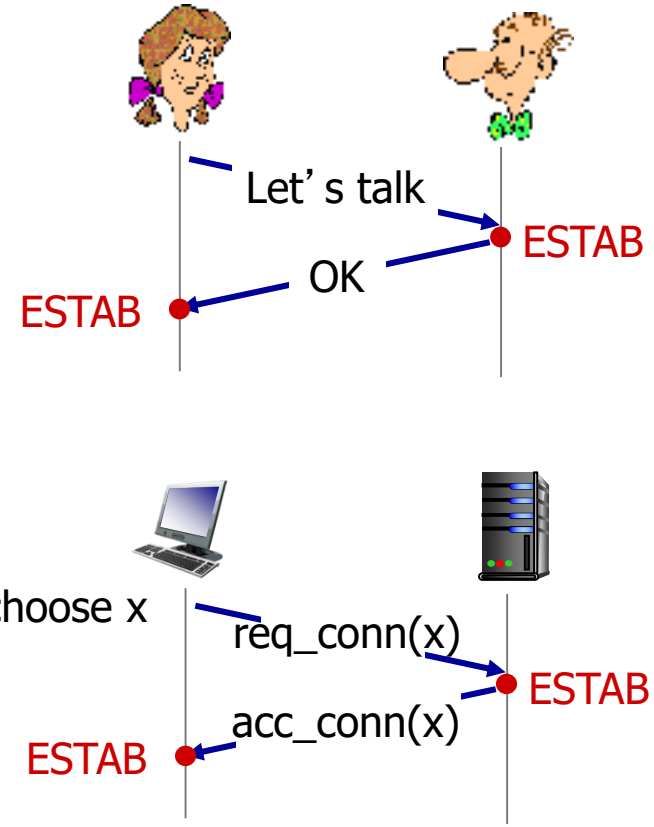



```
Socket clientSocket =
  newSocket("hostname","port
   number");
```

```
Socket connectionSocket =
  welcomeSocket.accept();
```

Συμφωνία για δημιουργία σύνδεσης

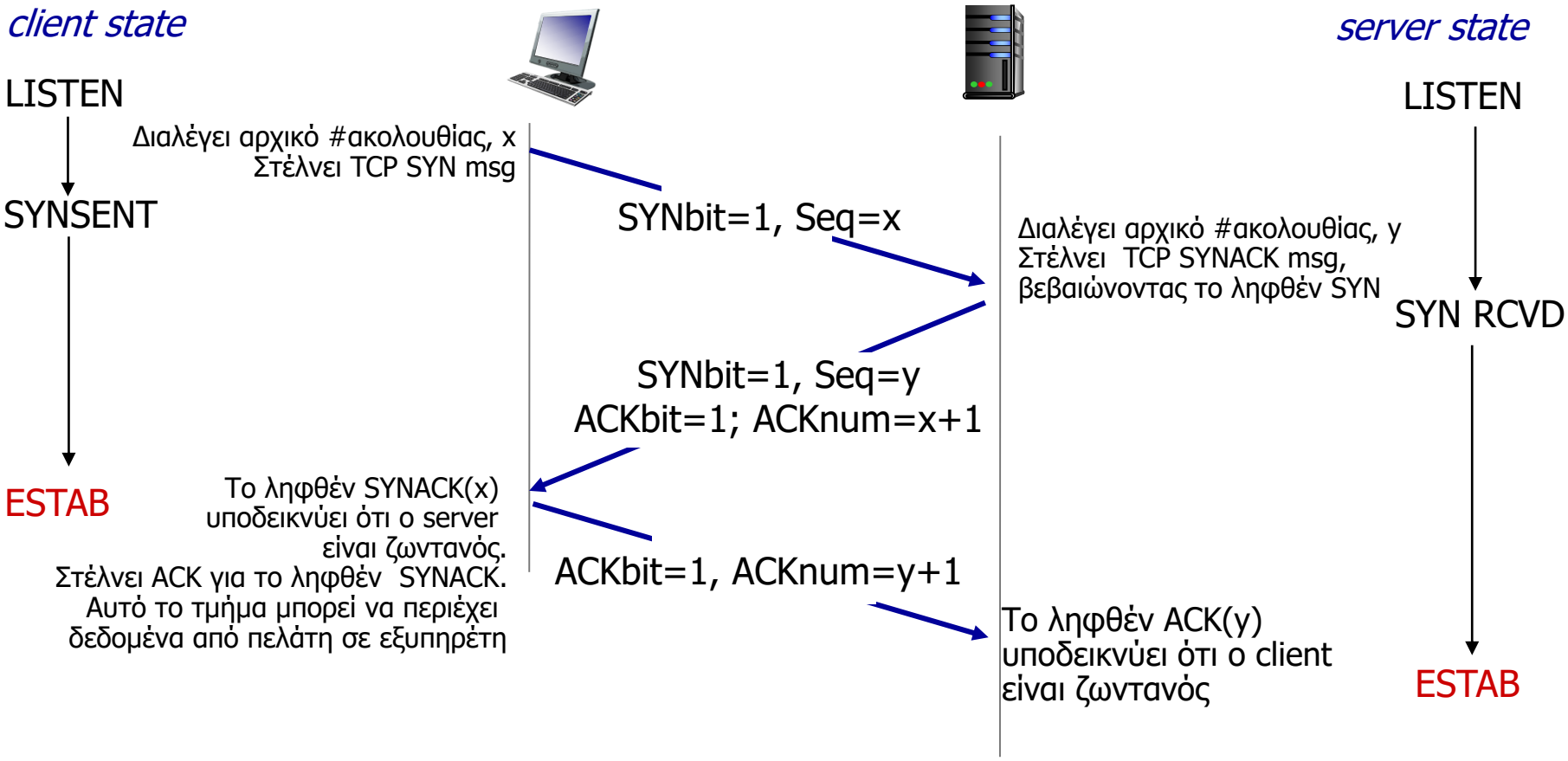
Διμερής χειραψία (2-way handshake):



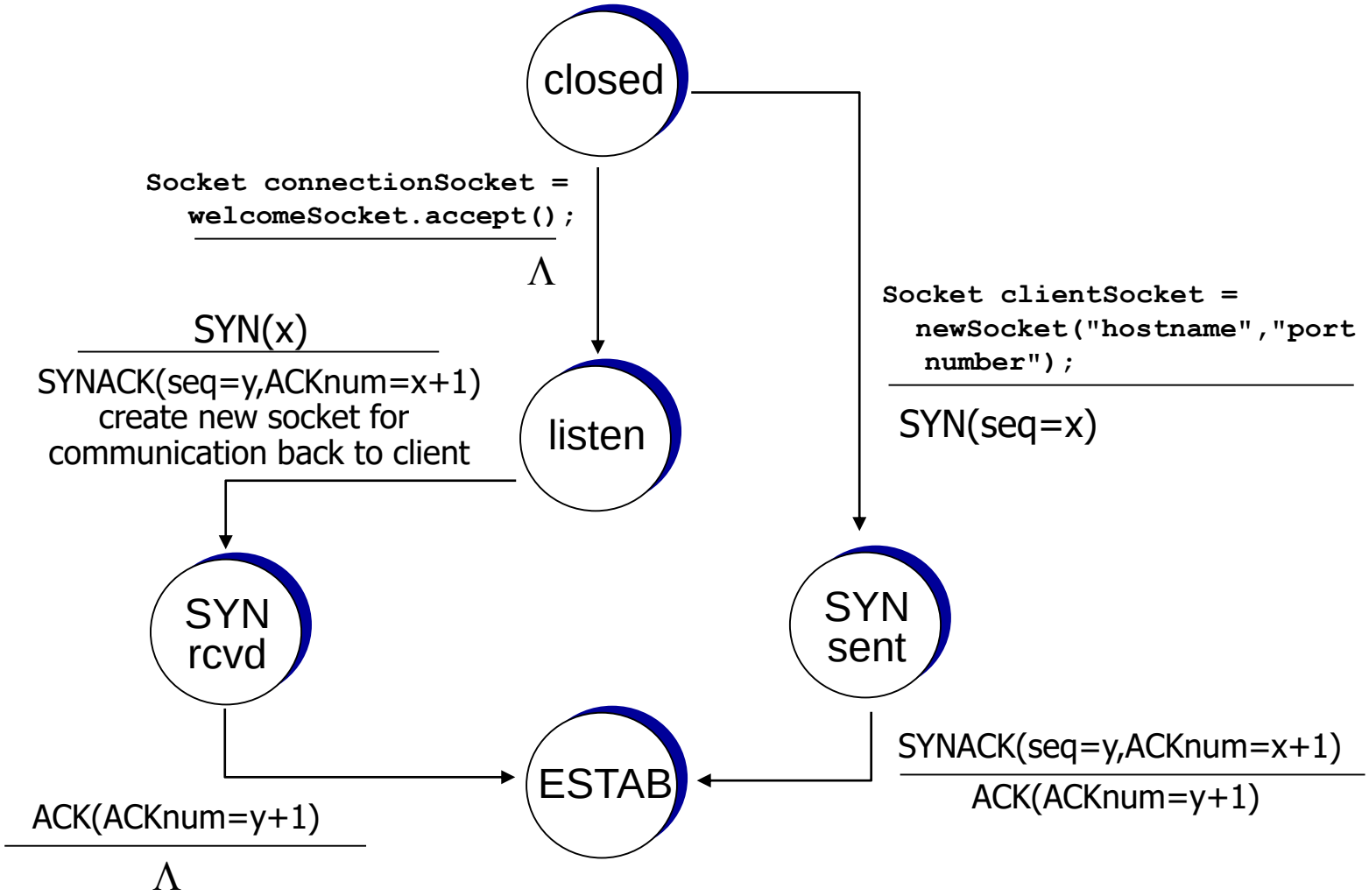
Ε: η διμερής χειραψία δουλεύει πάντα στο δίκτυο;

