ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ

Στρώμα μεταφοράς στο Internet

Περίληψη

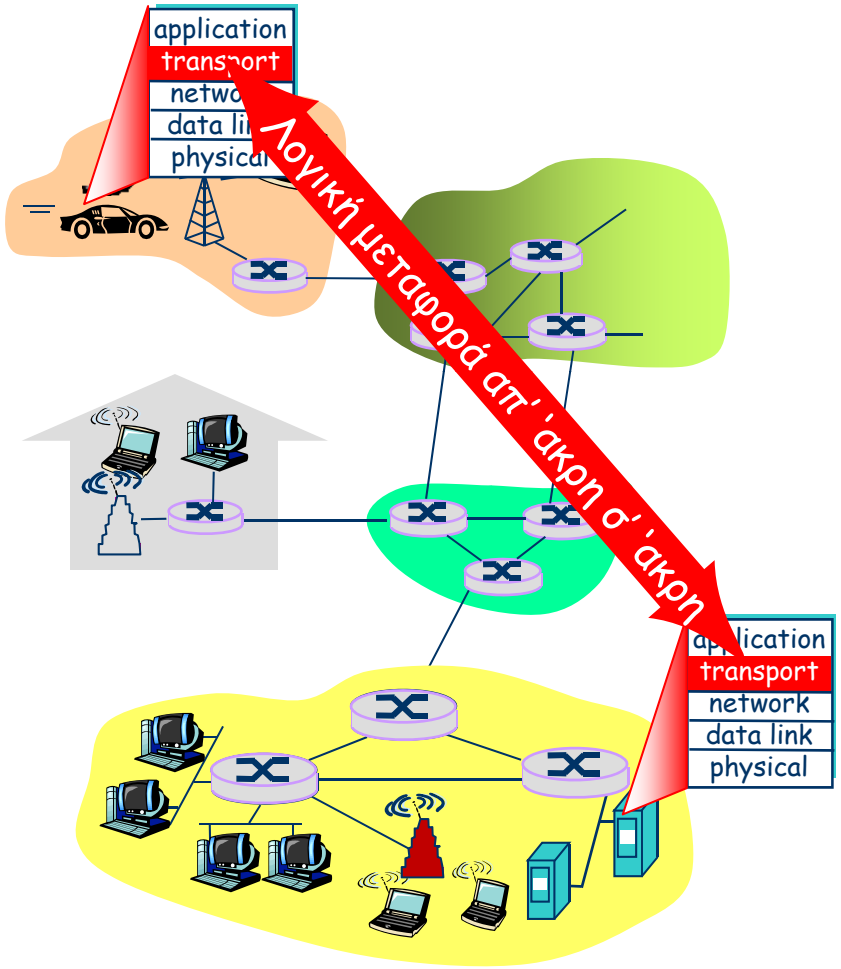
- Αρχές λειτουργίας του στρώματος μεταφοράς και βασικές υπηρεσίες του
  - πολυπλεξία/αποπολυπλεξία
  - αξιόπιστη μεταφορά δεδομένων
  - διαχείριση συνδέσεων
  - έλεγχος ροής
- Πρωτόκολλα στρώματος μεταφοράς στο Internet:
  - UDP: μεταφορά χωρίς σύνδεση
  - TCP: μεταφορά με σύνδεση

Περιεχόμενα

- Υπηρεσίες και πρωτόκολλα του στρώματος μεταφοράς
- Πολυπλεξία/αποπολυπλεξία
- Μεταφορά χωρίς σύνδεση: UDP
- Αξιόπιστη μεταφορά δεδομένων
- Μεταφορά με σύνδεση: TCP
  - υπηρεσία συρμού byte
  - δομή τεμαχίου
  - αξιόπιστη μετάδοση στο TCP
  - διαχείριση χρονομετρητών
  - διαχείριση συνδέσεων
  - μεταφορά δεδομένων
  - έλεγχος ροής

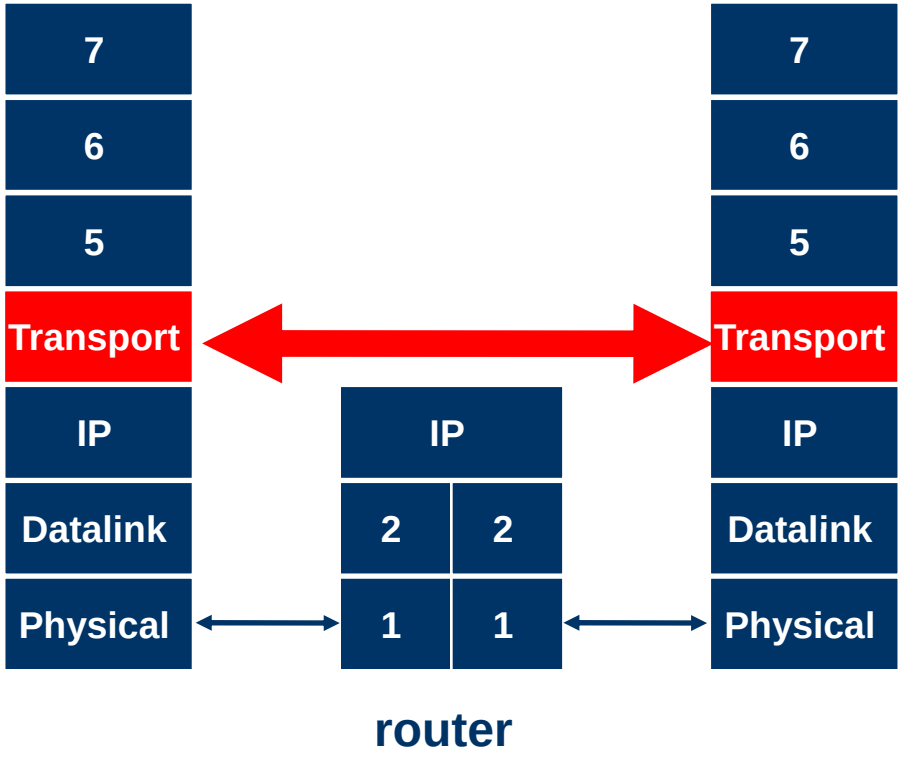
Στρώμα μεταφοράς: Υπηρεσίες και πρωτόκολλα

- παρέχει *λογική επικοινωνία* μεταξύ διαδικασιών εφαρμογής που τρέχουν σε διαφορετικούς host
- τα πρωτόκολλα μεταφοράς τρέχουν στους host
  - εκπομπή: χωρίζει τα μηνύματα εφαρμογής σε τεμάχια και τα διοχετεύει στο στρώμα δικτύου
  - λήψη: επανασυναρμολογεί τα τεμάχια σε μηνύματα και τα διοχετεύει στο στρώμα εφαρμογής
- υπάρχουν περισσότερα από ένα πρωτόκολλα μεταφοράς διαθέσιμα στις εφαρμογές
  - Internet: TCP and UDP



Πρωτόκολλα μεταφοράς

- Είναι τα πρώτα πρωτόκολλα, αρχίζοντας από το φυσικό στρώμα, που λειτουργουν από άκρη σε άκρη.
- Η επικεφαλίδα του πρωτοκόλλου μεταφοράς που δημιουργείται στην πηγή εξετάζεται μόνο από τον προορισμό
- Οι δρομολογητές θεωρούν την επικεφαλίδα του πρωτοκόλλου μεταφοράς ως μέρος του ωφέλιμου φορτίου του δεδομενογράμματος



Στρώμα μεταφοράς και στρώμα δικτύου

- *στρώμα δικτύου:* λογική επικοινωνία μεταξύ host
- *στρώμα μεταφοράς:* λογική επικοινωνία μεταξύ διαδικασιών εφαρμογής
  - βασίζεται στις υπηρεσίες δικτύου και τις βελτιώνει

- Αναλογία με αλληλογραφία:
  - *Δύο σπίτια, σε διαφορετικές πόλεις, με αρκετά παιδιά στο κάθε σπίτι που αλληλογραφούν μεταξύ τους. Η Ελένη στο ένα σπίτι και ο Γιώργος στο άλλο μαζεύουν και μοιράζουν την αλληλογραφία.*
  - διαδικασίες εφαρμογής = παιδιά
  - μηνύματα εφαρμογής = επιστολές σε φακέλους
  - hosts = σπίτια
  - πρωτόκολλο μεταφοράς = Ελένη και Γιώργος
  - πρωτόκολλο στρώματος δικτύου = ταχυδρομική υπηρεσία

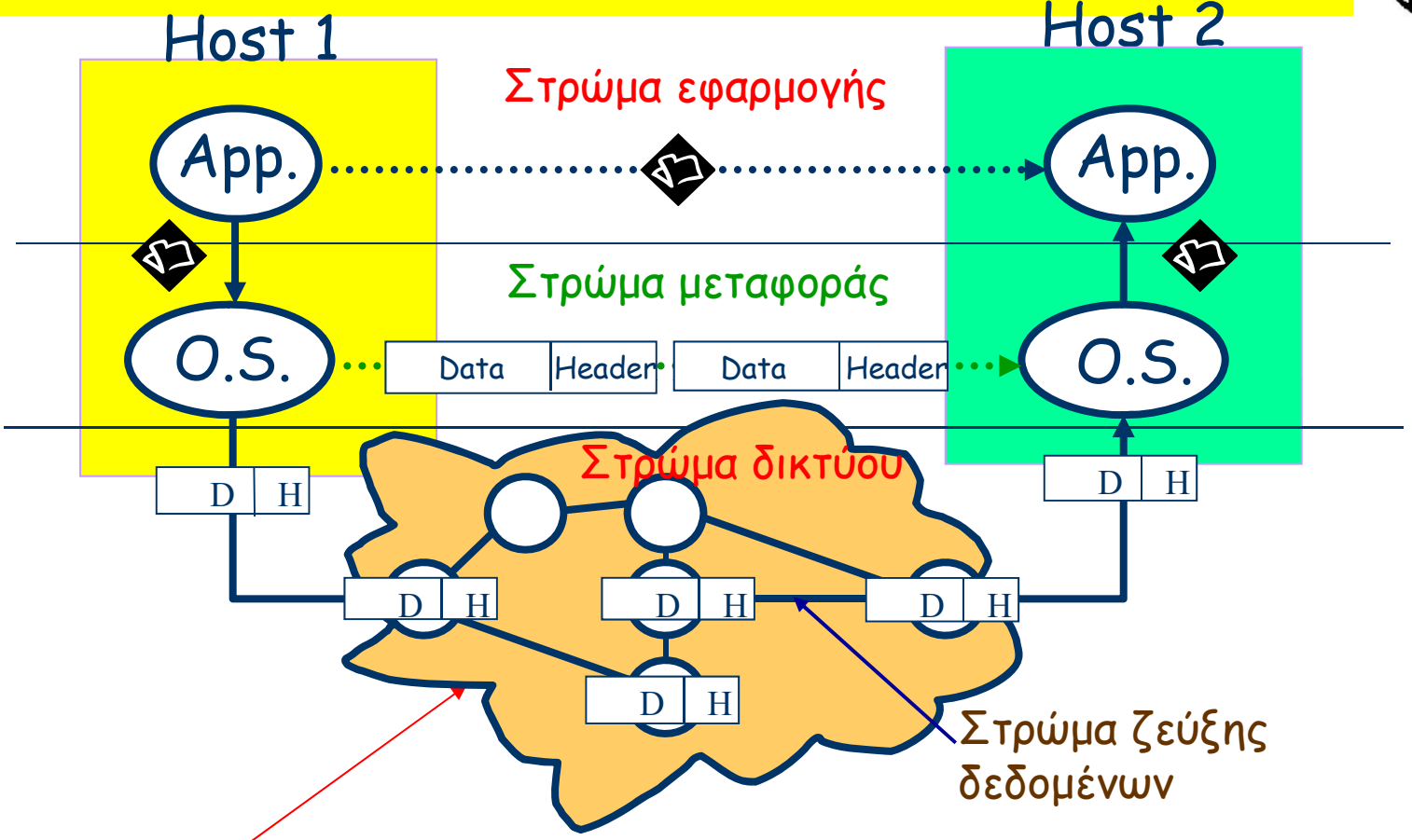
Στρώμα μεταφοράς και στρώμα ζεύξης

- Τα πρωτόκολλα μεταφοράς έχουν κοινά χαρακτηριστικά με εκείνα της ζεύξης δεδομένων, αλλά υπάρχουν και σημαντικές διαφορές:
  - Στο στρώμα μεταφοράς απαιτείται η ρητή διευθυνσιοδότηση των προορισμών
  - Η αρχική εγκατάσταση της σύνδεσης είναι πιο πολύπλοκη
  - Ενδεχόμενη ύπαρξη αποθηκευτικής χωρητικότητας στο δίκτυο
  - Ποσοτική διαφορά (μεγάλος και δυναμικά μεταβαλλόμενος αριθμός συνδέσεων)
- Η προσωρινή αποθήκευση και ο έλεγχος ροής απαιτούν διαφορετική προσέγγιση στο στρώμα μεταφοράς απ' ότι στο στρώμα ζεύξης δεδομένων

Στρώμα μεταφοράς

- Παρέχει *αξιόπιστη* ή *μη αξιόπιστη* μεταφορά δεδομένων μεταξύ διαδικασιών εφαρμογής.
- Υποστηρίζει μηνύματα *αυθαίρετου μήκους*
- Παρέχει τρόπο απόφασης για το ποια τεμάχια πηγαίνουν σε ποιες εφαρμογές (πολυπλεξία/αποπολυπλεξία)
- Ρυθμίζει πότε οι host πρέπει να στέλνουν

Στρώμα μεταφοράς

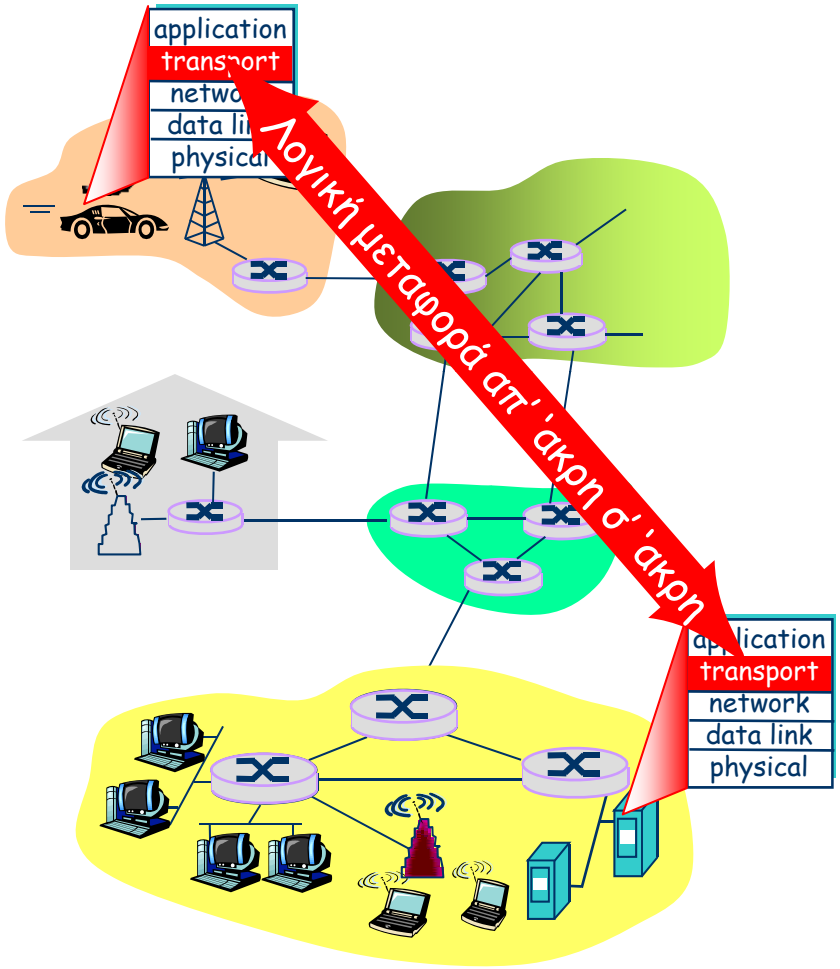


- Υπηρεσία best-effort
- Πακέτα περιορισμένου μήκους
- Πακέτα καθυστερούν, χάνονται,...
- Επικοινωνία μεταξύ host

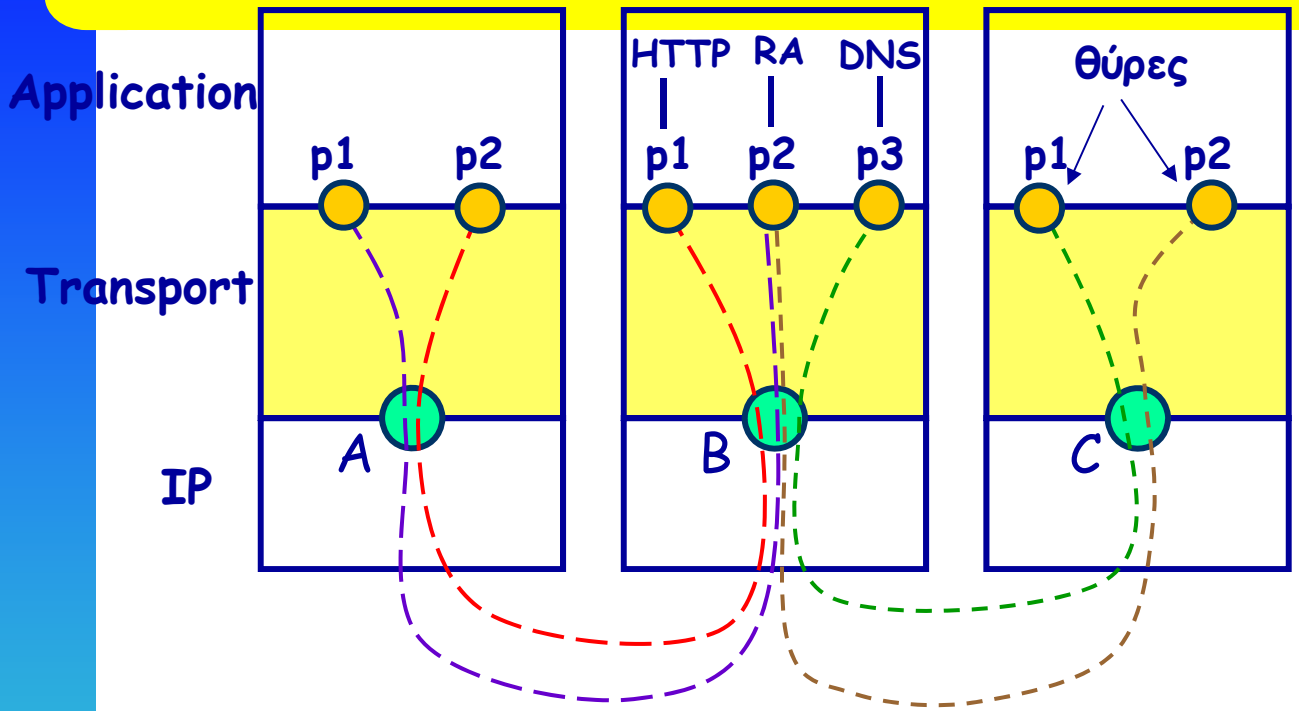
- Ποια εφαρμογή λαμβάνει ποια πακέτα;
- Με ποιο ρυθμό πρέπει να στέλνουν οι host στο δίκτυο;