- μεταβλητές καθυστερήσεις
- επαναμεταδιδόμενα μηνύματα (π.χ. req_conn(x)) λόγω απωλειών μηνυμάτων
- αναδιάταξη μηνύματος
- δεν μπορεί να δει την "άλλη" πλευρά

TCP 3-μερής χειραψία



TCP 3-μερής χειραψία FSM



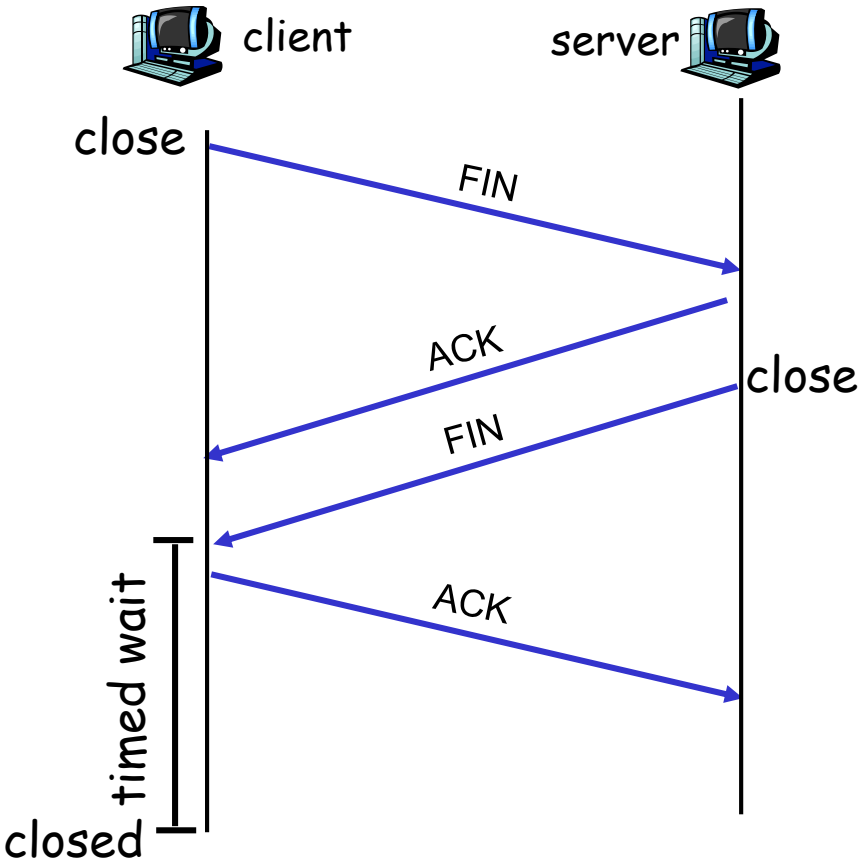
TCP: Κλείσιμο σύνδεσης

Κλείσιμο μιας σύνδεσης:

Ο πελάτης κλείνει τη socket:
`clientSocket.close()`

Βήμα 1: Το τερματικό σύστημα πελάτης στέλνει τμήμα ελέγχου του TCP FIN στον εξυπηρέτη.

Βήμα 2: Ο εξυπηρέτης λαμβάνει FIN, απαντά με ACK. Κλείνει τη σύνδεση, στέλνει FIN.



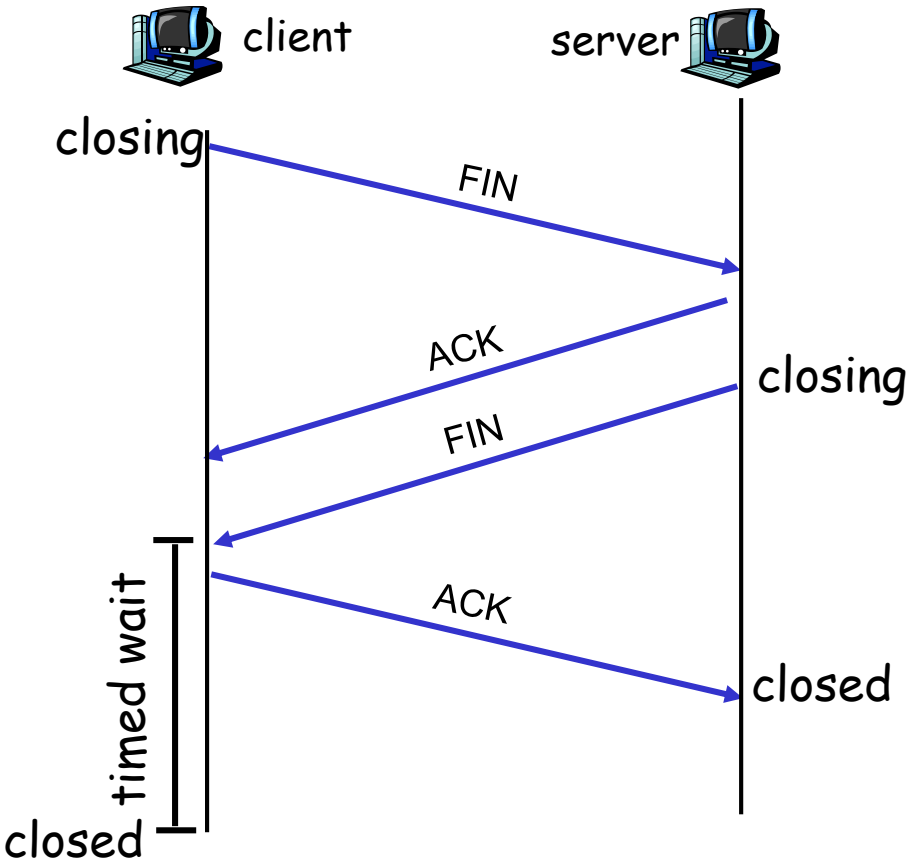
TCP: Κλείσιμο σύνδεσης (συν.)

Βήμα 3: ο πελάτης λαμβάνει FIN, απαντά με ACK.

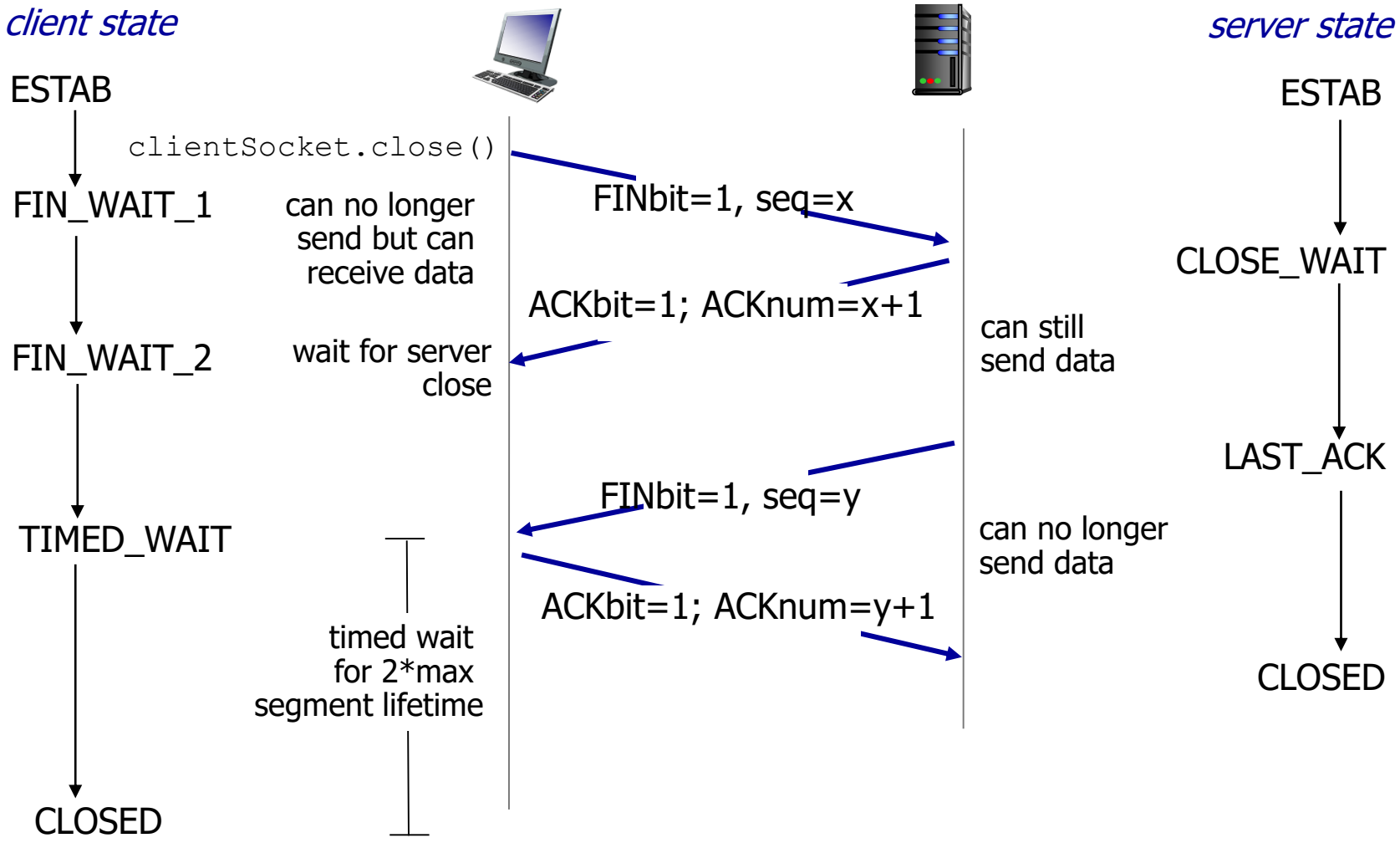
- Εισέρχεται σε πεπερασμένη αναμονή ("timed wait") – θα απαντήσει με ACK σε λαμβανόμενα FINs

Βήμα 4: Ο εξυπηρέτης, λαμβάνει ACK. Κλειστή σύνδεση.

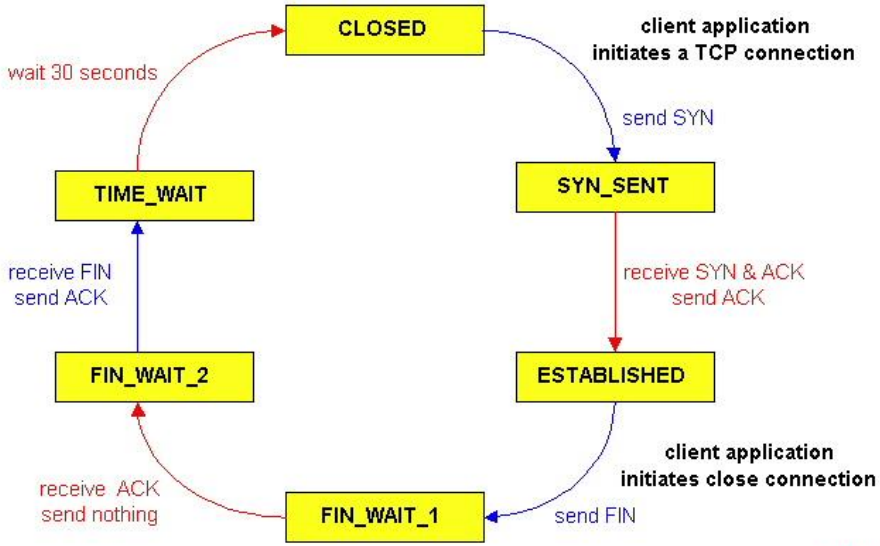
Σημείωση: με μικρή τροποποίηση, μπορεί να διαχειριστεί ταυτόχρονα FINs.



TCP: Κλείσιμο σύνδεσης (ανάλυση)

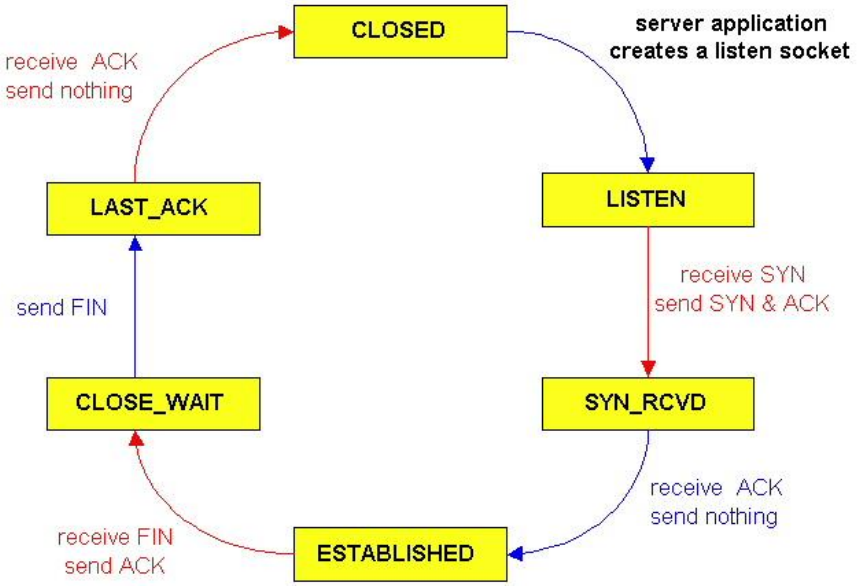


Ο κύκλος ζωής μιας TCP σύνδεσης



TCP client lifecycle

TCP server lifecycle



Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

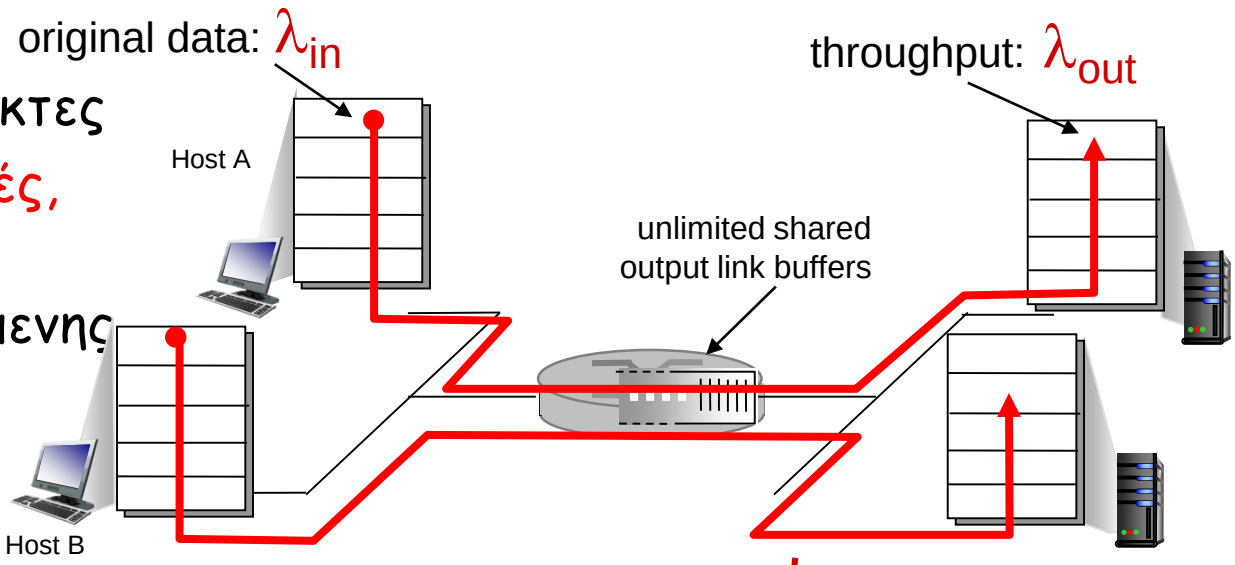
Αρχές του Ελέγχου Συμφόρησης

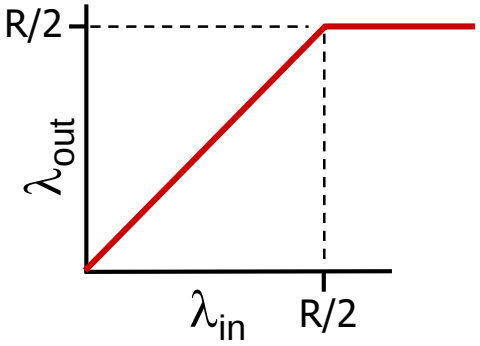
Συμφόρηση:

- "πολλές πηγές που στέλνουν πολλά δεδομένα πολύ γρήγορα για να τα χειριστεί το δίκτυο"
- διαφορετικό από τον έλεγχο ροής!
- συμπτώματα:
  - χαμένα πακέτα (υπερχείλιση ενταμιευτών στους δρομολογητές)
  - μεγάλες καθυστερήσεις (αναμονή στους ενταμιευτές των δρομολογητών)
- σημαντικό πρόβλημα!

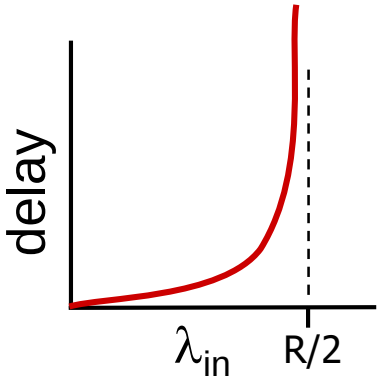
Αίτια/κόστη συμφόρησης: σενάριο 1

- δύο αποστολείς, δύο δέκτες
- απεριόριστοι ενταμιευτές, ένας δρομολογητής
- χωρητικότητα εξερχόμενης ζεύξης: R
- χωρίς αναμεταδόσεις