Στρώμα μεταφοράς στο Internet

- μη αξιόπιστη παράδοση χωρίς διατήρηση της σειράς: UDP
  - μινιμαλιστική επέκταση της υπηρεσίας "best-effort" του IP
- αξιόπιστη παράδοση με διατήρηση της σειράς: TCP
  - εγκατάσταση σύνδεσης
  - έλεγχος ροής
  - έλεγχος συμφόρησης
- υπηρεσίες που δεν προσφέρονται:
  - εξασφάλιση καθυστέρησης
  - εξασφάλιση εύρους ζώνης



Στρώμα μεταφοράς στο Internet



- UDP: αναξιόπιστη μεταφορά
- TCP: αξιόπιστη, με τη σειρά

- Και το TCP και το UDP επεκτείνουν την επικοινωνία IP, μεταξύ host, σε επικοινωνία μεταξύ διαδικασιών εφαρμογής: πολυπλεξία/αποπολυπλεξία του στρώματος μεταφοράς
- Ποια πακέτα λαμβάνει η κάθε εφαρμογή;
  - Λύση: αντιστοίχηση κάθε υποδοχής σε μια θύρα
  - Κάθε τεμάχιο έχει ειδικά πεδία για τους αριθμούς θυρών
  - Οι υποδοχές προσδιορίζονται μονοσήμαντα

Θύρες

- Χώρος διευθύνσεων θυρών των 16-bit για το UDP και το TCP
- Ο client πρέπει να ξέρει τη θύρα του server
- Πασίγνωστες θύρες (0-1023): όλοι συμφωνούν ποιες υπηρεσίες τρέχουν σ' αυτές τις θύρες
  - π.χ., telnet:23, SMTP:25, DNS:53, http:80,
- Εφήμερες θύρες (1024-65535): δίδονται στους client

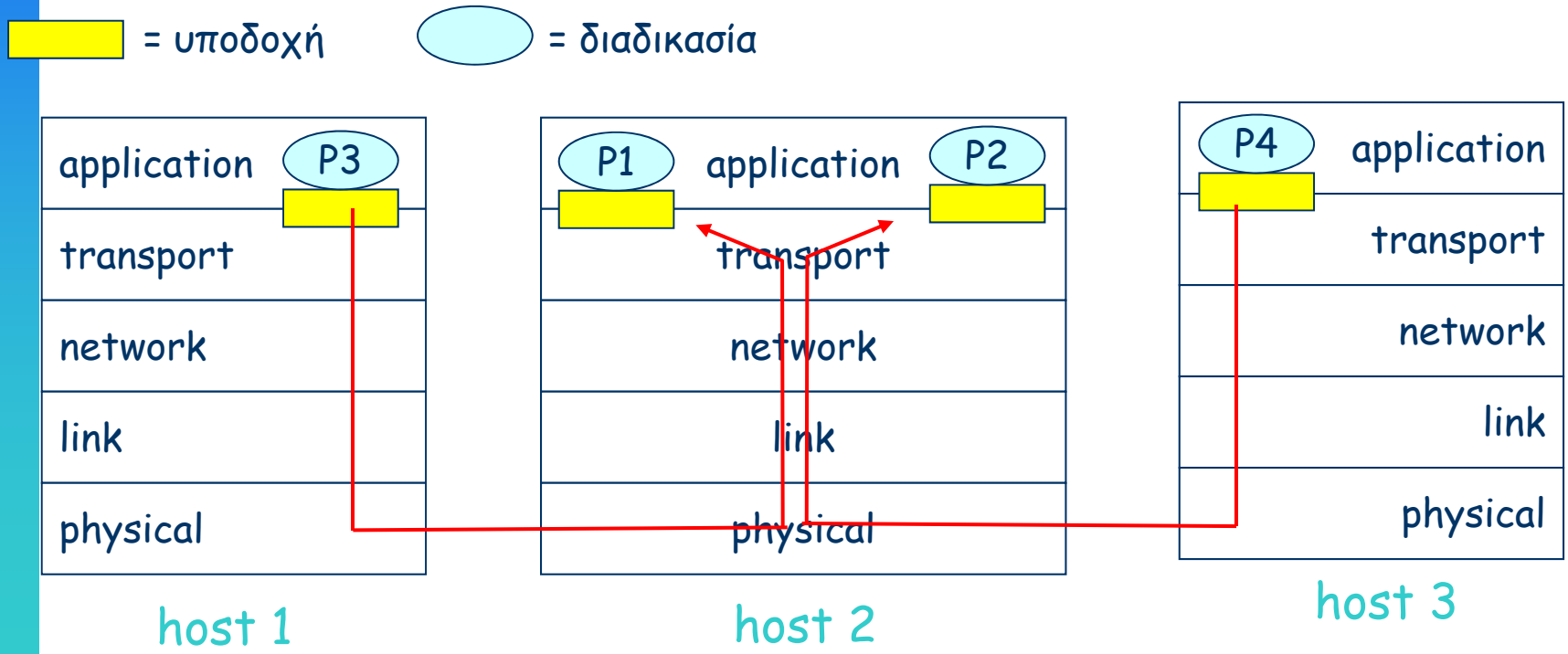
Πολυπλεξία/αποπολυπλεξία

Πολυπλεξία στον host αποστολής:

συλλογή των δεδομένων από πολλές υποδοχές, χαρακτηρισμός τους με επικεφαλίδα (χρησιμοποιείται μετά στην αποπολυπλεξία)

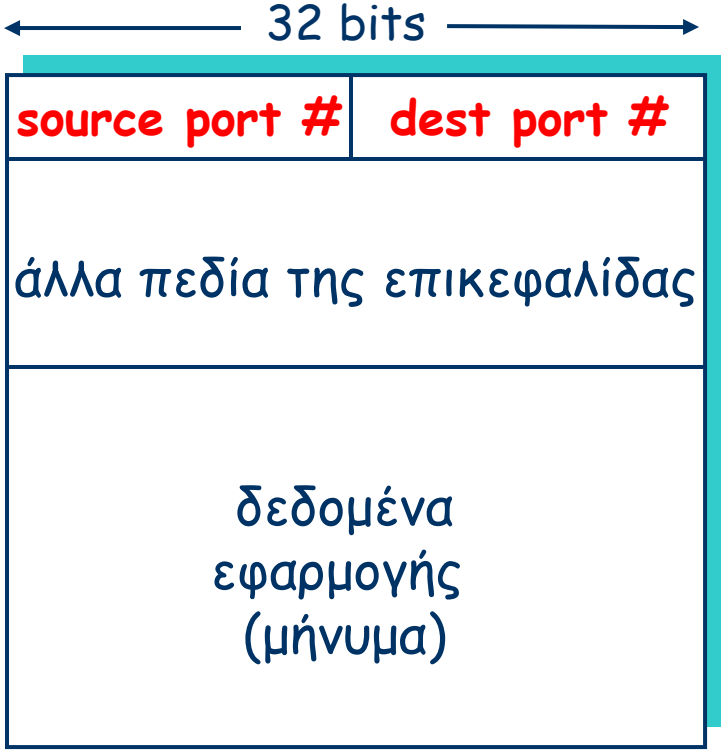
Αποπολυπλεξία στον host λήψης:

παράδοση των λαμβανομένων τεμαχίων στις σωστές υποδοχές



Πώς λειτουργεί η αποπολυπεξία

- ο host λαμβάνει δεδομενογράμματα IP
  - κάθε δεδομενόγραμμα έχει διευθύνσεις source IP και destination IP
  - κάθε δεδομενόγραμμα μεταφέρει ένα τεμάχιο στρώματος μεταφοράς
  - κάθε τεμάχιο έχει αριθμούς source και destination port
- ο host χρησιμοποιεί τις διευθύνσεις IP και τους αριθμούς θυρών για να κατευθύνει το τεμάχιο στην κατάλληλη υποδοχή



μορφή τεμαχίου στρώματος μεταφοράς

Αποπολυπλεξία χωρίς σύνδεση

- Δημιουργία υποδοχών με αριθμούς θυρών:

```
DatagramSocket mySocket1 = new DatagramSocket(9157);
DatagramSocket mySocket2 = new DatagramSocket(9222);
```

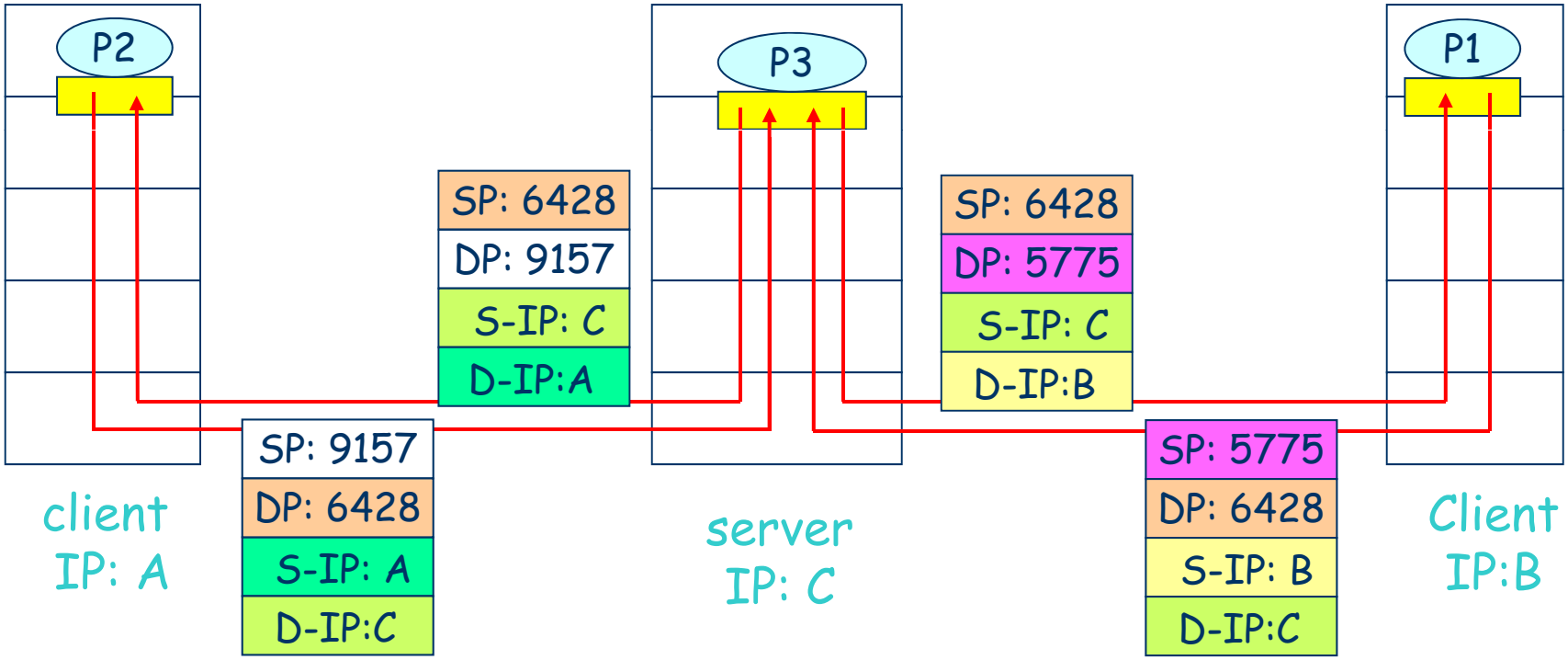
- Μια υποδοχή UDP προσδιορίζεται από το ζεύγος:

  (dest IP address, dest port number)

- Όταν ο host λαμβάνει τεμάχιο UDP:
  - ελέγχει τον αριθμό destination port στο τεμάχιο
  - κατευθύνει το τεμάχιο UDP στην υποδοχή με αυτόν τον αριθμό
- Δεδομενογράμματα IP με διαφορετικές source IP διευθύνσεις και/ή source port αριθμούς, αλλά με ίδια διεύθυνση dest IP και ίδιο αριθμό dest port, οδηγούνται στην ίδια υποδοχή

Αποπολυπλεξία χωρίς σύνδεση

```
DatagramSocket serverSocket = new DatagramSocket(6428);
```

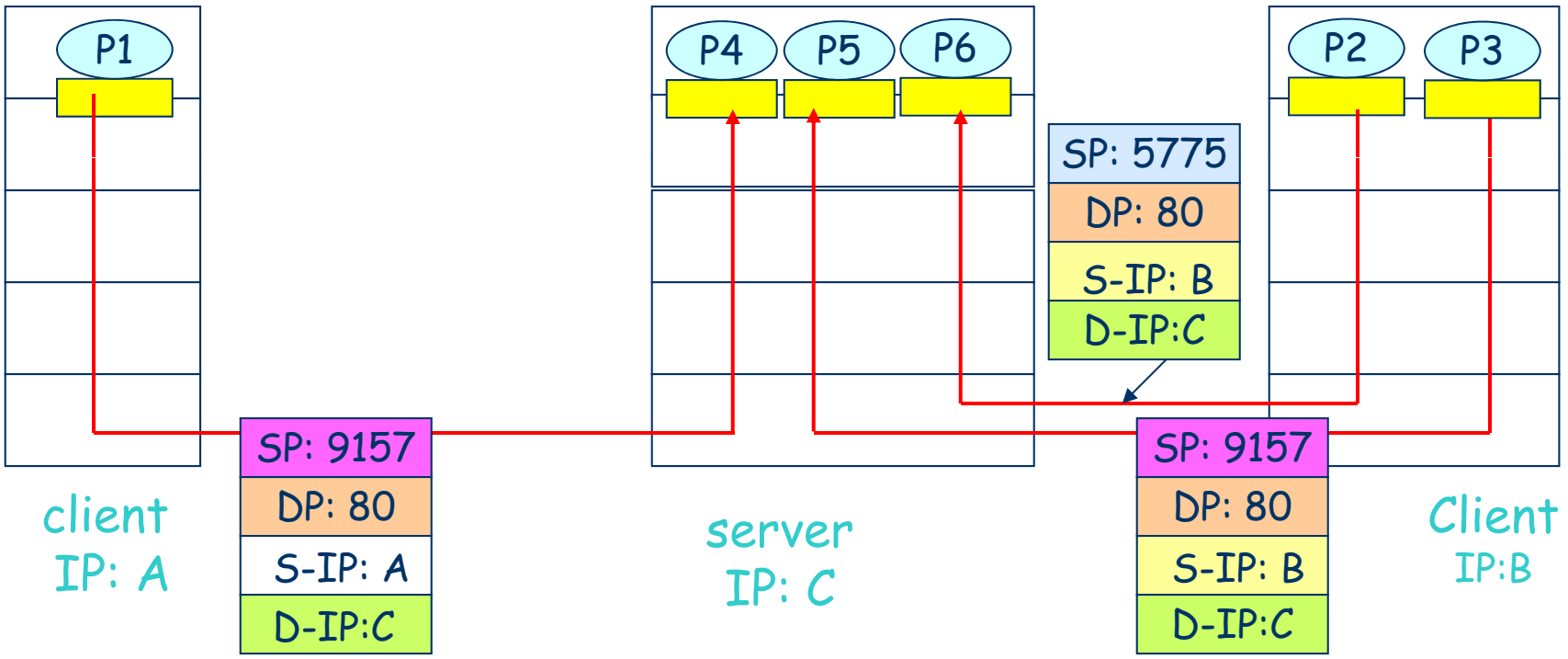


SP είναι η "διεύθυνση επιστροφής"