- μέγιστη ρυθμαπόδοση ανά σύνδεση: R/2

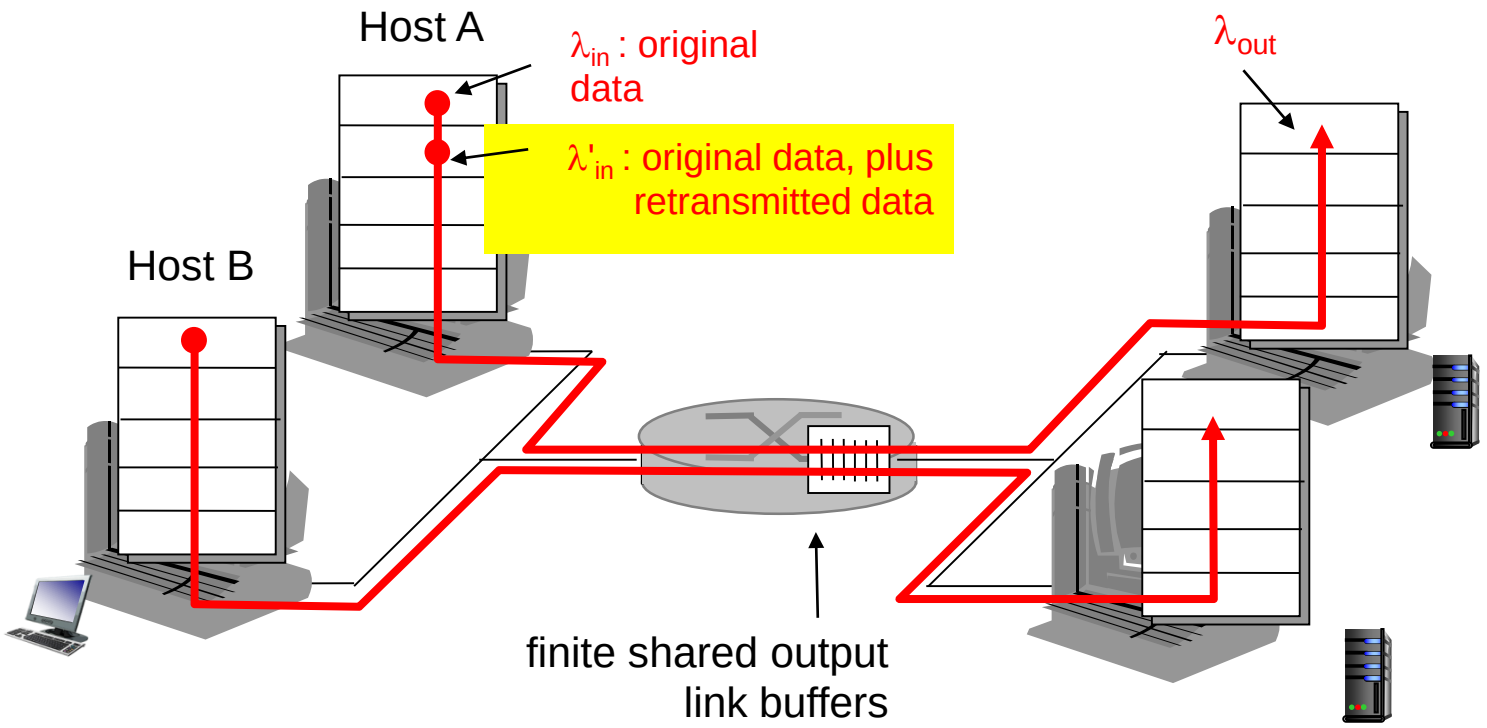


- μεγάλες καθυστερήσεις δεδομένου ότι ο ρυθμός άφιξης λ_{in} προσεγγίζει την χωρητικότητα

Αίτια/κόστη συμφόρησης: σενάριο 2

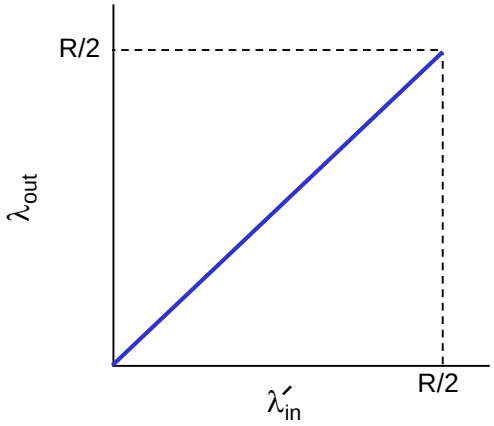
πεπερασμένοι ενταμιευτές, ένας δρομολογητής

- ο αποστολέας αναμεταδίδει τα χαμένα πακέτα
  - είσοδος επιπέδου εφαρμογής = έξοδος επιπέδου εφαρμογής: **λin = λout (goodput)** – εφόσον ότι στέλνεται τελικά λαμβάνεται ύστερα από κάποιες αναμεταδόσεις.
  - είσοδος επιπέδου μεταφοράς περιλαμβάνει αναμεταδόσεις: **λin' ≥ λin**

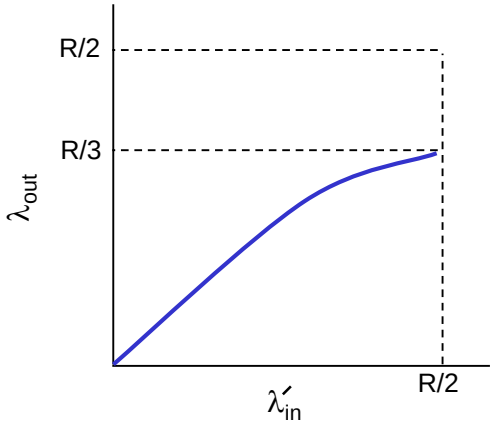


Αίτια/κόστη συμφόρησης: σενάριο 2 (συν.)

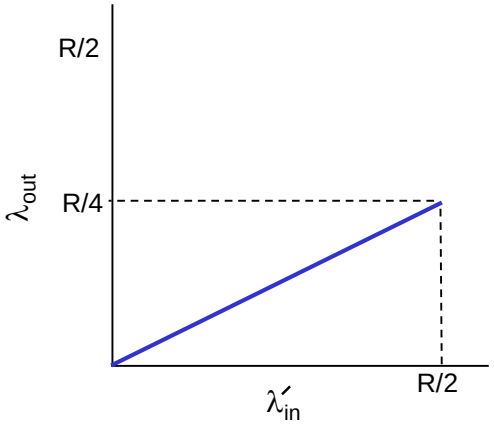
- λόγω αναμεταδόσεων: $\lambda'_{in} > \lambda_{out}$
- αναμετάδοση καθυστερημένων (όχι χαμένων) πακέτων κάνει το λ'_{in} μεγαλύτερο (από ότι αν αναμεταδίδονται μόνο τα χαμένα) για το ίδιο λ_{out}



a. Καμία απώλεια/αναμετάδοση



b. Μέχρι R/2-R/3 αναμεταδόσεις



c. Δύο φορές κάθε πακέτο

"κόστη" της συμφόρησης:

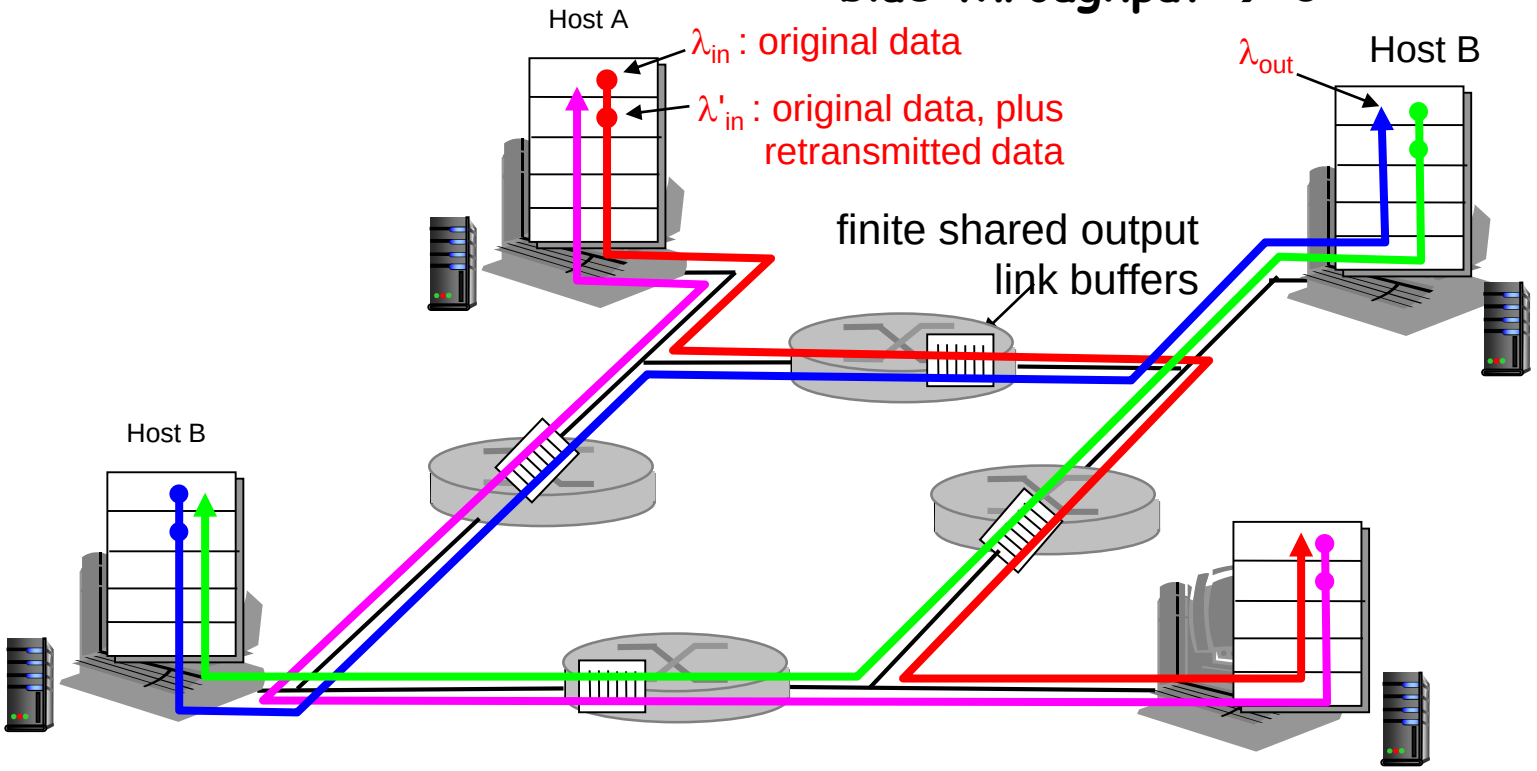
- περισσότερη δουλειά (αναμεταδόσεις) για δοσμένο "goodput"
- αχρείαστες αναμεταδόσεις: η ζεύξη μεταφέρει πολλαπλά αντίγραφα του πακέτου
  - μείωση "goodput"

Αίτια/κόστη της συμφόρησης: σενάριο 3

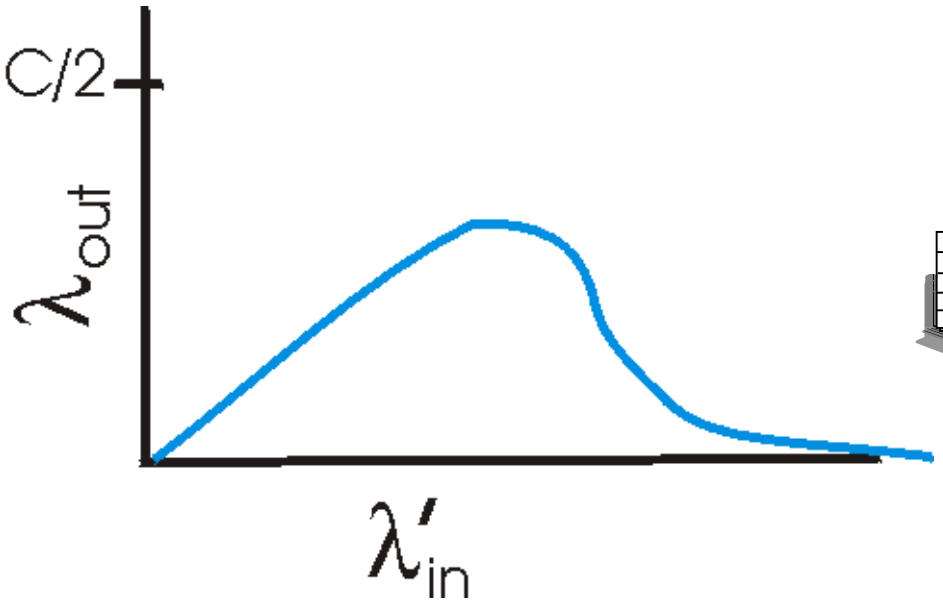
- Τέσσερεις αποστολείς
- Διαδρομές πολλαπλών τμημάτων
- Λήξη χρόνου/αναμετάδοση

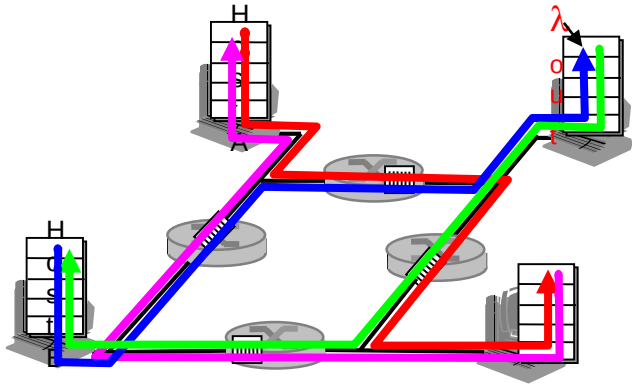
Ε: τι συμβαίνει καθώς το λ_{in} και το λ'_{in} αυξάνουν ?

Α: καθώς το κόκκινο λ'_{in} αυξάνει, όλα τα μπλε πακέτα που φθάνουν στην πάνω ουρά απορρίπτονται,
blue throughput → 0



Αίτια/κόστη της συμφόρησης: σενάριο 3





Ένα άλλο "κόστος" της συμφόρησης:

- όταν ένα πακέτο χάνεται, η "upstream" (αντιρευματική) χωρητικότητα που χρησιμοποιήθηκε για τη μετάδοσή του έχει σπαταληθεί!

Προσεγγίσεις στον έλεγχο συμφόρησης

Από άκρο σε άκρο έλεγχος συμφόρησης:

- καμιά άμεση ανάδραση από το δίκτυο
- η συμφόρηση συνάγεται από τις απώλειες, καθυστερήσεις που παρατηρούν τα τερματικά συστήματα
- προσέγγιση που ακολουθεί το TCP

Έλεγχος συμφόρησης επιβοηθούμενος από το δίκτυο:

- οι δρομολογητές παρέχουν ανάδραση στα τερματικά συστήματα
  - ένα bit που υποδεικνύει συμφόρηση (SNA, DECbit, TCP/IP ECN, ATM)
  - σαφής ρυθμός με τον οποίο ο αποστολέας θα πρέπει να στέλνει

Μελέτη περίπτωσης: Έλεγχος συμφόρησης ATM ABR

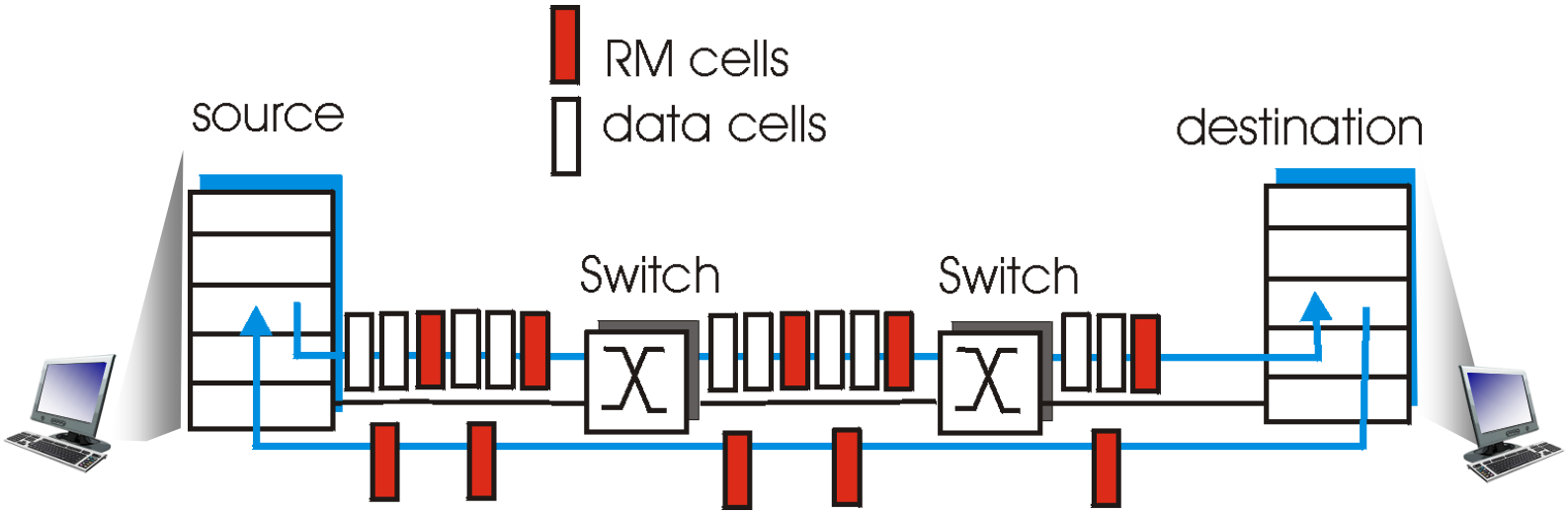
ABR: available bit rate (διαθέσιμος ρυθμός bit):

- "ελαστική υπηρεσία"
- Αν η διαδρομή του αποστολέα είναι υποφορτωμένη ("underloaded"):
  - Ο αποστολέας θα πρέπει να χρησιμοποιήσει όσο από το διαθέσιμο εύρος ζώνης επιθυμεί/μπορεί
- Αν η διαδρομή του αποστολέα είναι σε συμφόρηση:
  - Ο αποστολέας ρυθμίζει το ρυθμό μετάδοσης στον ελάχιστο εγγυημένο ρυθμό

RM (resource management) cells (κελιά διαχείρισης πόρων):

- Στέλνονται από τον αποστολέα, διασπαρμένα σε κελιά δεδομένων
- Κάποια bits στο RM cell τίθενται από τους μεταγωγούς (επιβοηθούμενο από το δίκτυο)
  - NI bit: όχι αύξηση στο ρυθμό (μέτρια συμφόρηση)
  - CI bit: ένδειξη συμφόρησης
- Τα RM cells επιστρέφονται στον αποστολέα από το δέκτη, με τα bits άθικτα

Μελέτη περίπτωσης: Έλεγχος συμφόρησης ATM ABR



- πεδίο ER (explicit rate): δύο bytes στο κελί RM
  - μεταγωγός με συμφόρηση μπορεί να θέσει χαμηλότερη τιμή του ER στο κελί
  - έτσι ο ρυθμός αποστολής του αποστολέα ισούται με το μέγιστο υποστηριζόμενο ρυθμό στη διαδρομή
- EFCI bit στα κελιά δεδομένων: τίθεται 1 σε μεταγωγό με συμφόρηση
  - αν το κελί δεδομένων που προηγείται του κελιού RM έχει το EFCI ίσο με 1, ο προορισμός θέτει 1 το CI bit στο κελί RM που στέλνει πίσω