Αποπολυπλεξία με σύνδεση

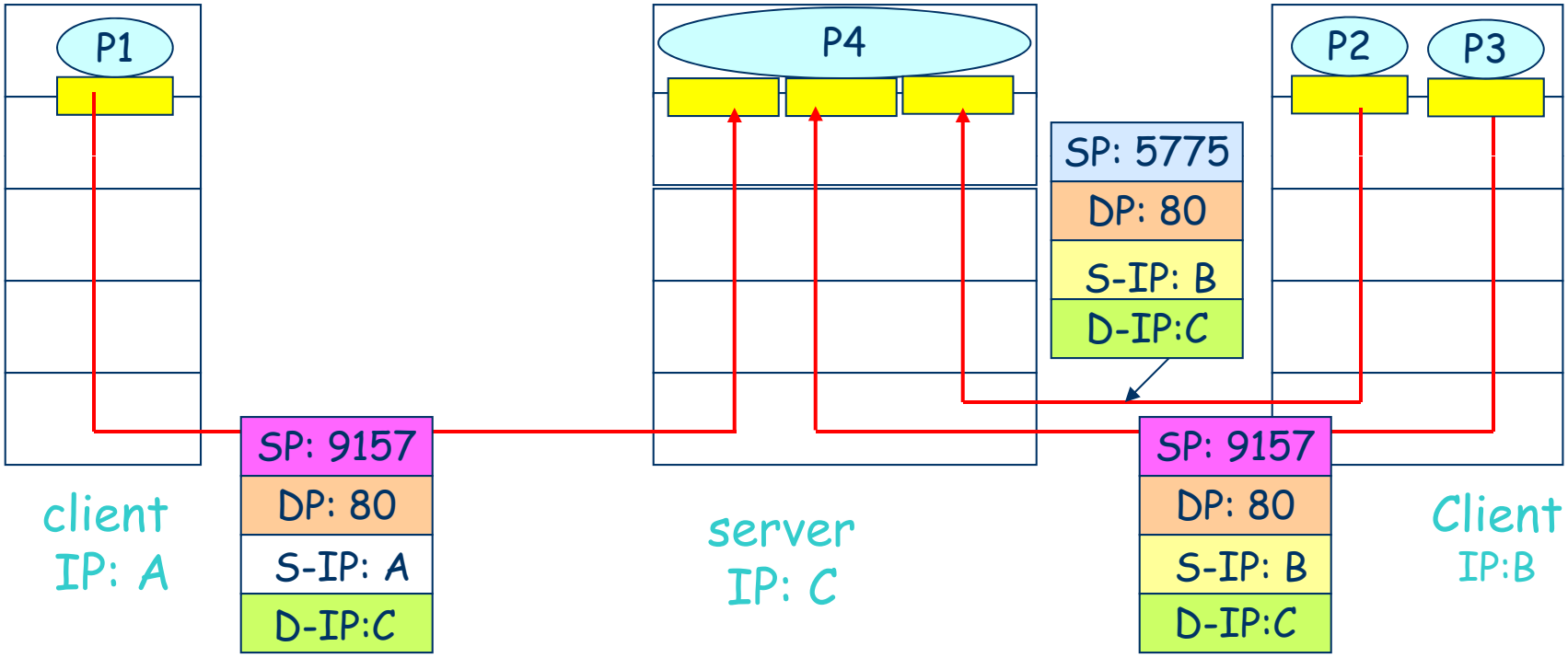
- Η υποδοχή TCP καθορίζεται από την τετράδα:
  - source IP address, source port number, dest IP address, dest port number
- Ο host λήψης χρησιμοποιεί και τις 4 τιμές για να κατευθύνει το τεμάχιο στην κατάλληλη υποδοχή
- Ένας Server host μπορεί να υποστηρίζει πολλές υποδοχές TCP ταυτόχρονα:
  - κάθε υποδοχή καθορίζεται από τη δική της τετράδα
- Οι Web servers έχουν διαφορετικές υποδοχές για κάθε συνδεόμενο client
  - στο μη-επίμονο HTTP έχουμε διαφορετική υποδοχή για κάθε αίτηση/απάντηση

Αποπολυπλεξία με σύνδεση



Αποπολυπλεξία με σύνδεση

Threaded Web Server



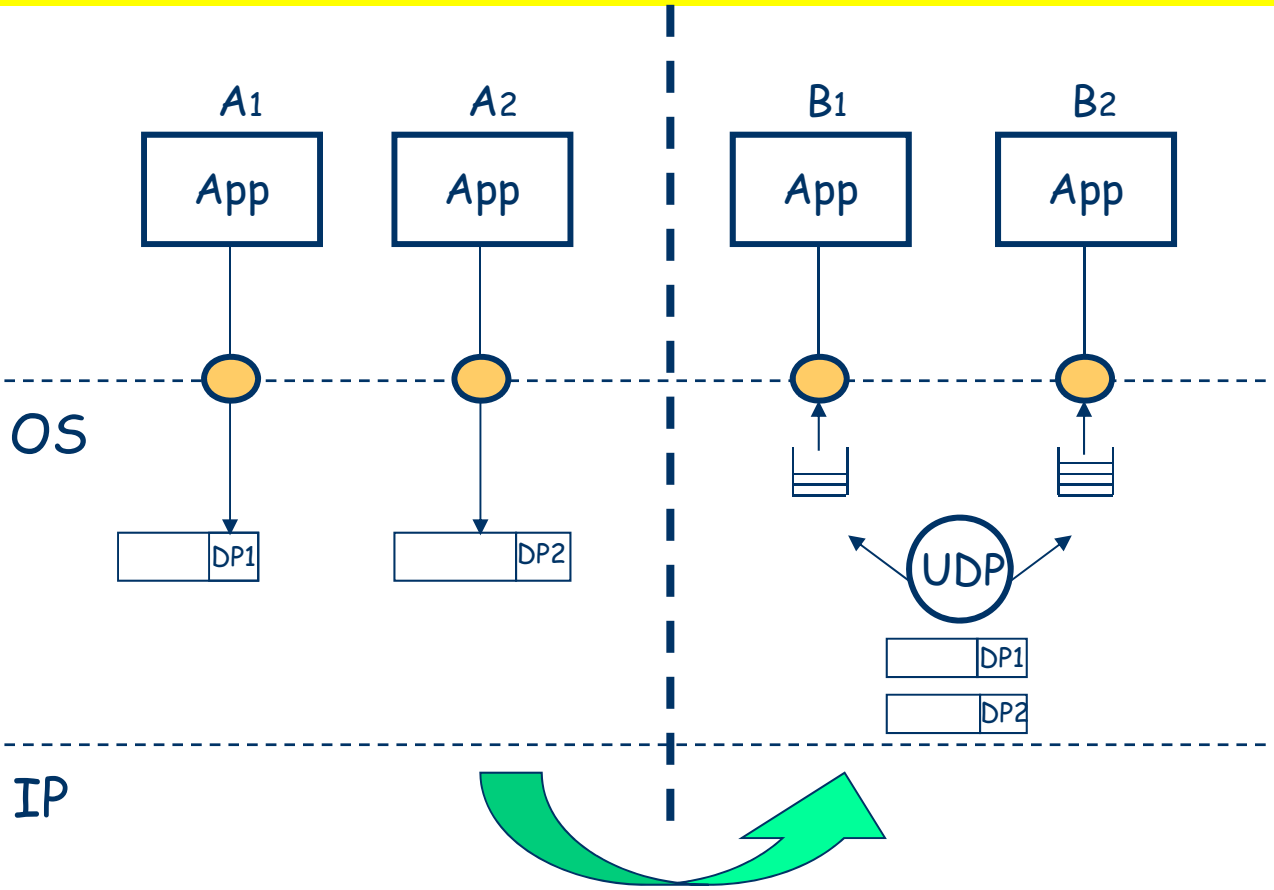
UDP: User Datagram Protocol [RFC 768]

- Λιτό πρωτόκολλο μεταφοράς στο Internet
- Υπηρεσία "καλύτερης προσπάθειας". Τα τεμάχια UDP μπορεί:
  - να χαθούν
  - να παραδοθούν εκτός σειράς στο ανώτερο στρώμα
- *Χωρίς σύνδεση:*
  - όχι χειραψία μεταξύ του πομπού και του δέκτη UDP
  - κάθε τεμάχιο UDP αντιμετωπίζεται ανεξάρτητα από τα άλλα

Γιατί υπάρχει το UDP;

- δεν εγκαθίσταται σύνδεση (που μπορεί να εισάγει καθυστέρηση)
- απλό: δεν διατηρείται κατάσταση σύνδεσης στον πομπό, δέκτη
- μικρή επικεφαλίδα στο τεμάχιο
- όχι έλεγχος συμφόρησης: το UDP μπορεί να στέλνει όσο γρήγορα μπορεί

UDP

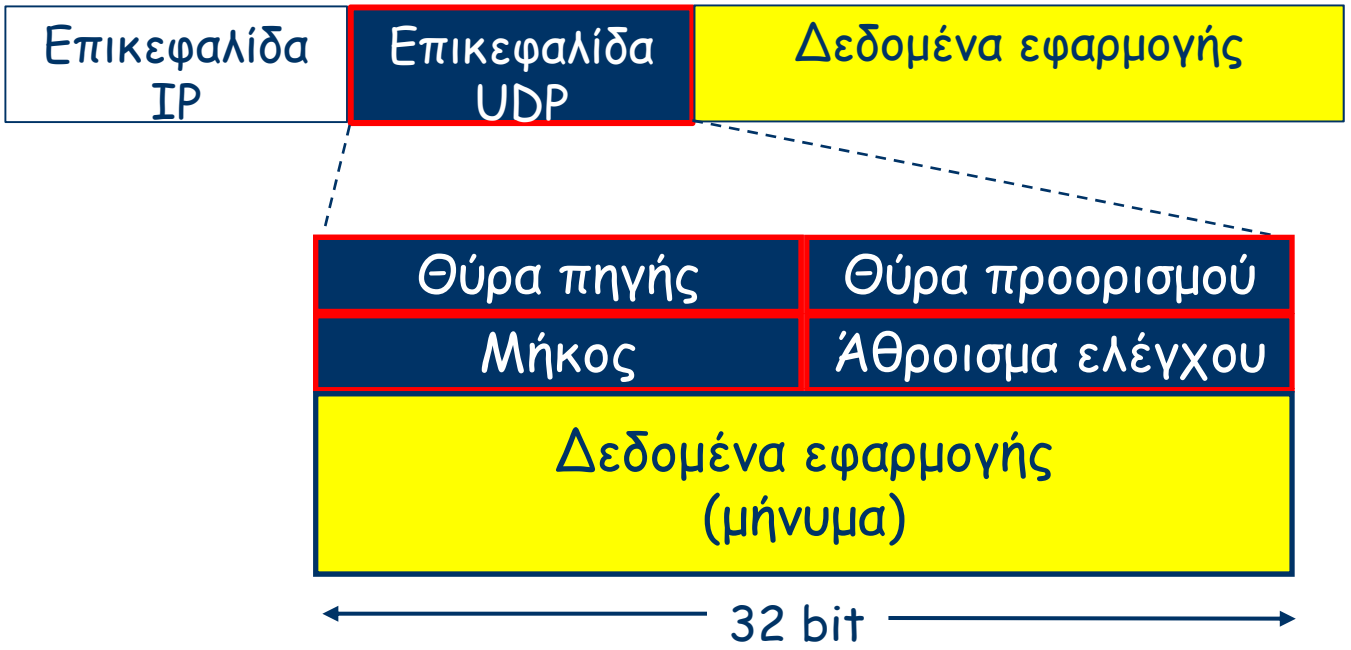


Το UDP, όπως και το TCP, χρησιμοποιεί αριθμούς θυρών για να αποπολυπλέξει τα τεμάχια

UDP

- μηνύματα μέχρι και 64KB
- παρέχει πολυπλεξία/αποπολυπλεξία στο IP
- πλεονεκτεί έναντι του TCP στο ότι δεν αυξάνει την καθυστέρηση απ' άκρη σ' άκρη πάνω από το IP
- χρησιμοποιείται σε εφαρμογές streaming multimedia
  - ανοχή σε απώλειες
  - ευαισθησία στον ρυθμό μετάδοσης
- άλλες χρήσεις του UDP
  - DNS
  - SNMP
  - RIP
- για αξιόπιστη μεταφορά με UDP, χρειάζεται προσθήκη αξιοπιστίας στο στρώμα εφαρμογής
  - αποκατάσταση λαθών ειδική για την εφαρμογή

Μορφή τεμαχίου UDP



- Οι **αριθμοί θυρών** προσδιορίζουν τις διαδικασίες αποστολής και λήψης. Η μέγιστη τιμή αριθμού θύρας είναι $2^{16}-1 = 65,535$
- Το **μήκος τεμαχίου** UDP είναι τουλάχιστον 8 byte (π.χ., άδειο πεδίο Data) και το πολύ 65,535 byte
- Το **άθροισμα ελέγχου** περιλαμβάνει την επικεφαλίδα του UDP και μερικά πεδία της επικεφαλίδας IP

Άθροισμα ελέγχου UDP

Στόχος: ανίχνευση σφαλμάτων στο μεταδιδόμενο τεμάχιο

- Συμπεριλαμβάνει την ψευδοεπικεφαλίδα

- Πομπός:
- μεταχειρίζεται τα περιεχόμενα του τεμαχίου ως ακολουθία ακεραίων των 16-bit
- checksum: πρόσθεση των περιεχομένων του τεμαχίου και λήψη του συμπληρώματος ως προς 1 του αθροίσματος
- ο πομπός τοποθετεί την τιμή του checksum στο πεδίο checksum του τεμαχίου UDP

- Δέκτης:
- υπολογίζει το checksum του λαμβανόμενου τεμαχίου συμπεριλαμβανομένου και του πεδίου checksum
- ελέγχει αν το υπολογισθέν checksum ισούται με την τιμή 1111111111111111:
  - ΟΧΙ - ανιχνεύθηκε σφάλμα
  - ΝΑΙ - δεν ανιχνεύθηκε σφάλμα. Αλλά μπορεί να υπάρχουν σφάλματα….

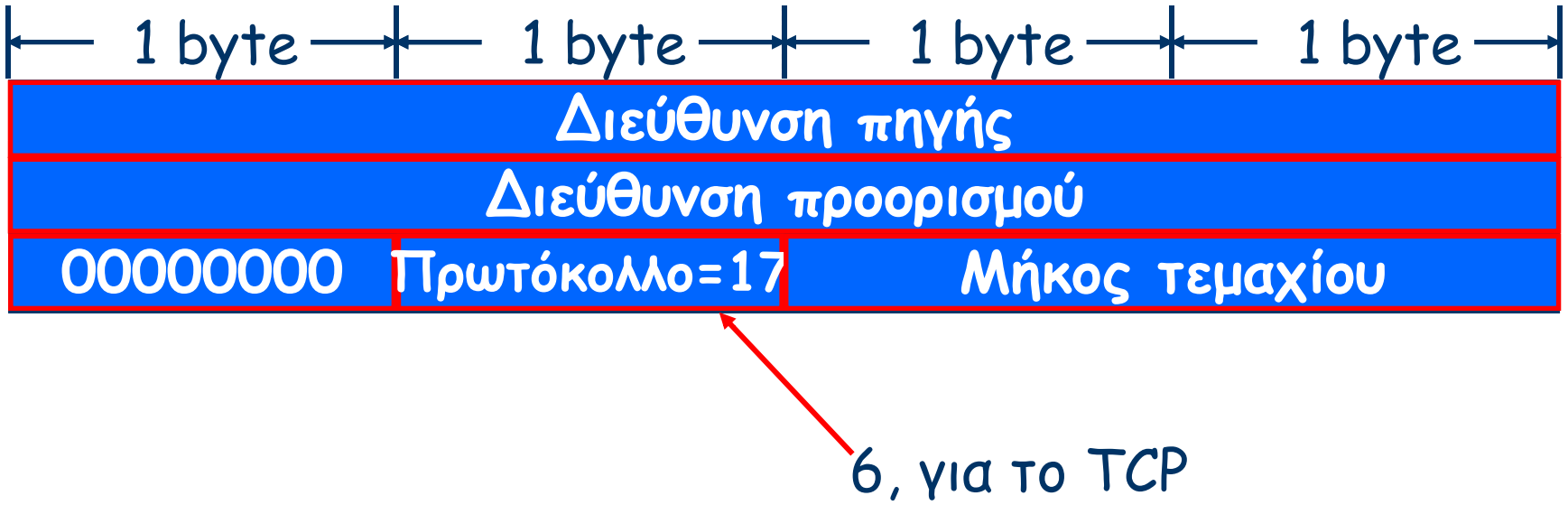
- Γιατί το UDP προβλέπει checksum;

Παράδειγμα αθροίσματος ελέγχου

- Σημείωση
  - Κατά την πρόσθεση αριθμών, το κρατούμενο από το πιο σημαντικό bit πρέπει να προστίθεται στο αποτέλεσμα
- Παράδειγμα: πρόσθεση δύο ακεραίων 16-bit

	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
wraparound	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
sum		1	0	1	1	1	0	1	1	1	0	1	1	1	1	0	0
																	1
checksum		0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1

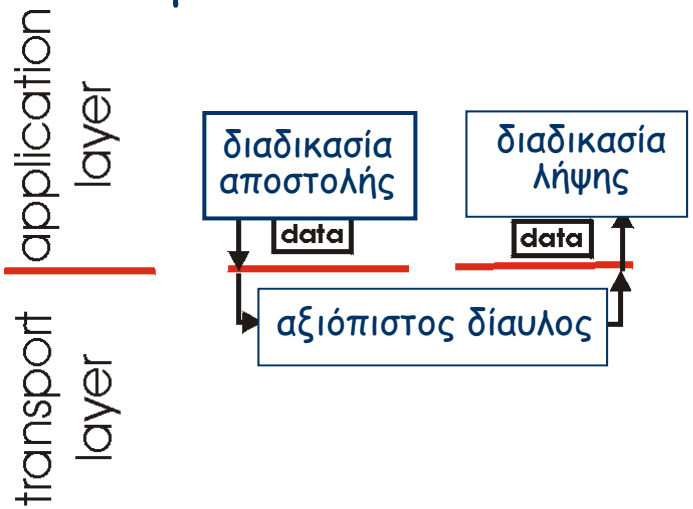
Ψευδοεπικεφαλίδα



- Η ψευδοεπικεφαλίδα εξασφαλίζει ότι το datagram έχει πραγματικά οδηγηθεί στο σωστό προορισμό, host και θύρα
- Χρησιμοποιείται μόνο για τον υπολογισμό του Checksum και δεν μεταδίδεται

Αξιόπιστη μετάδοση δεδομένων

- μας ενδιαφέρει για τα στρώματα εφαρμογής, μεταφοράς, ζεύξης δεδομένων και είναι από τα πιο ενδιαφέροντα θέματα στα δίκτυα.



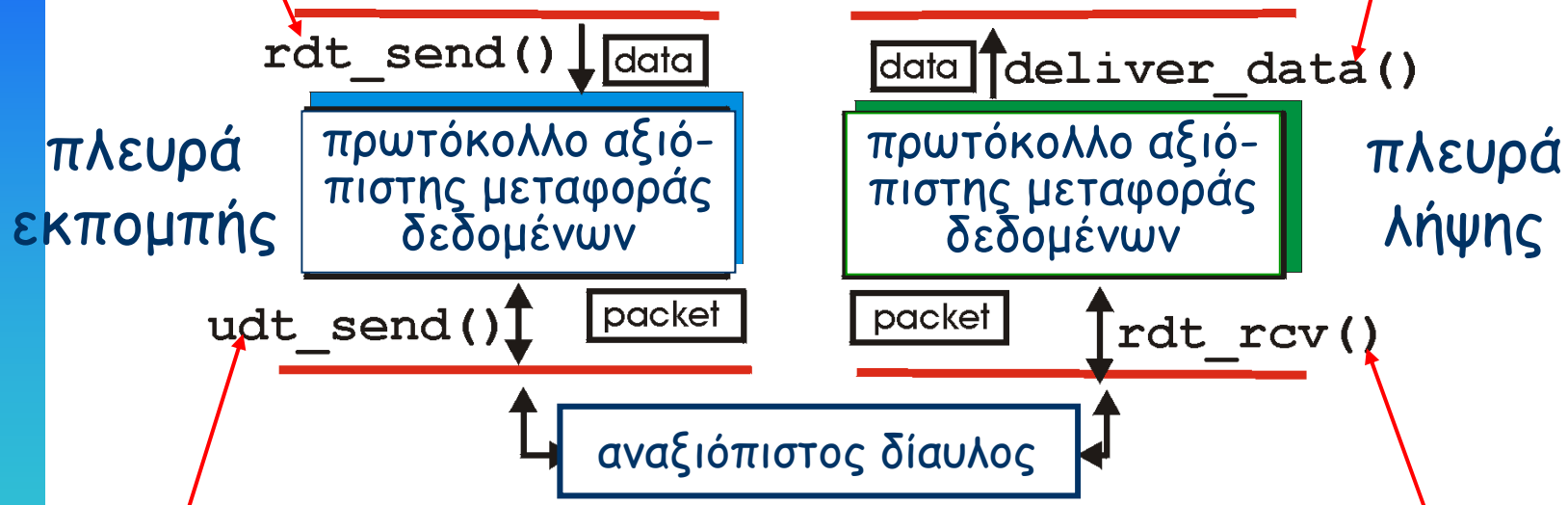
παρεχόμενη υπηρεσία

- τα χαρακτηριστικά το αναξιόπιστου διαύλου καθορίζουν την πολυπλοκότητα του πρωτοκόλλου αξιόπιστης μετάδοσης δεδομένων (rdt)

Αξιόπιστη μετάδοση δεδομένων

rdt_send(): καλείται από άνω (π.χ., από app.) για να παραδοθούν τα δεδομένα που προορίζονται για το ανώτερο στρώμα του δέκτη

deliver_data(): καλείται από την **rdt** για την παράδοση δεδομένων προς τα άνω



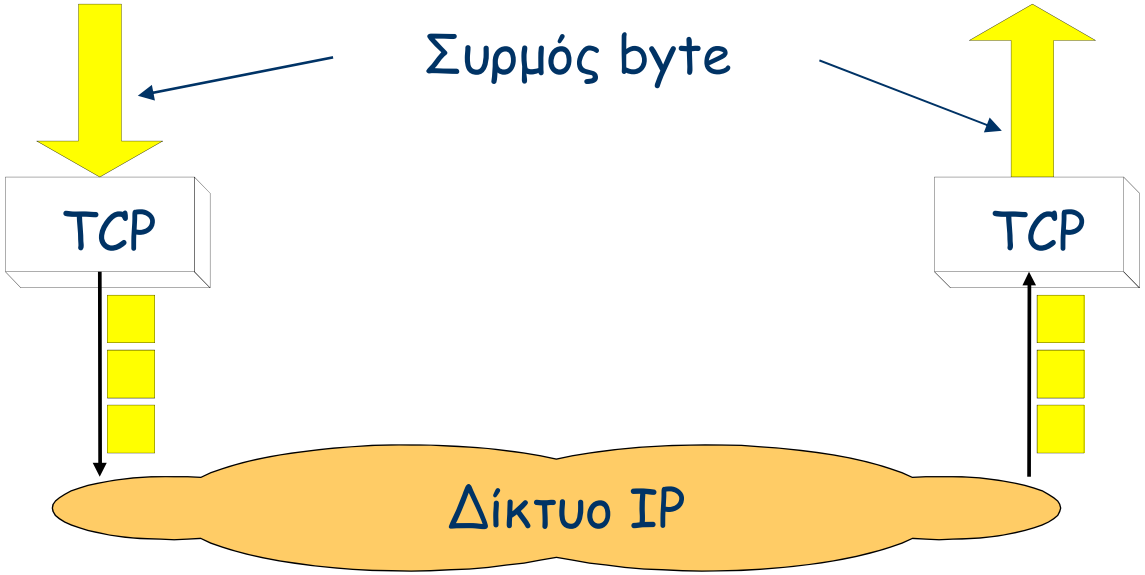
udt_send(): καλείται από την **rdt**, για τη μεταφορά πακέτου πάνω από τον αναξιόπιστο δίαυλο προς τον δέκτη

rdt_rcv(): καλείται όταν το πακέτο φθάνει στην πλευρά λήψης του διαύλου

TCP: Εισαγωγή

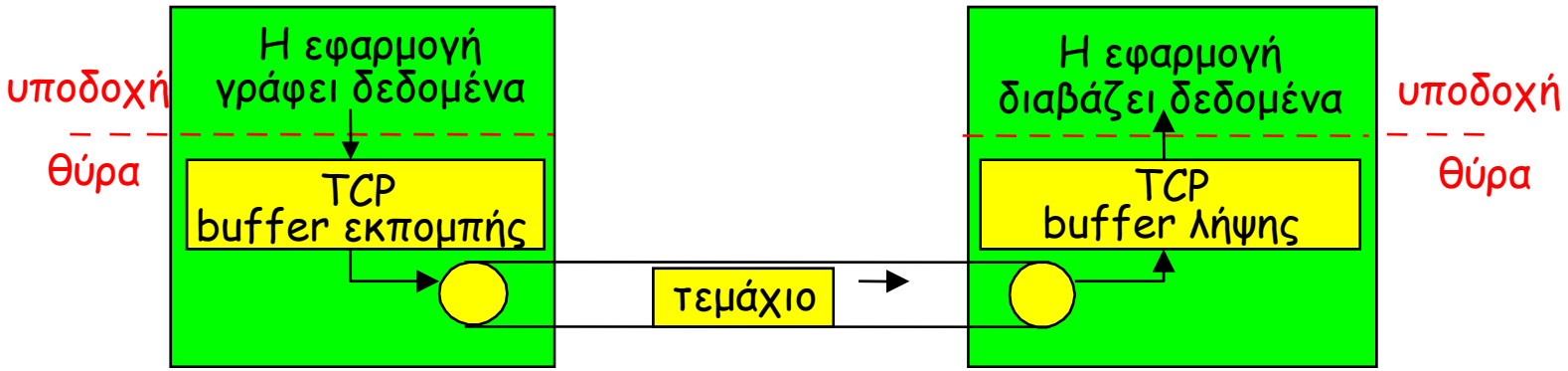
TCP = Transmission Control Protocol

- Είναι πρωτόκολλο με σύνδεση
- Παρέχει αξιόπιστη μετάδοση συρμού byte απ' άκρη σ' άκρη πάνω από μη αξιόπιστο δίκτυο.



TCP: Εισαγωγή

- μετάδοση σημείου προς σημείο:
  - ένας πομπός ένας δέκτης
- αξιόπιστη μετάδοση (παράδοση με τη σειρά) συρμού byte
  - το μήνυμα μπορεί να έχει αυθαίρετο μήκος
- συνεχής παροχή
  - ο έλεγχος συμφόρησης και ροής στο TCP καθορίζουν το μέγεθος του παραθύρου
- buffers εκπομπής και λήψης

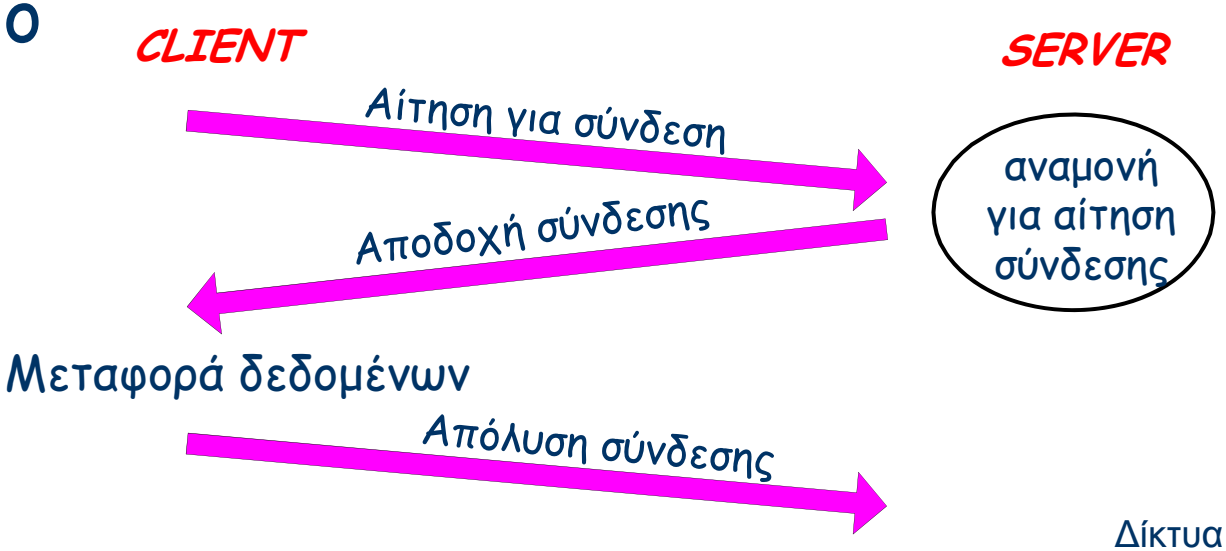


TCP: Εισαγωγή

- αμφίδρομη μετάδοση
  - αμφίδρομη ροή δεδομένων στην ίδια σύνδεση
  - MSS: maximum segment size
- υπηρεσία με σύνδεση
  - τριμερής χειραψία αρχικοποιεί την κατάσταση πομπού και δέκτη πριν την ανταλλαγή δεδομένων
- έλεγχος ροής
  - ο πομπός δεν υπερχειλίζει τον δέκτη
- παρέχει πολυπλεξία/αποπολυπλεξία πάνω από το IP
- έλεγχος και αποφυγή συμφόρησης
- παραδείγματα χρησιμοποίησης: file transfer, chat, web, SMTP (e-mail)

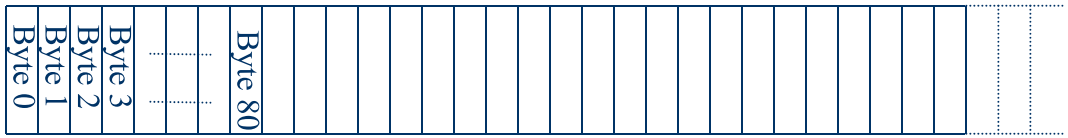
TCP: Πρωτόκολλο με σύνδεση

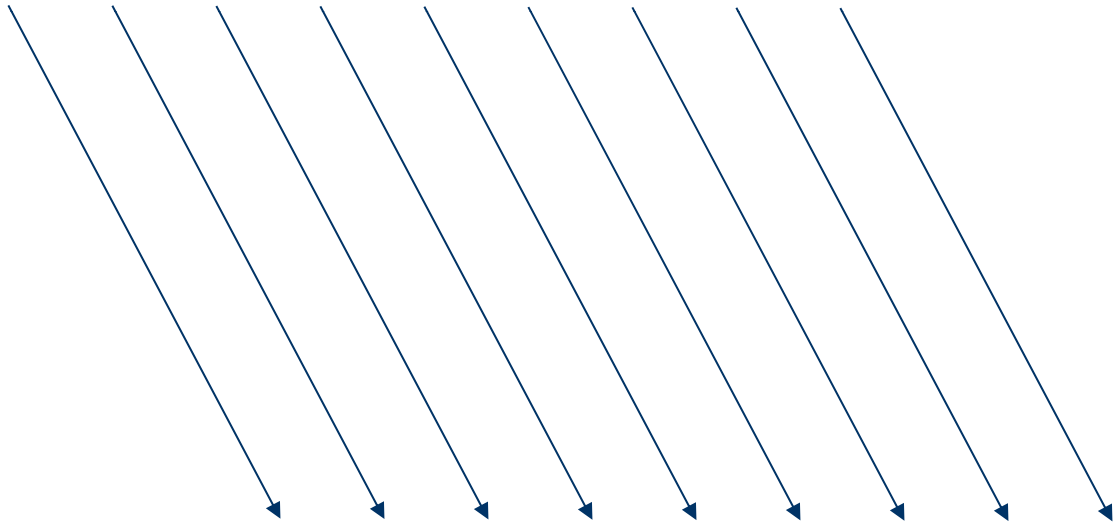
- Πριν από οποιανδήποτε μεταφορά δεδομένων, το TCP εγκαθιστά μια **σύνδεση**:
  - Η μια οντότητα TCP αναμένει για σύνδεση ("**server**")
  - Η άλλη οντότητα TCP ("**client**") συνδέεται με τον server
- Η διαδικασία εγκατάστασης σύνδεσης είναι στην πραγματικότητα πιο πολύπλοκη.
- Κάθε σύνδεση είναι αμφίδρομη, σημείου προς σημείο



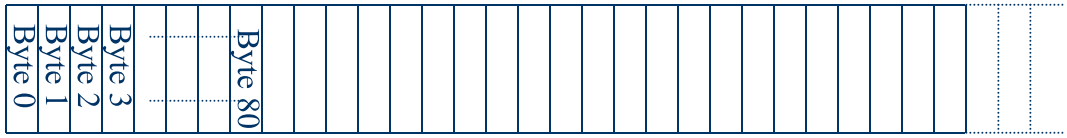
TCP: Υπηρεσία συρμού byte

Host A



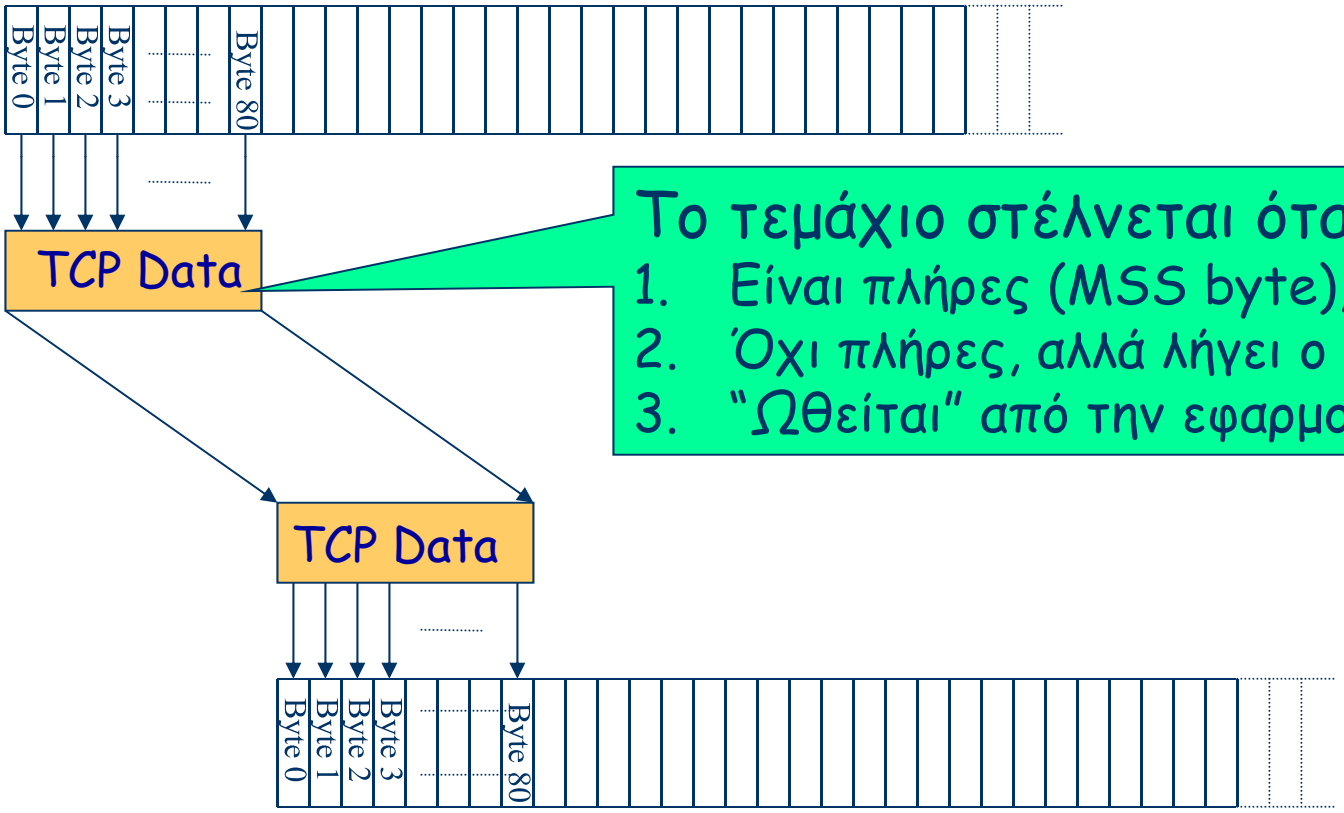


Host B



TCP: Υπηρεσία συρμού byte

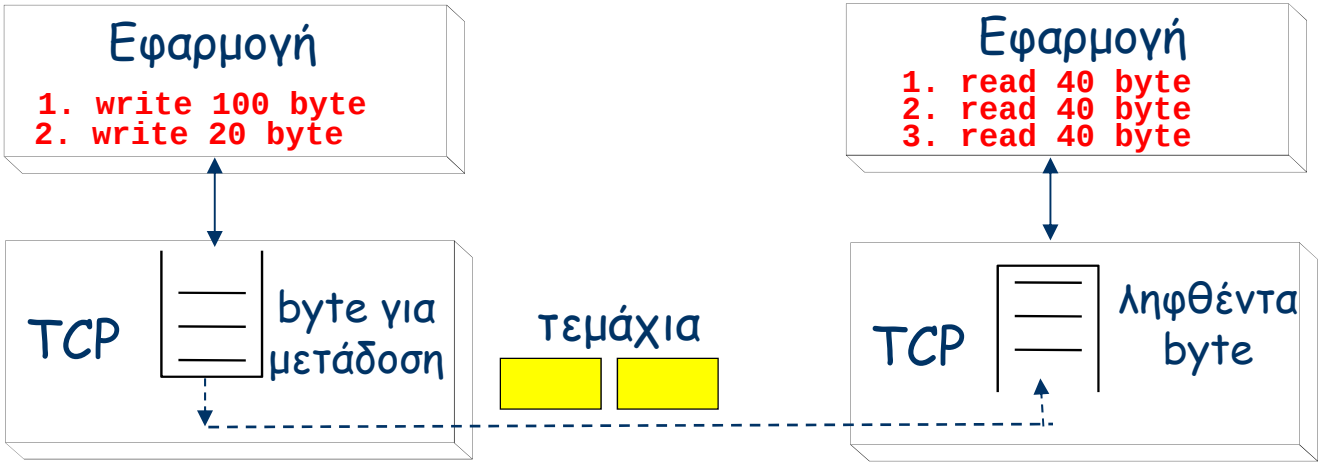
Host A



Host B

TCP: Υπηρεσία συρμού byte

- Στα κατώτερα στρώματα, το TCP παραδίδει δεδομένα σε τμήματα, τα τεμάχια.
- Στα ανώτερα στρώματα, το TCP παραδίδει δεδομένα ως ακολουθία από byte και δεν καθορίζει όρια μεταξύ των byte
- Συνεπώς, τα ανώτερα στρώματα δεν γνωρίζουν την αρχή και το τέλος των τεμαχίων!