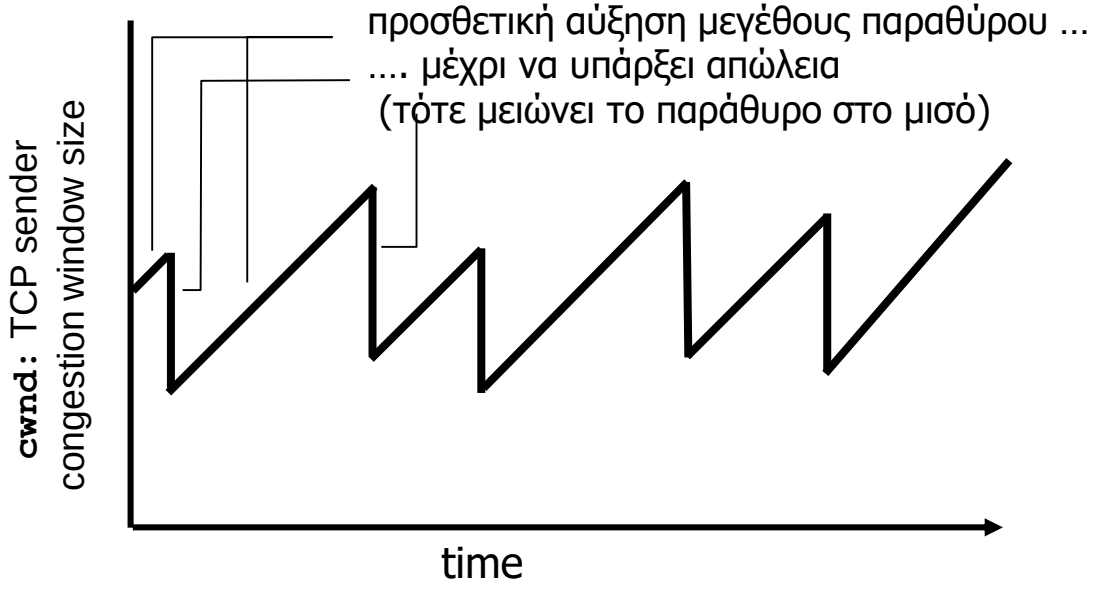
Κεφάλαιο 3: περίγραμμα

- 3.1 Υπηρεσίες επιπέδου μεταφοράς
- 3.2 Πολύπλεξη και αποπολύπλεξη
- 3.3 Ασυνδεσμική μεταφορά: UDP
- 3.4 Αρχές της αξιόπιστης μεταφοράς δεδομένων
- 3.5 Συνδεσμική μεταφορά: TCP
  - Δομή τμήματος
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Διαχείριση σύνδεσης
- 3.6 Αρχές ελέγχου συμφόρησης
- 3.7 Έλεγχος συμφόρησης του TCP

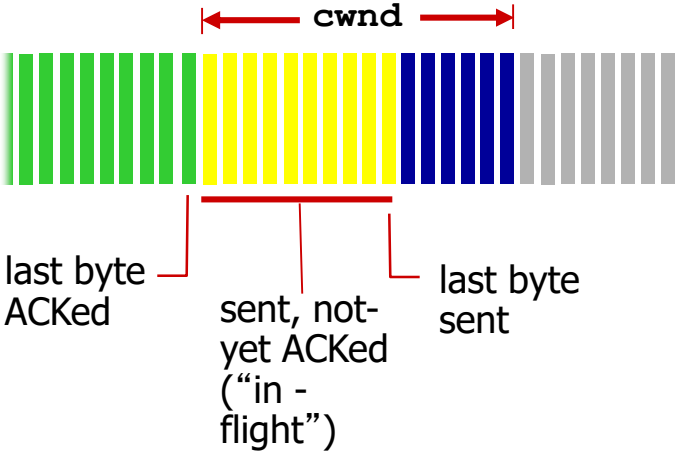
Έλεγχος συμφόρησης TCP: προσθετική αύξηση, πολλαπλασιαστική μείωση (AIMD)

- *Προσέγγιση:* αύξηση ρυθμού μετάδοσης αποστολέα (μέγεθος παραθύρου), ανίχνευση του χρησιμοποιήσιμου εύρους ζώνης, μέχρι να εμφανιστεί απώλεια
  - *προσθετική αύξηση (additive increase):* αύξηση του **cwnd (CongWin)** κατά 1 MSS κάθε RTT μέχρι να εμφανιστεί απώλεια
  - *πολλαπλασιαστική μείωση (multiplicative decrease)* : μείωση του **CongWin** στο μισό μετά από απώλεια

AIMD
«πριονωτή»
συμπεριφορά



Έλεγχος συμφόρησης TCP: λεπτομέρειες



- Ο αποστολέας περιορίζει τη μετάδοση:

  LastByteSent-LastByteAcked ≤ CongWin

- Το CongWin είναι δυναμικό, συνάρτηση της παρατηρούμενης συμφόρησης του δικτύου

$$\text{ρυθμός} = \frac{\text{CongWin}}{\text{RTT}} \text{ bytes/sec}$$

TCP ρυθμός αποστολής:

- *περίπου*: στέλνει **CongWin** bytes, περιμένει **RTT** για ACKs, τότε στέλνει περισσότερα bytes

Πώς παρατηρεί ο αποστολέας τη συμφόρηση;

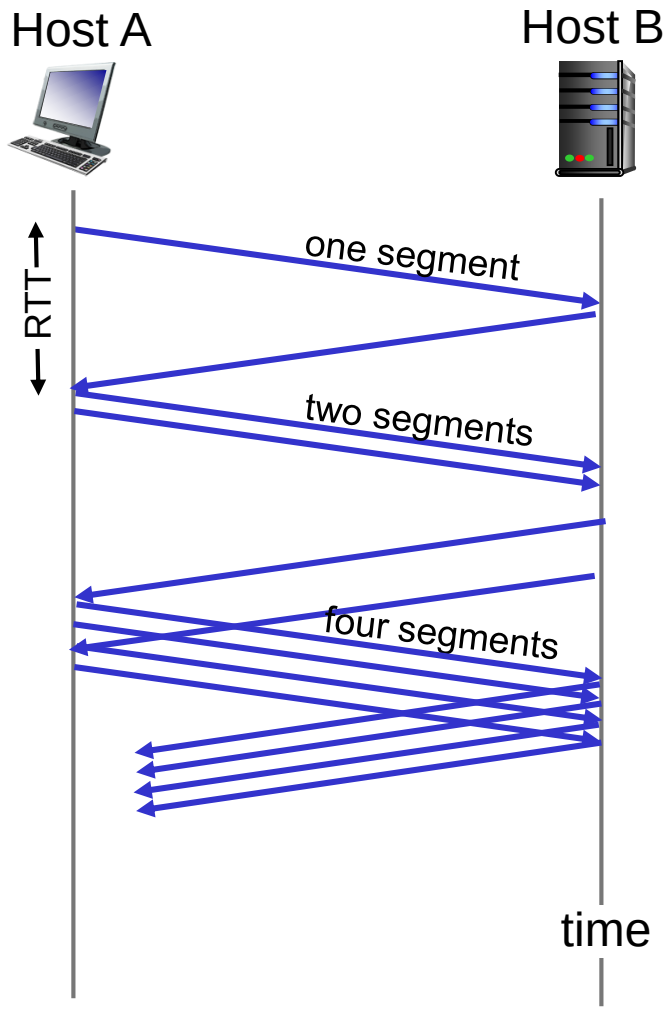
- Γεγονός απώλειας = Λήξη χρόνου (timeout) ή 3 διπλότυπα ACKs
- Ο αποστολέας TCP μειώνει το ρυθμό (**CongWin**) μετά το γεγονός της απώλειας

Τρεις μηχανισμοί:

- AIMD
- Αργή εκκίνηση
- Συντηρητισμός μετά από γεγονότα timeout

Αργή εκκίνηση του TCP

- Όταν ξεκινά η σύνδεση, ο ρυθμός αυξάνεται εκθετικά μέχρι την πρώτη απώλεια:
  - αρχικά CongWin = 1 MSS
  - το CongWin διπλασιάζεται σε κάθε RTT
  - γίνεται αυξάνοντας το CongWin με κάθε ACK που λαμβάνεται
- Σύνοψη: αρχικός ρυθμός αργός, αλλά ανεβαίνει εκθετικά γρήγορα



Αργή εκκίνηση του TCP(συν.)

- Όταν ξεκινά η σύνδεση, **CongWin** = 1 MSS
  - Π.Χ.: MSS = 500 bytes & RTT = 200 msec
  - Αρχικός ρυθμός = 20 kbps
    (500bytes*8bits/byte*1/0.2sec)

- Το διαθέσιμο εύρος ζώνης ενδέχεται να είναι >> MSS/RTT
  - Είναι επιθυμητή η γρήγορη επίτευξη ενός σεβαστού ρυθμού

TCP: ανίχνευση, αντίδραση σε απώλειες

- οι απώλειες υποδεικνύονται από τα timeout:
  - το CongWin ορίζεται σε 1 MSS
  - στη συνέχεια το παράθυρο αυξάνεται εκθετικά (όπως στην αργή εκκίνηση) μέχρι ένα κατώφλι (Threshold), μετά αυξάνεται γραμμικά
- απώλειες υποδεικνύονται από 3 διπλότυπα ACK: TCP RENO
  - διπλότυπα ACKs υποδεικνύουν δίκτυο ικανό να παραδώσει ορισμένα τμήματα
  - το CongWin μειώνεται στο μισό παράθυρο, μετά αυξάνεται γραμμικά
- TCP Tahoe πάντα θέτει το CongWin στο 1 (timeout ή 3 διπλά ACK)