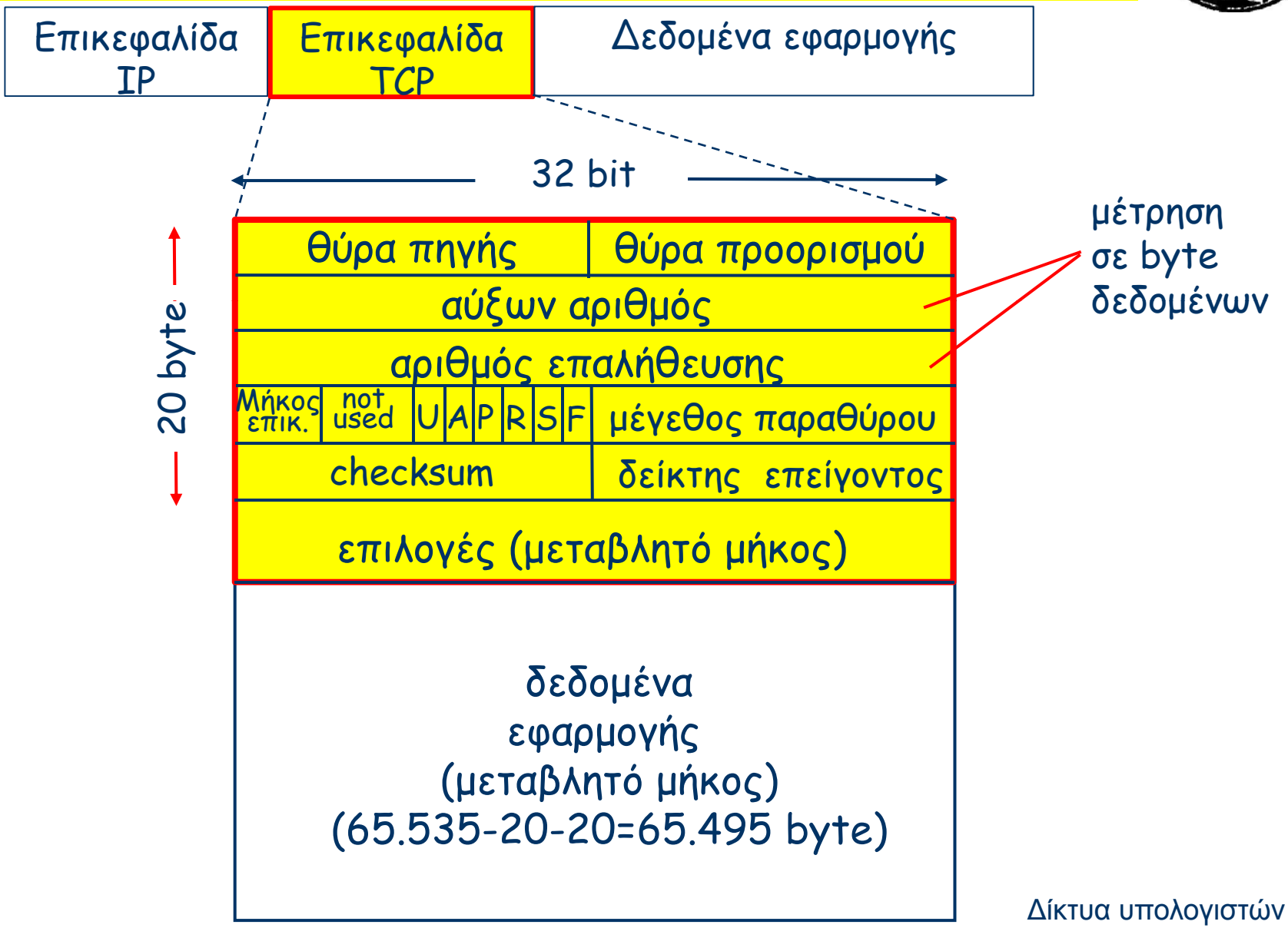


TCP: Υπηρεσία συρμού byte

MSS: Maximum segment size

- Εξαρτάται από την υλοποίηση του TCP (καθοριζόμενη από το OS) και μπορεί να επιλεγεί
- Συνήθεις τιμές: 1500, 536, 512 byte
- Επιλέγεται το μέγιστο μέγεθος κατά τρόπο που να αποφεύγεται ο θρυμματισμός στο IP
- MSS είναι ο μέγιστος αριθμός δεδομένων του στρώματος εφαρμογής που περιέχονται στο τεμάχιο και όχι το μέγιστο μέγεθος του τεμαχίου

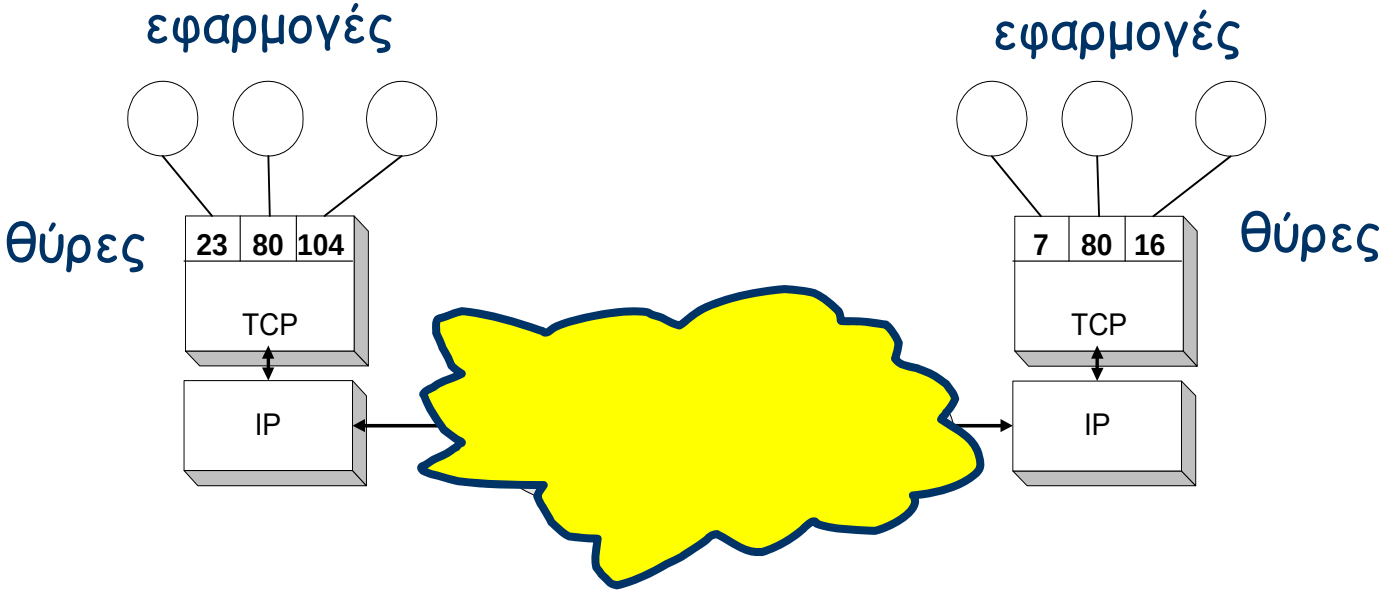
TCP: Δομή τεμαχίου



TCP: Δομή τεμαχίου

Αριθμοί Θυρών

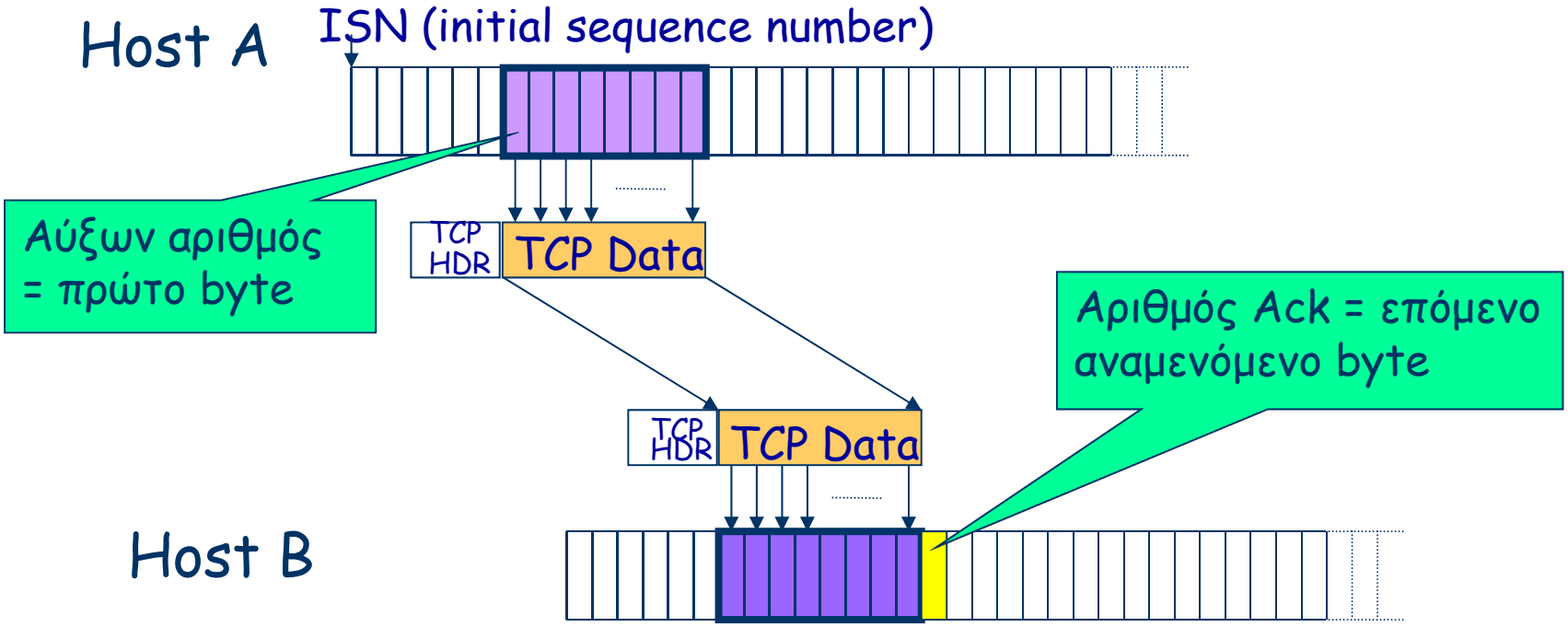
- Ο αριθμός θύρας προσδιορίζει την υποδοχή μιας εφαρμογής.
- Ένα ζεύγος (IP address, port number) προσδιορίζει το ένα άκρο μια σύνδεσης.
- Δύο ζεύγη (client IP address, client port number) και (server IP address, server port number) προσδιορίζουν μια σύνδεση TCP.



TCP: Δομή τεμαχίου

Αύξοντες αριθμοί και επαληθεύσεις

- Οι αύξοντες αριθμοί και οι επαληθεύσεις στο TCP έχουν μήκος 32 bit.
- Η περιοχή τιμών είναι $0 \le$ Sequence number $\le 2^{32} - 1 \approx 4.3$ Gbyte
- Ο client και ο server επιλέγουν ο καθένας τους τον ISN τυχαία κατά την εγκατάσταση της σύνδεσης.



TCP: Δομή τεμαχίου

Αριθμός επαλήθευσης

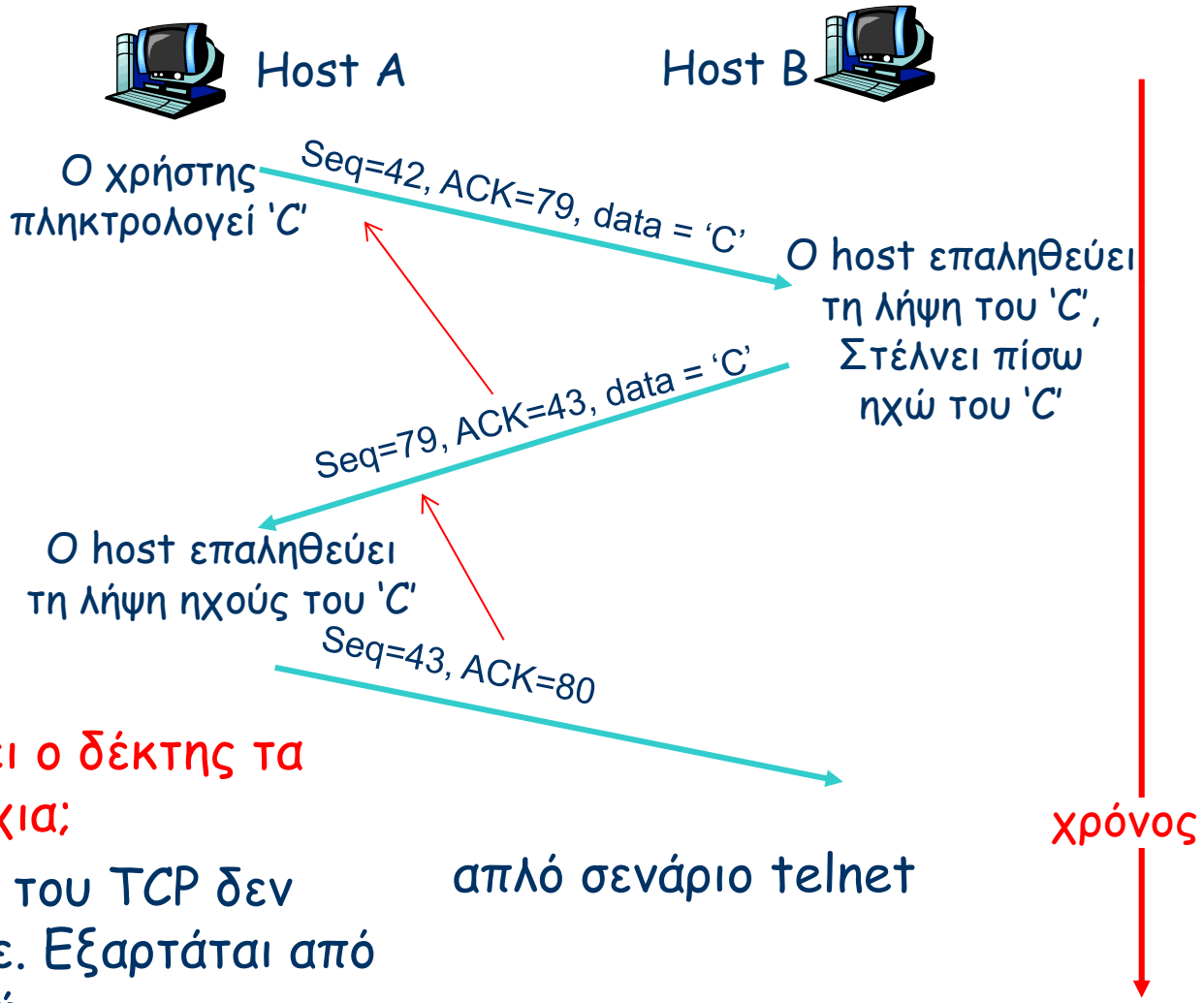
- Το TCP χρησιμοποιεί παραλλαγή του πρωτοκόλλου ολισθαίνοντος παραθύρου για τον έλεγχο ροής μεταξύ πομπού και δέκτη
  - όχι NACK (**N**egative **ACK**nowledgement)
  - μόνο συσσωρευτικές ACK
- Παράδειγμα:

  Ο πομπός στείλει δύο τεμάχια με "1...1500" και "1501...3000", αλλά ο δέκτης λαμβάνει μόνο το δεύτερο.

  Ο δέκτης δεν μπορεί να επαληθεύσει το δεύτερο τεμάχιο. Μπορεί να στείλει μόνο AckNo = 1

TCP: Δομή τεμαχίου

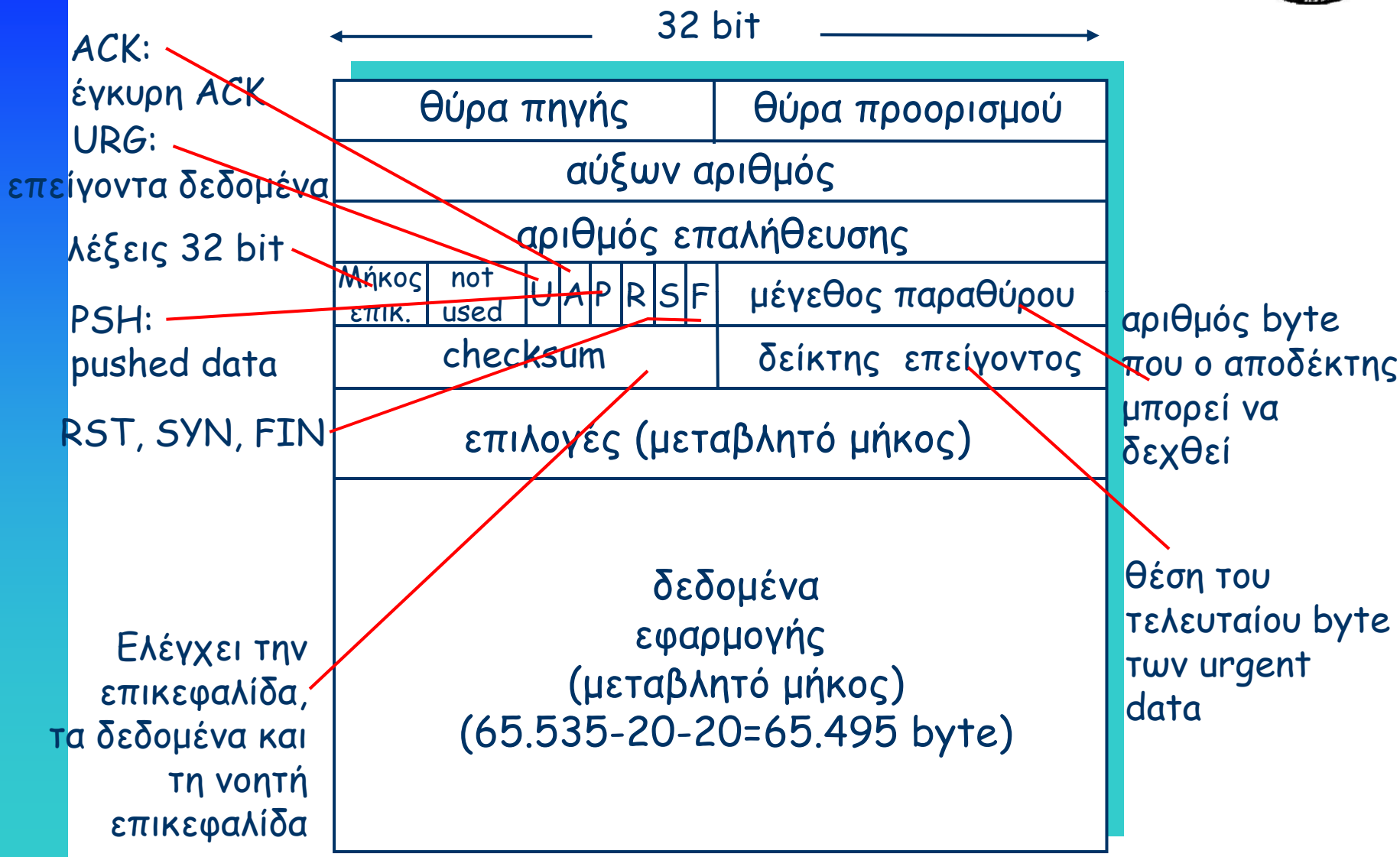
Αύξοντες αριθμοί: παράδειγμα Telnet



απλό σενάριο telnet

- Πώς αντιμετωπίζει ο δέκτης τα εκτός σειράς τεμάχια;
  - Οι προδιαγραφές του TCP δεν αναφέρουν τίποτε. Εξαρτάται από τον κατασκευαστή

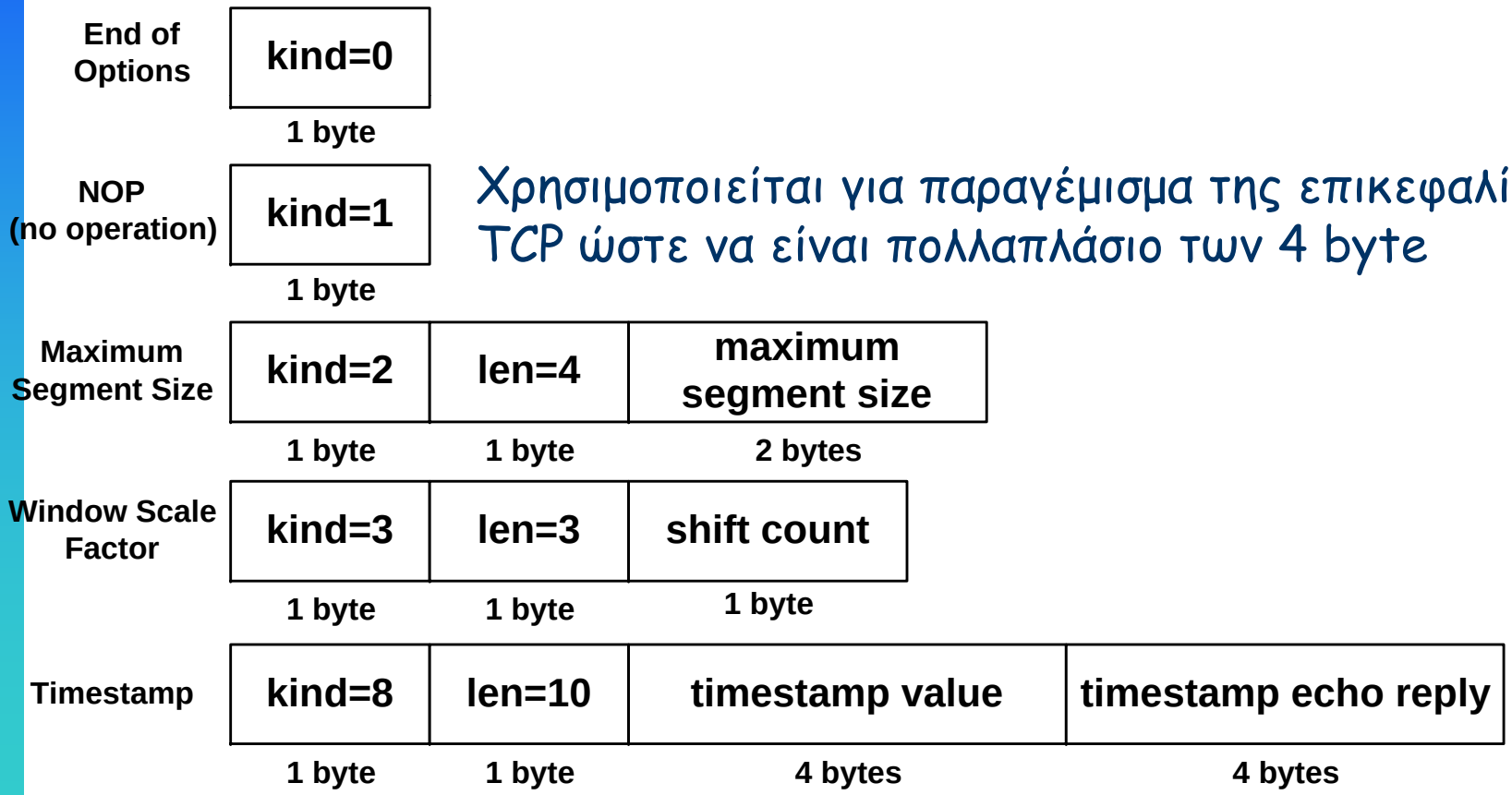
TCP: Δομή τεμαχίου



TCP: Δομή τεμαχίου

Προαιρετικές επιλογές (options)

- είναι ένας τρόπος να προστεθούν επιπλέον δυνατότητες που δεν καλύπτονται από την κανονική επικεφαλίδα



TCP: Αξιόπιστη μετάδοση

- Το TCP δημιουργεί υπηρεσία αξιόπιστης μετάδοσης πάνω από την αναξιόπιστη υπηρεσία του IP
- Στέλνει τεμάχια με συνεχή παροχή
- Δύο τύποι σφαλμάτων:
  - **Απωλεσθέντα τεμάχια**
  - **Κατεστραμμένα τεμάχια**
- Το TCP έχει αθροίσματα ελέγχου για την επικεφαλίδα και τα δεδομένα. Τεμάχια με μη έγκυρα αθροίσματα ελέγχου απορρίπτονται
- Ο δέκτης στέλνει επαληθεύσεις (ACKs) για τα σωστά τεμάχια. Οι ACK μπορεί να είναι συσσωρευτικές.

TCP: Αξιόπιστη μετάδοση

- Το TCP χρησιμοποιεί μοναδικό χρονόμετρο επαναμετάδοσης
- Οι επαναμεταδόσεις των τεμαχίων προκαλούνται από λήξεις χρόνου ή διπλές ACK
- Ο αριθμός της ACK είναι ο επόμενος αναμενόμενος αύξων αριθμός
- Καθυστερημένη ACK: ο δέκτης TCP συνήθως καθυστερεί τη μετάδοση μιας ACK (για περίπου 200ms)
- Οι ACK δεν καθυστερούνται όταν τα πακέτα λαμβάνονται εκτός σειράς

TCP: Αξιόπιστη μετάδοση

Απλοποιημένος πομπός TCP

Άφιξη δεδομένων από το στρώμα εφαρμογής:

- Δημιουργία τεμαχίου με seq #
- Ο seq # αντιστοιχεί στο πρώτο byte του συρμού δεδομένων
- Εκκίνηση χρονομέτρου εάν δεν ξεκίνησε ήδη (χρονόμετρο για προηγούμενο ανεπιβεβαίωτο τεμάχιο)
- Χρόνος εκπνοής: `TimeOutInterval`

Εκπνοή χρόνου:

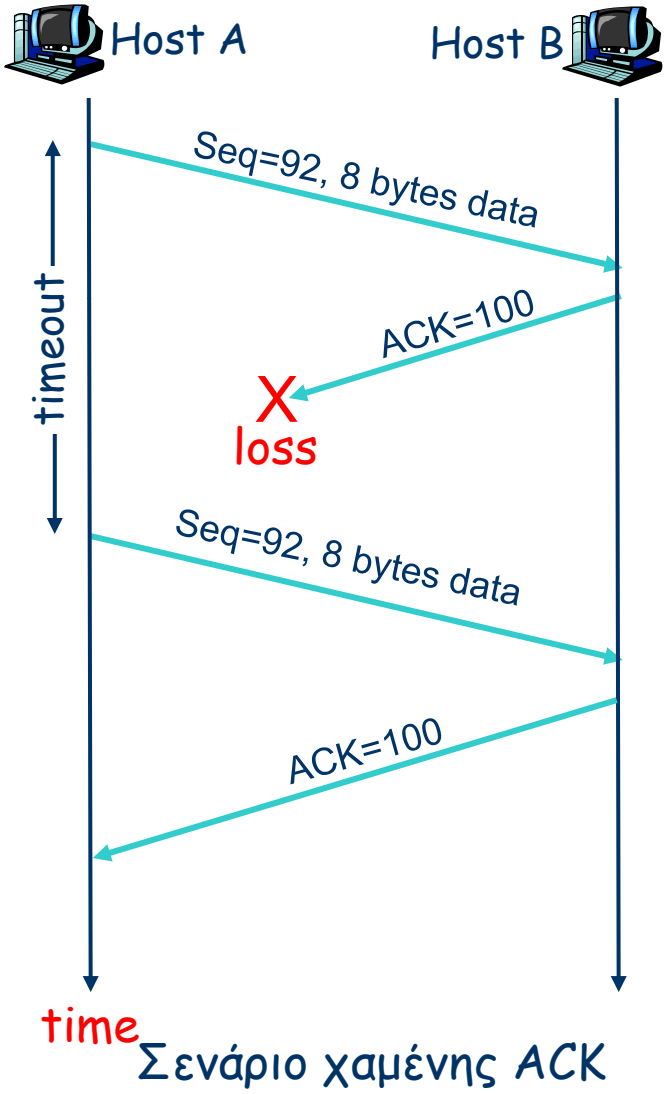
- Επανεκπομπή του τεμαχίου που προκάλεσε timeout
- Επανεκκίνηση του timer

Λήψη Ack:

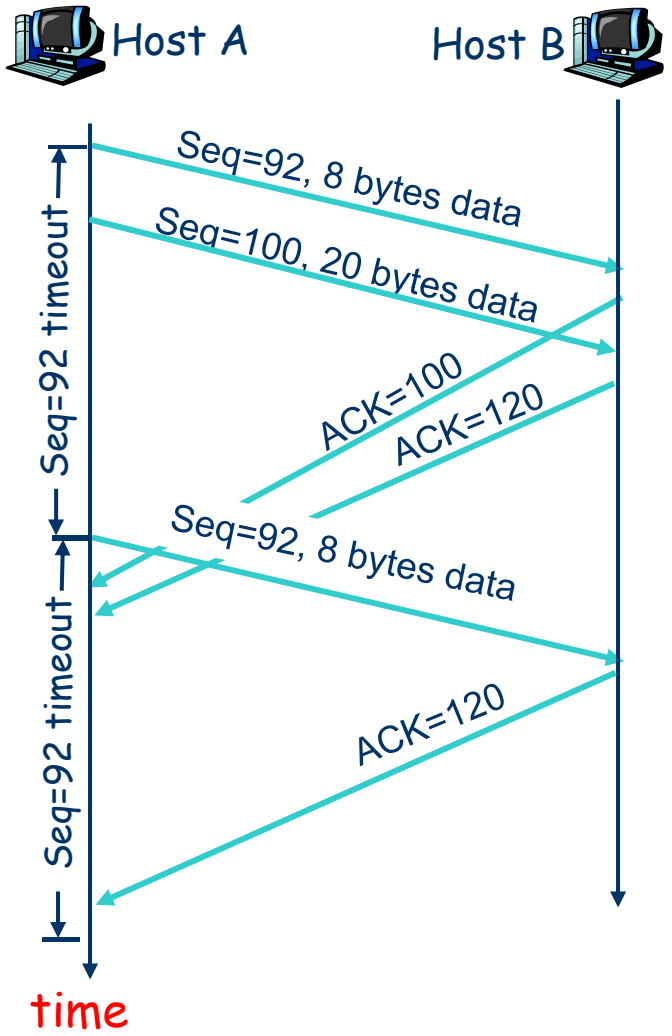
- Αν επαληθεύει προγούμενα ανεπαλήθευτα τεμάχια
  - ενημέρωση των ήδη επαληθευθέντων
  - εκκίνηση του timer αν υπάρχουν εκκρεμούντα τεμάχια