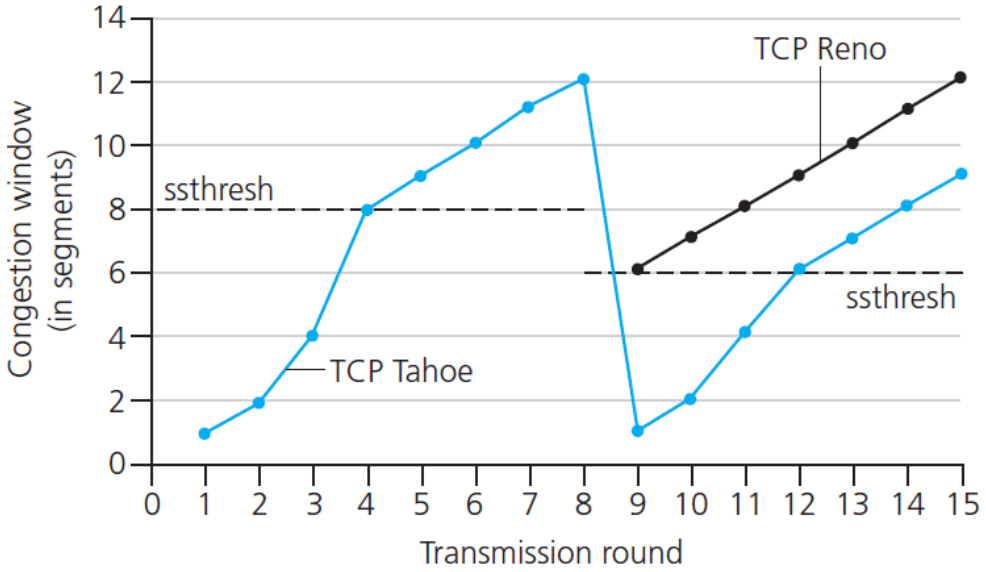
Φιλοσοφία:

- 3 ίδια ACK υποδεικνύουν δίκτυο ικανό να παραδώσει μερικά τμήματα
- timeout πριν από 3 ίδια ACK είναι «πιο ανησυχητικό»

TCP: Μετάβαση από αργή εκκίνηση σε αποφυγή συμφόρησης (Slow Start to Congestion Avoidance)

Ε: Πότε θα πρέπει να γίνει η αλλαγή από εκθετική σε γραμμική αύξηση;

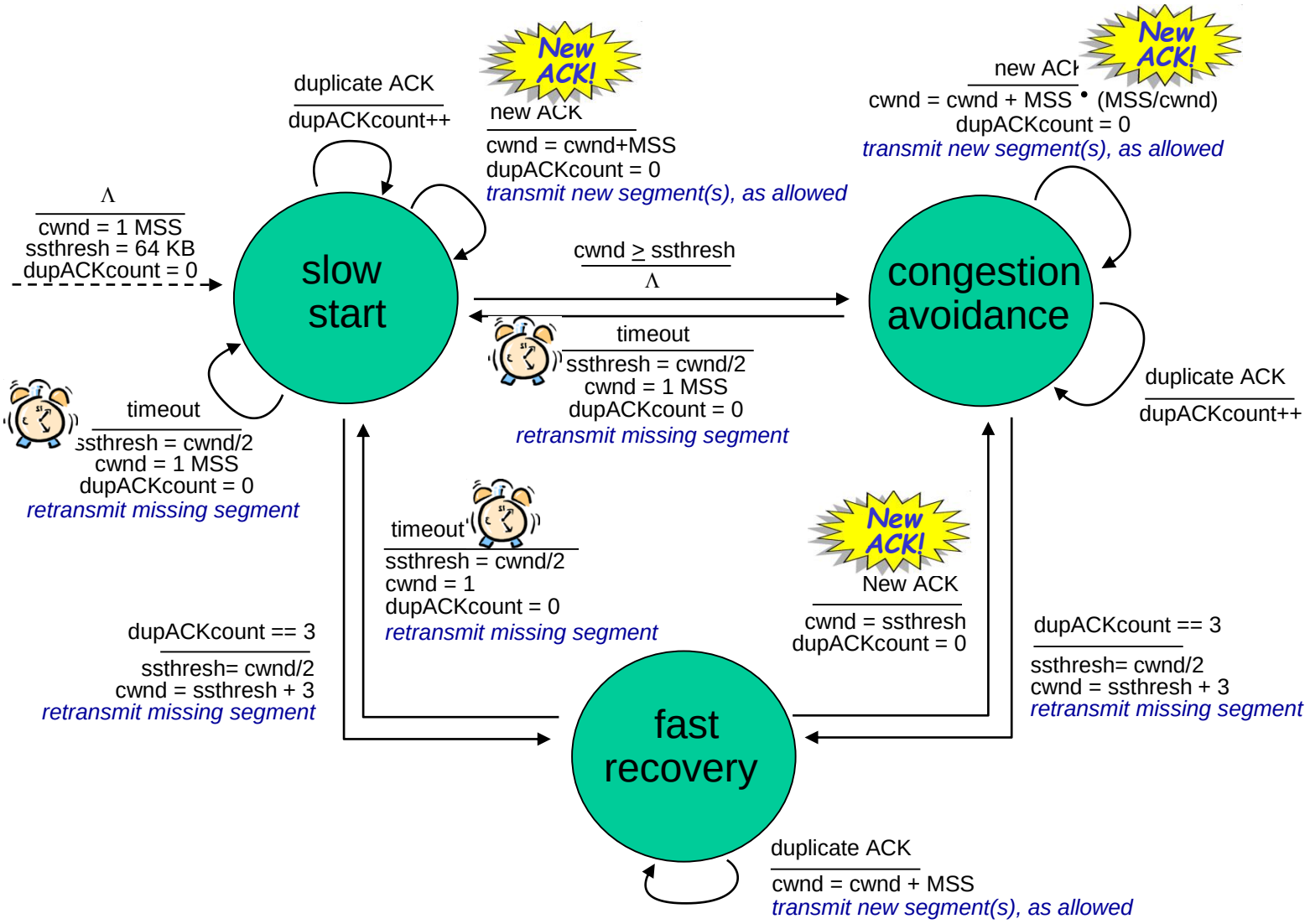
Α: Όταν το **CongWin** γίνει το 1/2 της τιμής του πριν το timeout.



Υλοποίηση:

- Μεταβλητό κατώφλι-Threshold (ssthresh)
- Σε γεγονός απώλειας, το Threshold τίθεται στο 1/2 του CongWin πριν το γεγονός της απώλειας

Σύνοψη: Έλεγχος συμφόρησης TCP



Σύνοψη: Έλεγχος Συμφόρησης του TCP (2)

- Όταν το CongWin είναι κάτω από το Threshold, ο αποστολέας είναι στη φάση αργής εκκίνησης, το παράθυρο αυξάνεται εκθετικά.
- Όταν το CongWin είναι πάνω από το Threshold, ο αποστολέας είναι στη φάση αποφυγής συμφόρησης, το παράθυρο αυξάνεται γραμμικά.
- Όταν εμφανιστεί τριπλό διπλότυπο ACK, το Threshold τίθεται σε CongWin/2 και το CongWin τίθεται σε Threshold.
- Όταν εμφανιστεί timeout, το Threshold τίθεται σε CongWin/2 και το CongWin τίθεται σε 1 MSS.

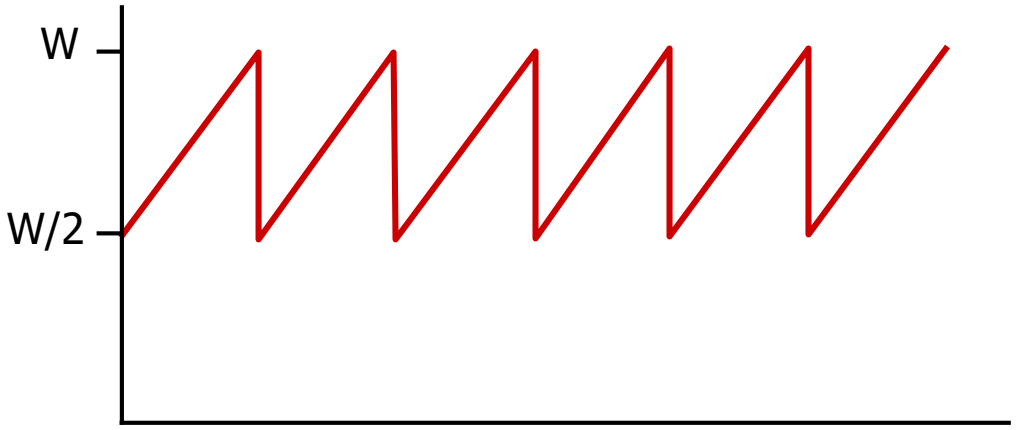
Έλεγχος συμφόρησης του αποστολέα TCP (3)

Κατάσταση	Συμβάν	Ενέργεια αποστολέα TCP	Σχόλια
Αργή Εκκίνηση Slow Start (SS)	Λήψη ACK για δεδομένα που δεν έχουν επιβεβαιωθεί προηγουμένως	CongWin = CongWin + MSS, If (CongWin > Threshold) θέσε κατάσταση σε «Αποφυγή Συμφόρησης»	Έχει ως αποτέλεσμα διπλασιασμό του CongWin σε κάθε RTT
Αποφυγή Συμφόρησης Congestion Avoidance (CA)	Λήψη ACK για δεδομένα που δεν έχουν επιβεβαιωθεί προηγουμένως	CongWin = CongWin+MSS * (MSS/CongWin)	Προσθετική αύξηση που έχει ως αποτέλεσμα αύξηση του CongWin κατά 1 MSS σε κάθε RTT
SS ή CA	Ανίχνευση συμβάντος απώλειας από τρία διπλότυπα ACK	Threshold = CongWin/2, CongWin = Threshold, θέσε κατάσταση σε «Αποφυγή Συμφόρησης»	Ταχεία επαναφορά, υλοποιώντας πολλαπλασιαστική μείωση. Το CongWin δεν θα πέσει κάτω από1 MSS.
SS ή CA	Λήξη χρόνου (Timeout)	Threshold = CongWin/2, CongWin = 1 MSS, Θέσε κατάσταση σε «Αργή Εκκίνηση»	Είσοδος σε «Αργή Εκκίνηση»
SS ή CA	Διπλότυπο ACK	Αύξηση του μετρητή διπλότυπων ACK για το τμήμα η λήψη του οποίου επιβεβαιώθηκε	Τα CongWin και Threshold δεν αλλάζουν