TCP: Αξιόπιστη μετάδοση

Σενάρια επαναμετάδοσης

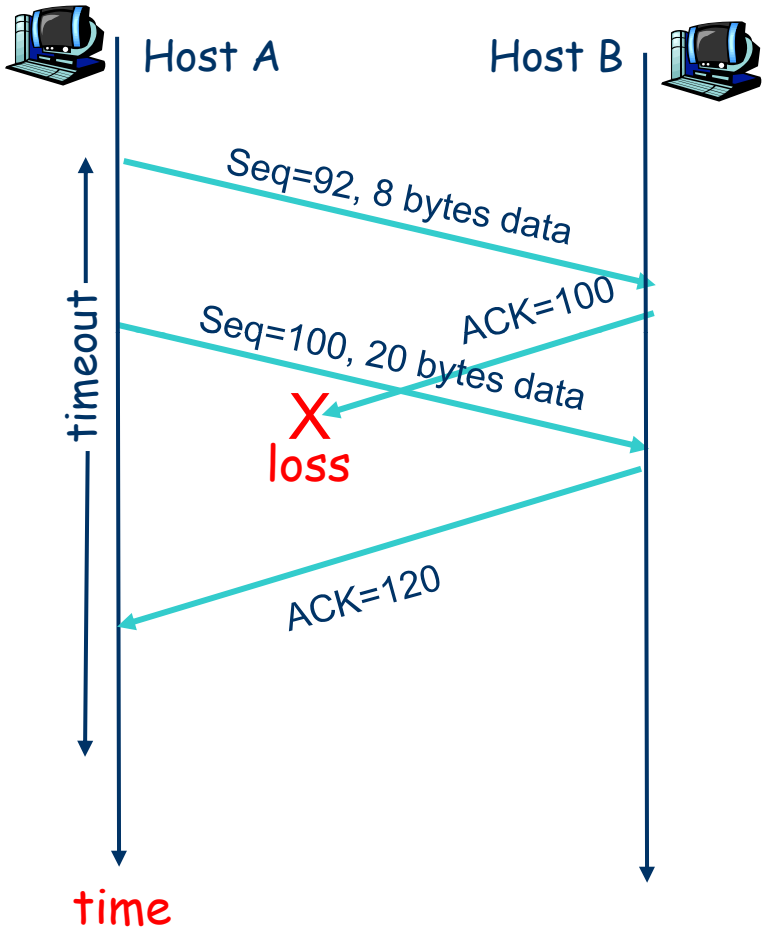


Σενάριο χαμένης ACK



Πρόωρο timeout

TCP: Αξιόπιστη μετάδοση



Σενάριο συγκεντρωτικής ACK

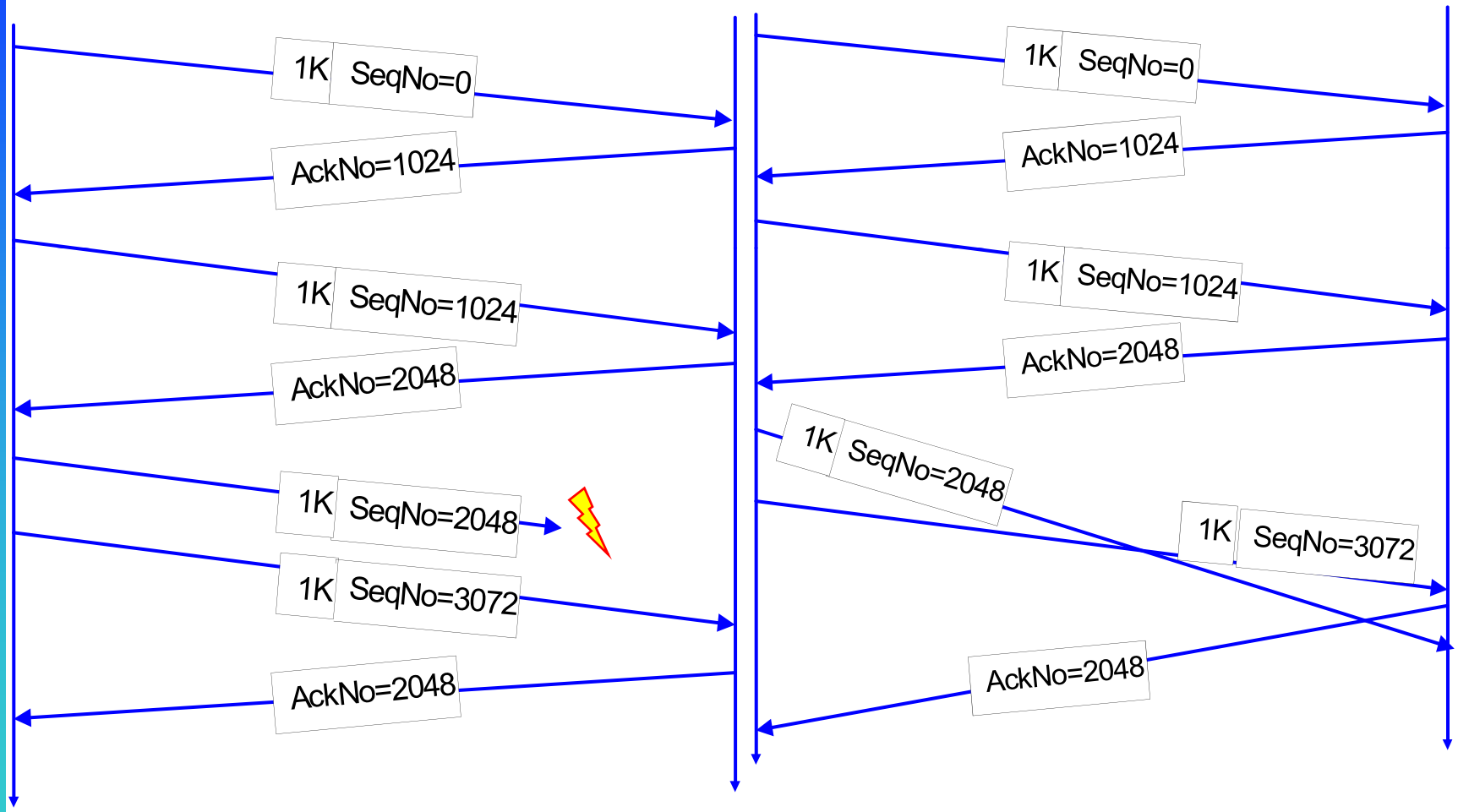
TCP: Αξιόπιστη μετάδοση

Δημιουργία επαληθεύσεων

Γεγονός	Ενέργεια του δέκτη TCP
άφιξη τμαχίου στην κανονική σειρά, όχι κενά, οτιδήποτε άλλο έχει ήδη επαληθευτεί	καθυστερημένη ACK. Αναμονή 200ms για το επόμενο τμάχιο. Αν δεν υπάρχει επόμενο τμάχιο, στέλνει ACK
άφιξη τμαχίου στην κανονική σειρά, όχι κενά, εκκρεμεί μία καθυστερημένη ACK	άμεση αποστολή μιας συσσωρευτικής ACK και για τα δύο τμάχια που αφίχθηκαν με κανονική σειρά
άφιξη τμαχίου εκτός σειράς με μεγαλύτερο αύξοντα αριθμό από τον αναμενόμενο	αποστολή επαναληπτικής ACK, που να δείχνει τον αύξοντα αριθμό του επόμενου αναμενόμενου byte
άφιξη τμαχίου που η αρίθμησή του είναι εν μέρει ή πλήρως εντός του αναμενόμενου διαστήματος	άμεση αποστολή ACK, αν το τμάχιο αρχίζει στο κατώτερο άκρο του αναμενόμενου διαστήματος αρίθμησης

TCP: Αξιόπιστη μετάδοση

Επαληθεύσεις στο TCP



Απώλεια τεμαχίου

Άφιξη εκτός σειράς

TCP: Αξιόπιστη μετάδοση

Επαναμεταδόσεις στο TCP

Ένας πομπός TCP επαναμεταδίδει ένα τεμάχιο, όταν θεωρήσει ότι το υπόψη τεμάχιο έχει χαθεί.

- Δεν έχει ληφθεί ACK και έχει λήξει το χρονόμετρο
- Έχουν ληφθεί πολλαπλές ACK για το ίδιο τεμάχιο

TCP: Αξιόπιστη μετάδοση

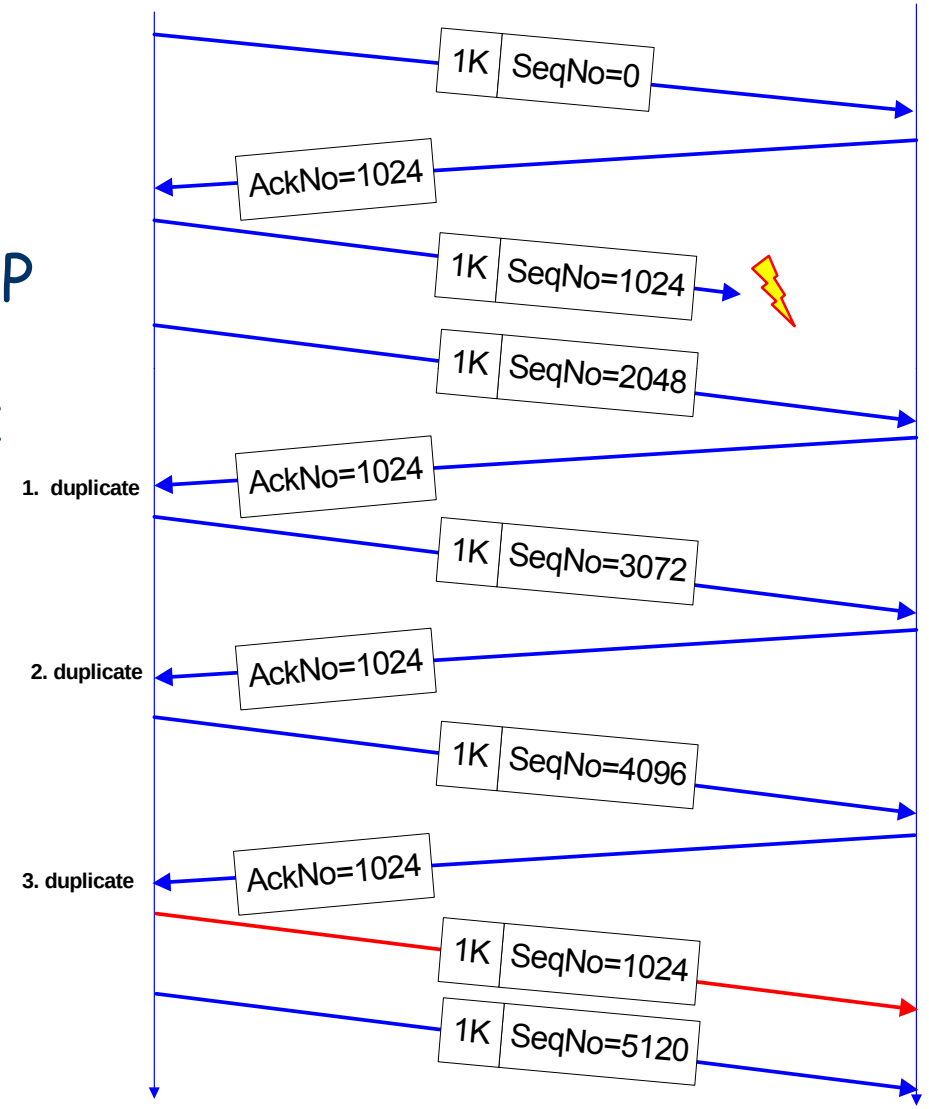
Ταχεία επαναμετάδοση

- Η περίοδος timeout είναι συχνά σχετικά μεγάλη:
  - μεγάλη καθυστέρηση πριν την αποστολή του χαμένου πακέτου
- Ανίχνευση των χαμένων τεμαχίων μέσω διπλών επαληθεύσεων.
  - Ο πομπός στέλνει συχνά πολλά τεμάχια το ένα πίσω απ' το άλλο
  - Αν χαθεί τεμάχιο, θα υπάρχουν ενδεχομένως πολλές ίδιες επαληθεύσεις.
- Ταχεία επαναμετάδοση: επαναποστολή του τεμαχίου πριν τη λήξη της χρονομέτρησης

TCP: Αξιόπιστη μετάδοση

Λήψη διπλών ACK

- Αν ληφθούν τρεις ACK στη σειρά για τα ίδια δεδομένα, ο πομπός TCP θεωρεί ότι το τεμάχιο μετά τα επαληθευόμενα δεδομένα χάθηκε.
- Τότε ο πομπός TCP επαναμεταδίδει το τεμάχιο που θεωρεί ότι χάθηκε, χωρίς να περιμένει τη λήξη χρόνου.
- Τούτο διορθώνει μεμονωμένες απώλειες τεμαχίων



TCP: Αξιόπιστη μετάδοση

Αλγόριθμος ταχείας επαναμετάδοσης

```
γεγονός: λήψη ACK, με τιμή της ACK ίση με y
            if (y > SendBase) {
                SendBase = y
                if (υπάρχουν τρέχοντα μη επαληθευθέντα ήδη τεμάχια)
                    εκκίνηση χρονομετρητή
              }
            else {
                αύξηση του μετρητή των διπλών ACKs που ελήφθησαν
                        για το y
                if (μετρητής ληφθεισών διπλών ACKs για το y = 3)
                    επανεκπομπή του τεμαχίου με αύξοντα αριθμό y
                }
```

διπλή ACK για ήδη επαληθευθέν τεμάχιο

Ταχεία επαναμετάδοση

SendBase-1: τελευταίο επαληθευθέν byte

TCP: Διαχείριση χρονομετρητών

- Το TCP χρησιμοποιεί πολλούς χρονομετρητές. Ο σπουδαιότερος είναι ο χρονομετρητής επαναμετάδοσης (retransmission timer)
- Όταν στέλνεται ένα τεμάχιο, ξεκινά ένας χρονομετρητής αναμετάδοσης
- Αν η λήψη του τεμαχίου επαληθευτεί πριν εκπνεύσει ο χρόνος, τότε ο χρονομετρητής σταματά
- Αν εκπνεύσει ο χρόνος πριν φθάσει η επαλήθευση, το τεμάχιο μεταδίδεται ξανά
- Πόσο μεγάλο πρέπει να είναι το χρονικό διάστημα πριν λήξει η χρονομέτρηση;

TCP: Διαχείριση χρονομετρητών

Χρόνοι Round Trip και Timeout

Πώς τίθεται η τιμή του timeout επαναμετάδοσης (Retransmission Timeout, RTO) στο TCP;

- μεγαλύτερη από RTT
  - αλλά το RTT μεταβάλλεται
- πολύ μικρό: πρόωρο timeout
  - Άσκοπες επαναμεταδόσεις
- πολύ μεγάλο: αργή αντίδραση, όταν χάνεται τεμάχιο

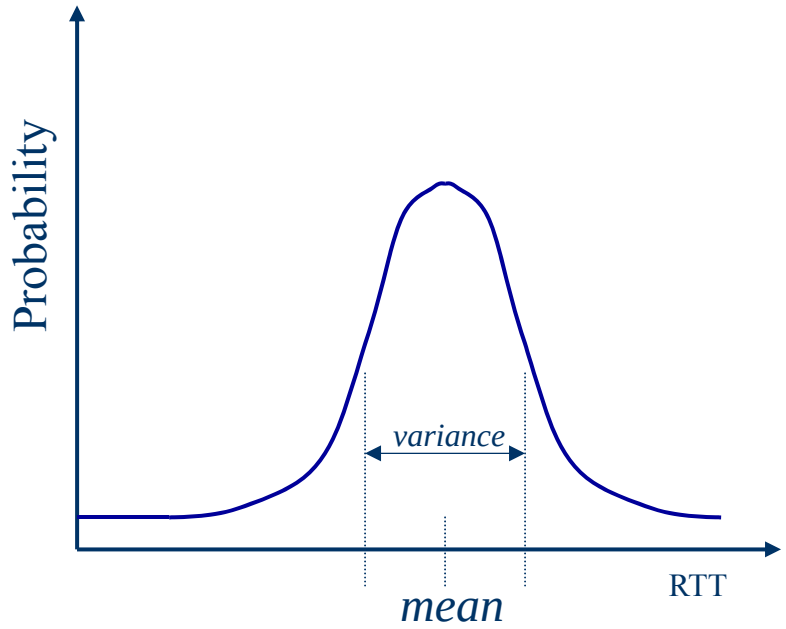
Πώς προσδιορίζεται το RTT;

- SampleRTT: ο χρόνος από τη μετάδοση του τεμαχίου μέχρι τη λήψη της ACK
- Επειδή το SampleRTT μεταβάλλεται, είναι επιθυμητή μια εξομαλυμένη τιμή για το RTT, όχι το τρέχον SampleRTT

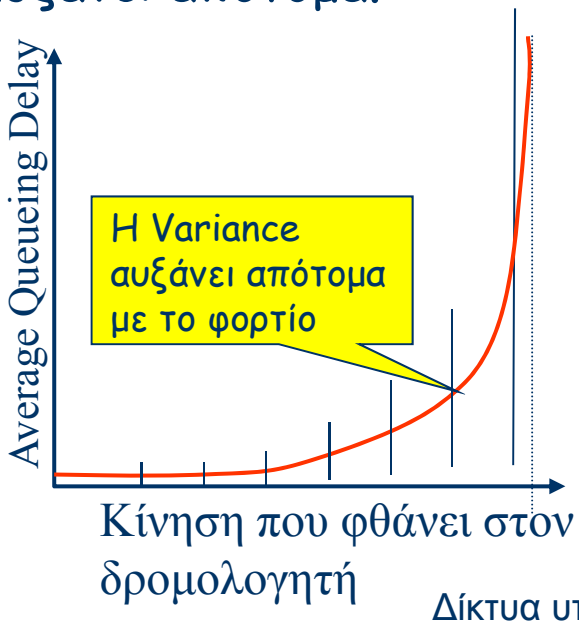
TCP: Διαχείριση χρονομετρητών

Χρόνοι Round Trip και Timeout

- Θα υπάρχει κάποια (άγνωστη) κατανομή των RTT.
- Προσπαθούμε να εκτιμήσουμε ένα RTO για να ελαχιστοποιήσουμε την πιθανότητα μιας εσφαλμένης λήξης χρόνου.



- Οι ουρές στους δρομολογητές μεγαλώνουν όταν υπάρχει περισσότερη κίνηση, μέχρι να γίνουν ασταθείς.
- Καθώς αυξάνει το φορτίο, η variance της καθυστέρησης αυξάνει απότομα.

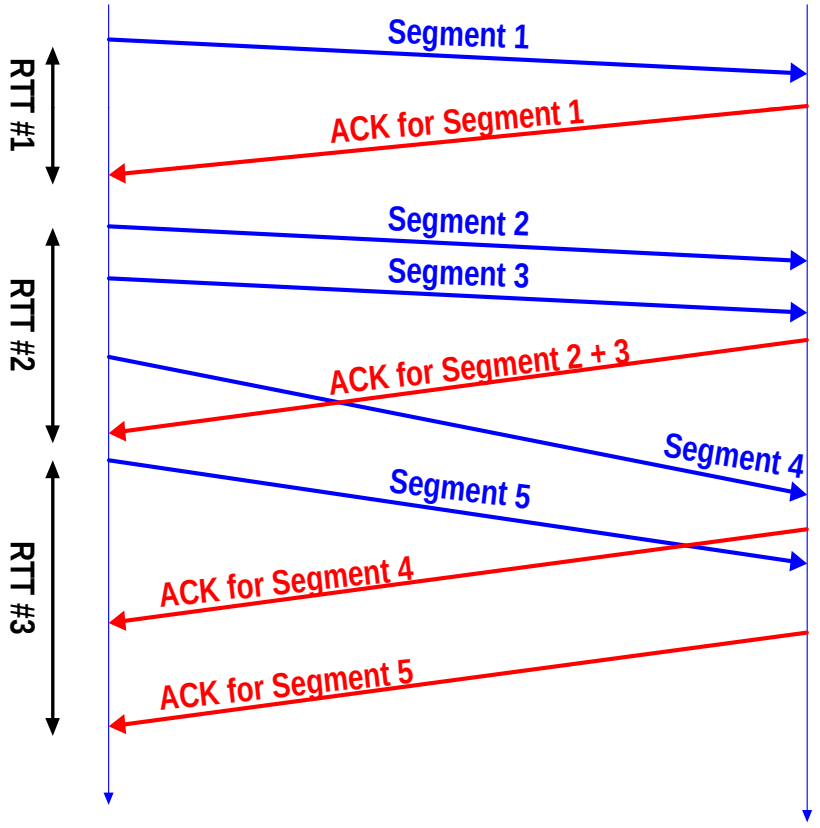


TCP: Διαχείριση χρονομετρητών

Ρύθμιση της τιμής του Timeout

Η τιμή του RTO τίθεται βάσει των μετρήσεων του RTT που πραγματοποιεί το TCP

- Κάθε σύνδεση TCP μετράει τη χρονική διαφορά μεταξύ της αποστολής ενός τεμαχίου και της λήψης της αντίστοιχης ACK
- Υπάρχει μόνο μια μέτρηση σε ισχύ κάθε φορά (δηλ., οι μετρήσεις δεν επικαλύπτονται)
- Στο διπλανό σχήμα φαίνονται τρεις μετρήσεις RTT



TCP: Διαχείριση χρονομετρητών

Ρύθμιση της τιμής του Timeout

- Η RTO υπολογίζεται βάσει των μετρήσεων του *RTT*
  - Χρησιμοποιείται εκθετικός σταθμισμένος κινούμενος μέσος όρος (*srtt*) για την εκτιμώμενη καθυστέρηση και τη variance (*rttvar*) της καθυστέρησης

- Οι μετρήσεις RTT εξομαλύνονται ως εξής:

$$srtt_{n+1} = \alpha \ SampleRTT + (1-\alpha)\, srtt_n$$

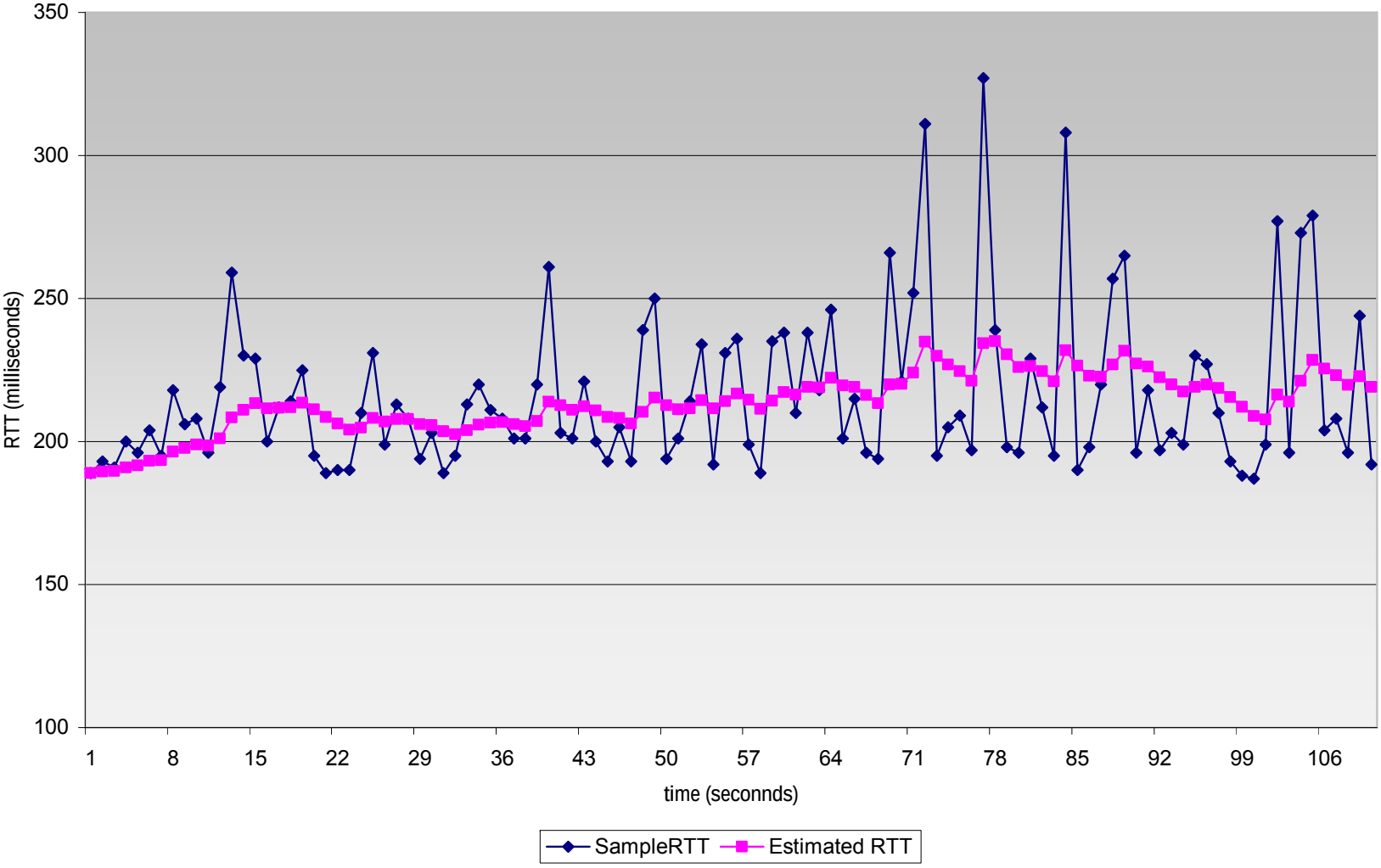
$$rttvar_{n+1} = \beta\,(\,|\,SampleRTT - srtt_n\,|\,) + (1-\beta)\, rttvar_n$$

$$RTO_{n+1} = srtt_{n+1} + 4\, rttvar_{n+1}$$

  - Οι τιμές του $\alpha = 1/8$ και $\beta = 1/4$

TCP: Διαχείριση χρονομετρητών

Ρύθμιση της τιμής του Timeout



TCP: Διαχείριση χρονομετρητών

Ρύθμιση της τιμής του Timeout

- Αρχική τιμή του RTO
  - Ο πομπός θέτει την αρχική τιμή του RTO:

$$RTO_0 = 3\ sec$$

- Υπολογισμός του RTO μετά την πρώτη μέτρηση του RTT

$$srtt_1 = SampleRTT$$

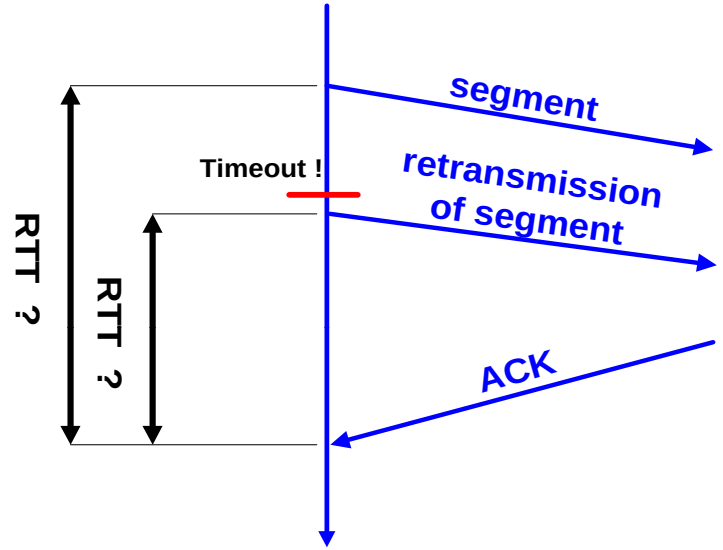
$$rttvar_1 = SampleRTT / 2$$

$$RTO_1 = srtt_1 + 4\ rttvar_{n+1}$$

TCP: Διαχείριση χρονομετρητών

Ρύθμιση της τιμής του Timeout

Αν ληφθεί ACK για τεμάχιο που επαναμεταδόθηκε, ο πομπός δεν μπορεί να ξέρει αν η ACK ανήκει στο αρχικό ή στο τεμάχιο που επαναμεταδόθηκε.



- **Αλγόριθμος του Karn**:
  - Μην ενημερώνεις την *RTT* για τεμάχια που επαναμεταδόθηκαν
  - Ξαναξεκίνα τις μετρήσεις RTT μόνο μετά τη λήψη ACK που αφορά κανονικό τεμάχιο
  - Όταν εμφανιστεί ένα timeout, η τιμή του RTO διπλασιάζεται (*εκθετική οπισθοχώρηση*)

$$RTO_{n+1} = min\,(\,2\,RTO_n\,,\,64)\ seconds$$

TCP: Διαχείριση συνδέσεων

- Εγκατάσταση σύνδεσης
- Απόλυση σύνδεσης
- Ειδικά σενάρια
- Διαγράμματα καταστάσεων

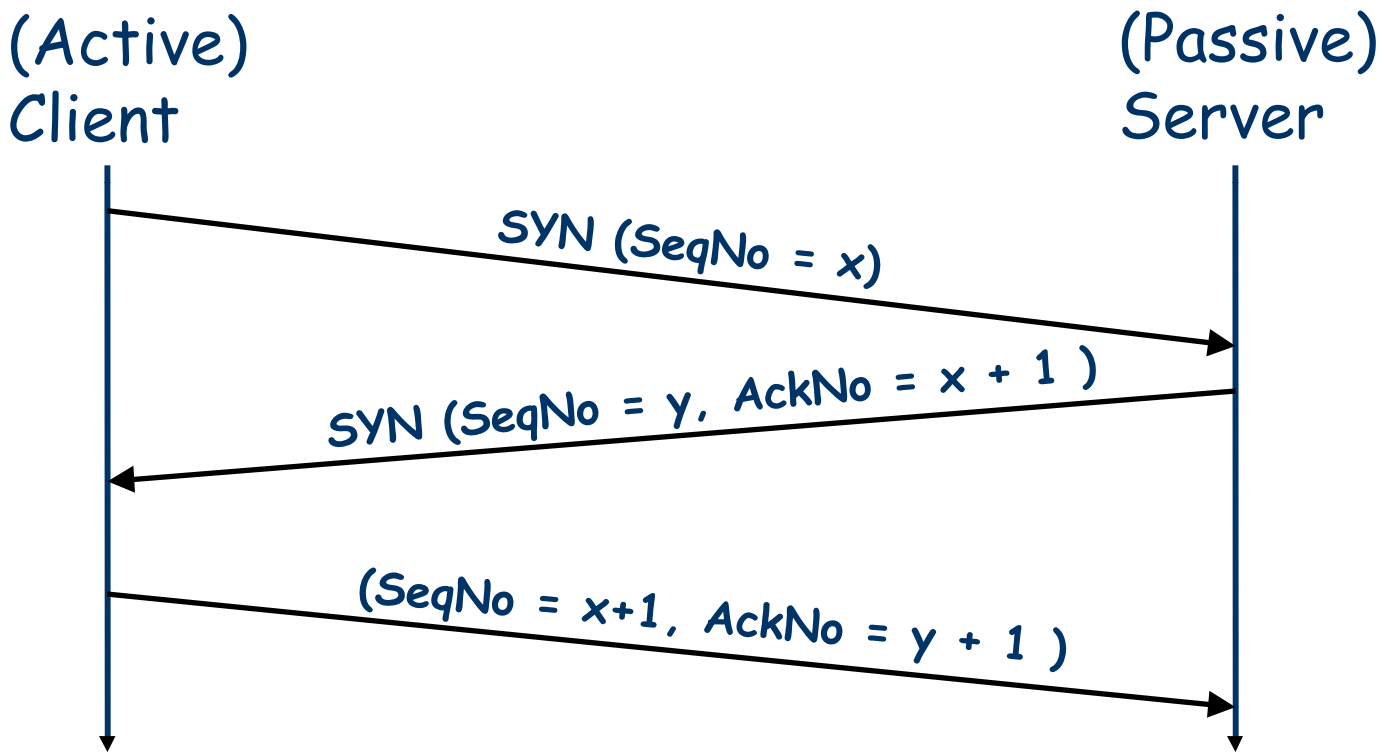
TCP: Διαχείριση συνδέσεων

Εγκατάσταση σύνδεσης

- Το TCP χρησιμοποιεί **τριμερή χειραψία** για να εγκαταστήσει μια σύνδεση:
- **Βήμα 1**: Ο client host στέλνει τεμάχιο TCP SYN με
  - το SYN bit ενεργοποιημένο
  - καθορίζει τον αρχικό αύξοντα αριθμό (ISN)
  - δεν στέλνει δεδομένα
- **Βήμα 2**: Ο server host λαμβάνει τεμάχιο SYN και απαντάει με τεμάχιο SYNACK
  - τα SYN και ACK bits ενεργοποιημένα
  - καθορίζει τον αρχικό αύξοντα αριθμό του
  - ACK για το ISN του client
- **Βήμα 3**: Ο client λαμβάνει SYNACK και απαντά με ACK.
  - ACK για το ISN του server. Το τεμάχιο μπορεί να περιέχει και δεδομένα

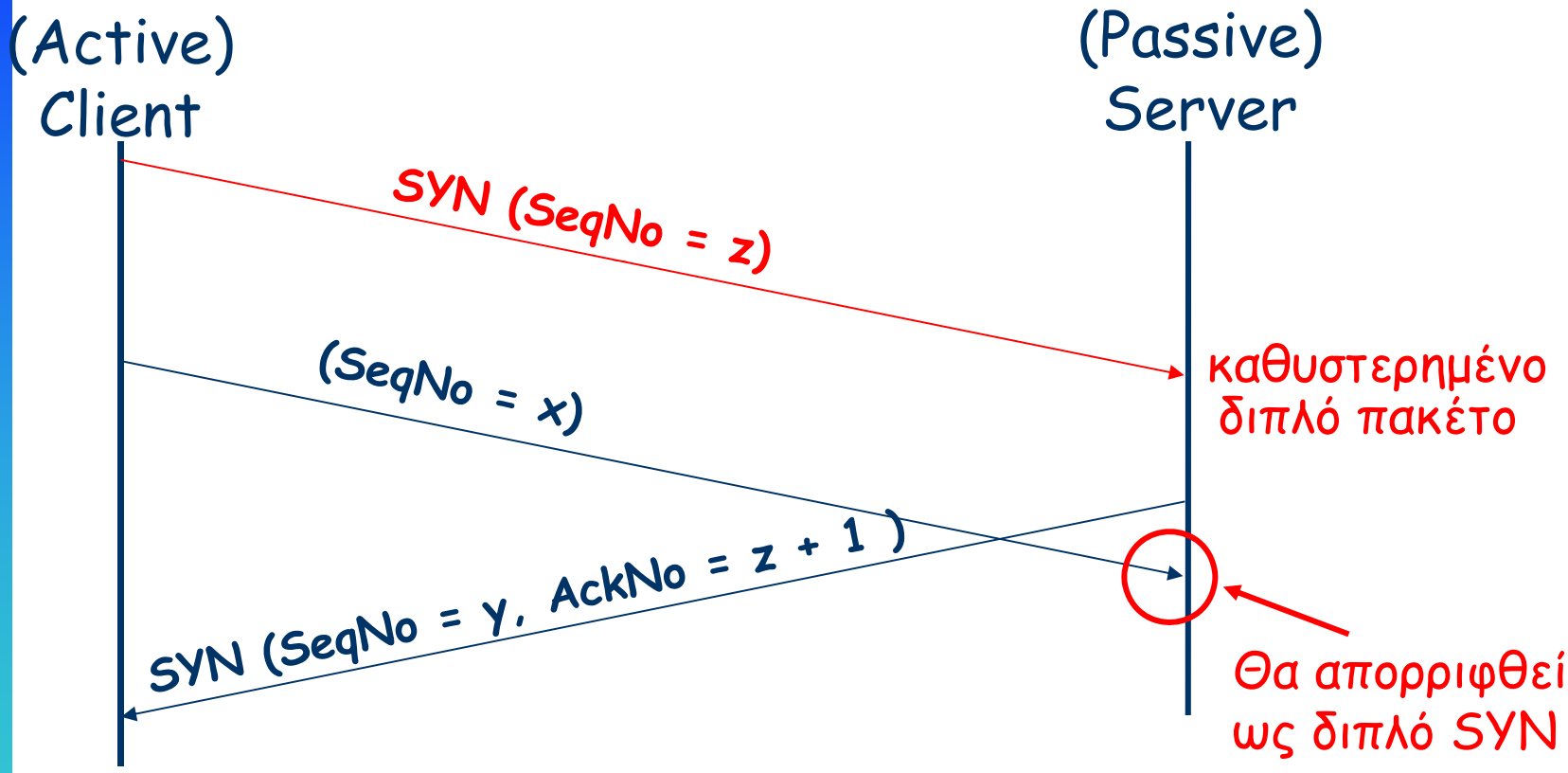
TCP: Διαχείριση συνδέσεων

Τριμερής χειραψία



TCP: Διαχείριση συνδέσεων

Γιατί δεν αρκεί η διμερής χειραψία

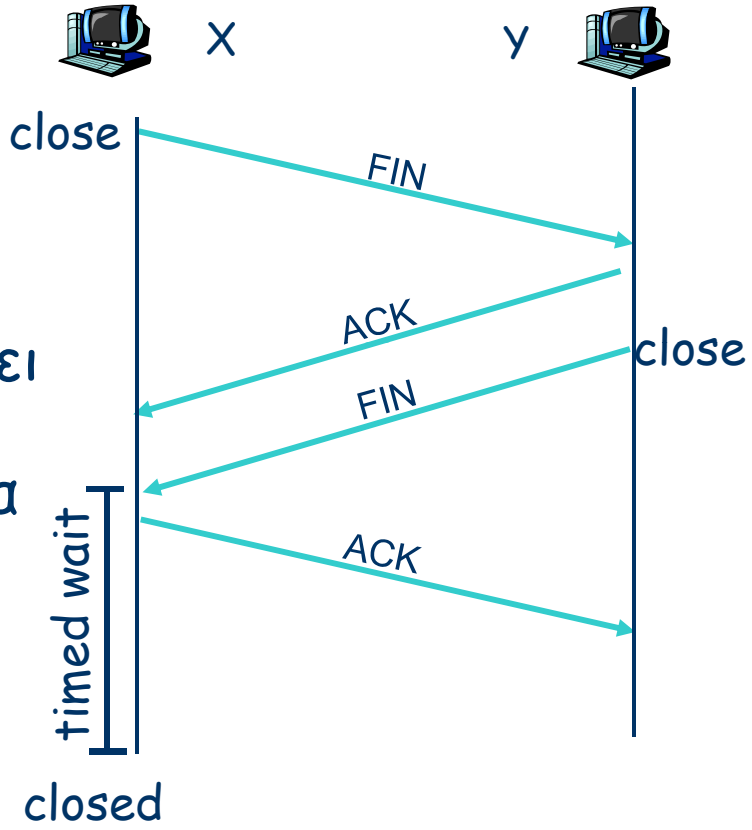


Όταν ο client αρχίζει τη μετάδοση δεδομένων (ξεκινώντας με SeqNo = x+1), ο server θα απορρίψει όλα τα δεδομένα.

TCP: Διαχείριση συνδέσεων