Ρυθμαπόδοση TCP

- Ποιά είναι η ρυθμαπόδοση του TCP ως συνάρτηση του μεγέθους παραθύρου και του RTT;
  - Αγνοώντας την αργή εκκίνηση, υποθέτοντας ότι υπάρχουν πάντα δεδομένα για αποστολή
- Έστω W το μέγεθος παραθύρου (σε bytes) όταν εμφανίζεται απώλεια.
  - Μέσο μέγεθος παραθύρου (# εν πτήσει bytes) είναι $\frac{3}{4}$ W
  - Μέση ρυθμαπόδοση: $\frac{3}{4}$ W ανά RTT

$$\text{Μέση ρυθμαπόδοση TCP} = \frac{3}{4}\,\frac{W}{RTT} \quad \text{bytes/sec}$$



Μέλλον του TCP: TCP πάνω από «μεγάλου μήκους, χοντρές σωληνώσεις» ("long, fat pipes")

Παράδειγμα: τμήματα 1500 bytes, **100ms RTT**, επιθυμητή ρυθμαπόδοση **10 Gbps**

- Απαιτούμενο μέγεθος παραθύρου W = 83,333 «εν πτήσει» τμήματα (segments)
- Ρυθμαπόδοση ως συνάρτηση της πιθανότητας απώλειας τμημάτων, **L:**

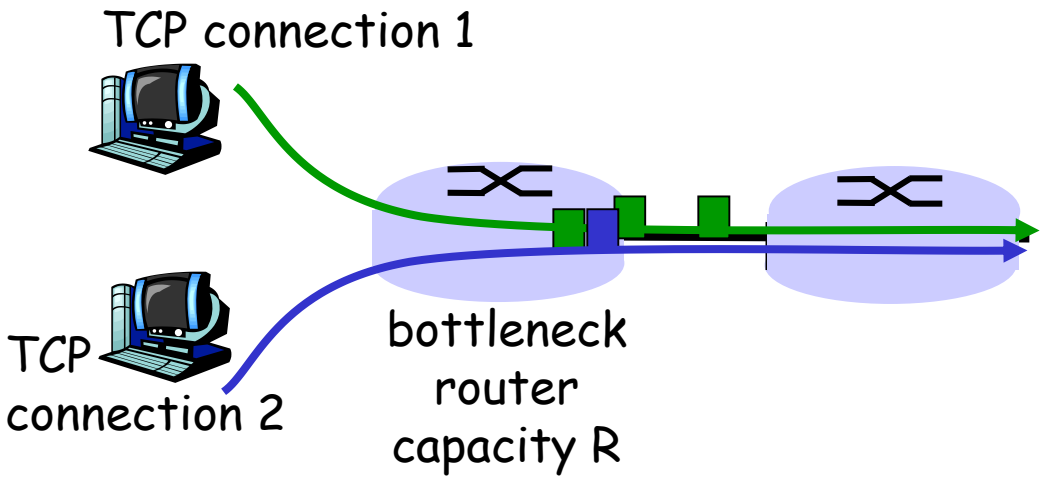
$$\text{Ρυθμαπόδοση TCP} = \frac{1.22 \cdot MSS}{RTT\sqrt{L}}$$

➜ για να επιτευχθεί ρυθμαπόδοση 10 Gbps απαιτείται ρυθμός απωλειών $L = 2 \cdot 10^{-10}$ *(πολύ μικρός ρυθμός απωλειών!)*

Νέες εκδόσεις του TCP για υψηλές ταχύτητες

Δικαιοσύνη του TCP

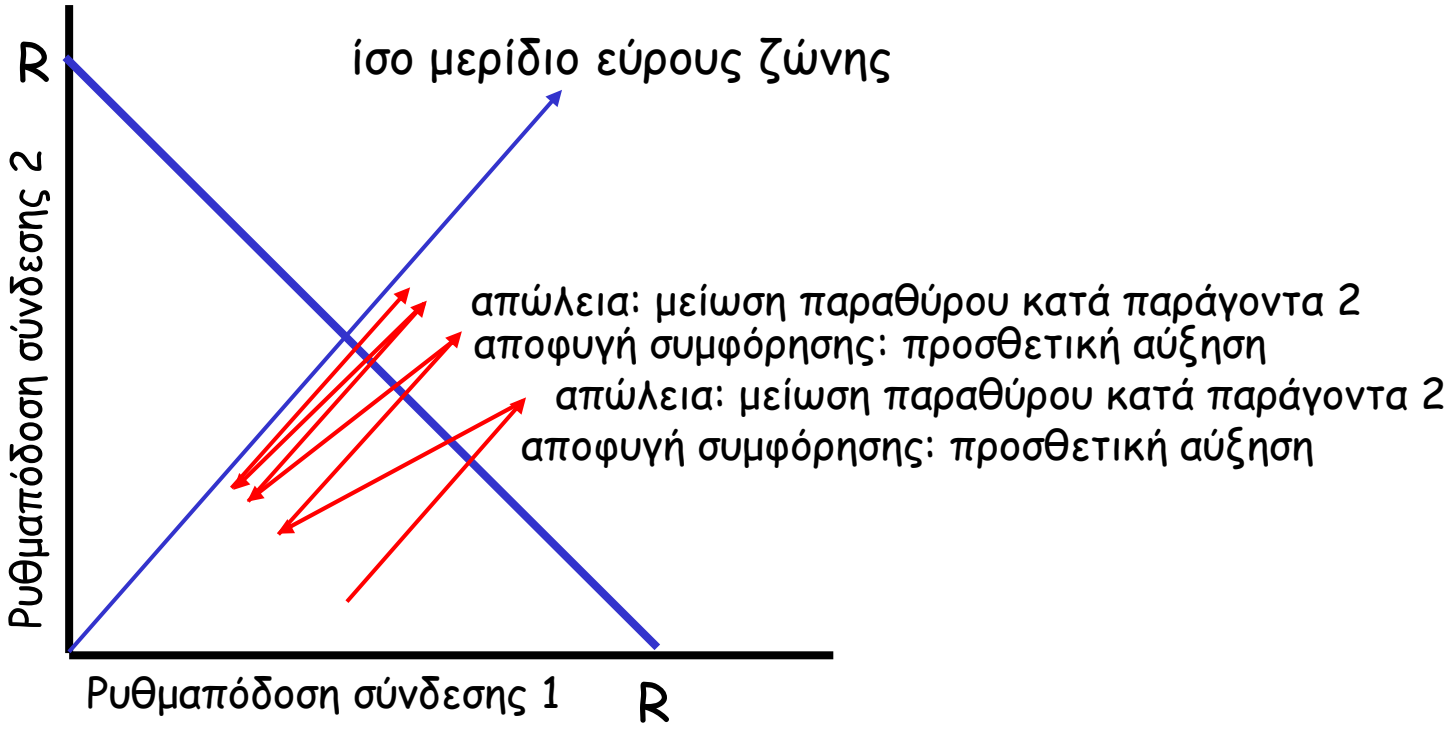
Στόχος : αν K συνδέσεις TCP μοιράζονται την ίδια μποτιλιαρισμένη ζεύξη (bottleneck link) εύρους ζώνης R, καθεμία θα έπρεπε να έχει μέσο ρυθμό R/K



Είναι το TCP δίκαιο;

Δύο ανταγωνιζόμενες συνδέσεις:

- Η προσθετική αύξηση δίνει κλίση 1, καθώς αυξάνει η ρυθμαπόδοση
- Η πολλαπλασιαστική μείωση μειώνει αναλογικά τη ρυθμαπόδοση



Δικαιοσύνη (συνέχεια)

Δικαιοσύνη και UDP

- Οι εφαρμογές πολυμέσων συχνά δε χρησιμοποιούν TCP
  - Δεν θέλουν έλεγχο του ρυθμού από τον έλεγχο συμφόρησης
- Αντί αυτού, UDP:
  - αποστολή audio/video με σταθερό ρυθμό, ανοχή σε απώλειες πακέτων

Δικαιοσύνη και παράλληλες συνδέσεις TCP

- Μία εφαρμογή μπορεί να ανοίξει πολλαπλές παράλληλες συνδέσεις μεταξύ δύο υπολογιστών
- Οι Web browsers το κάνουν
- Παράδειγμα: ζεύξη ρυθμού R που υποστηρίζει 9 συνδέσεις
  - Νέα εφαρμογή ζητά 1 σύνδεση TCP, παίρνει ρυθμό R/10
  - Νέα εφαρμογή ζητά 11 συνδέσεις TCP, παίρνει ρυθμό R/2 !

Κεφάλαιο 3: Σύνοψη

- Υπηρεσίες του επιπέδου μεταφοράς:
  - Πολύπλεξη, αποπολύπλεξη
  - Αξιόπιστη μεταφορά δεδομένων
  - Έλεγχος ροής
  - Έλεγχος συμφόρησης
- Πραγμάτωση και υλοποίηση στο Διαδίκτυο
  - UDP
  - TCP

Στη συνέχεια:

- Αφήνοντας το "άκρο" του δικτύου (επίπεδα εφαρμογής, μεταφοράς) προχωράμε στον "πυρήνα" του δικτύου

Τέλος Ενότητας

Σημείωμα Αναφοράς

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης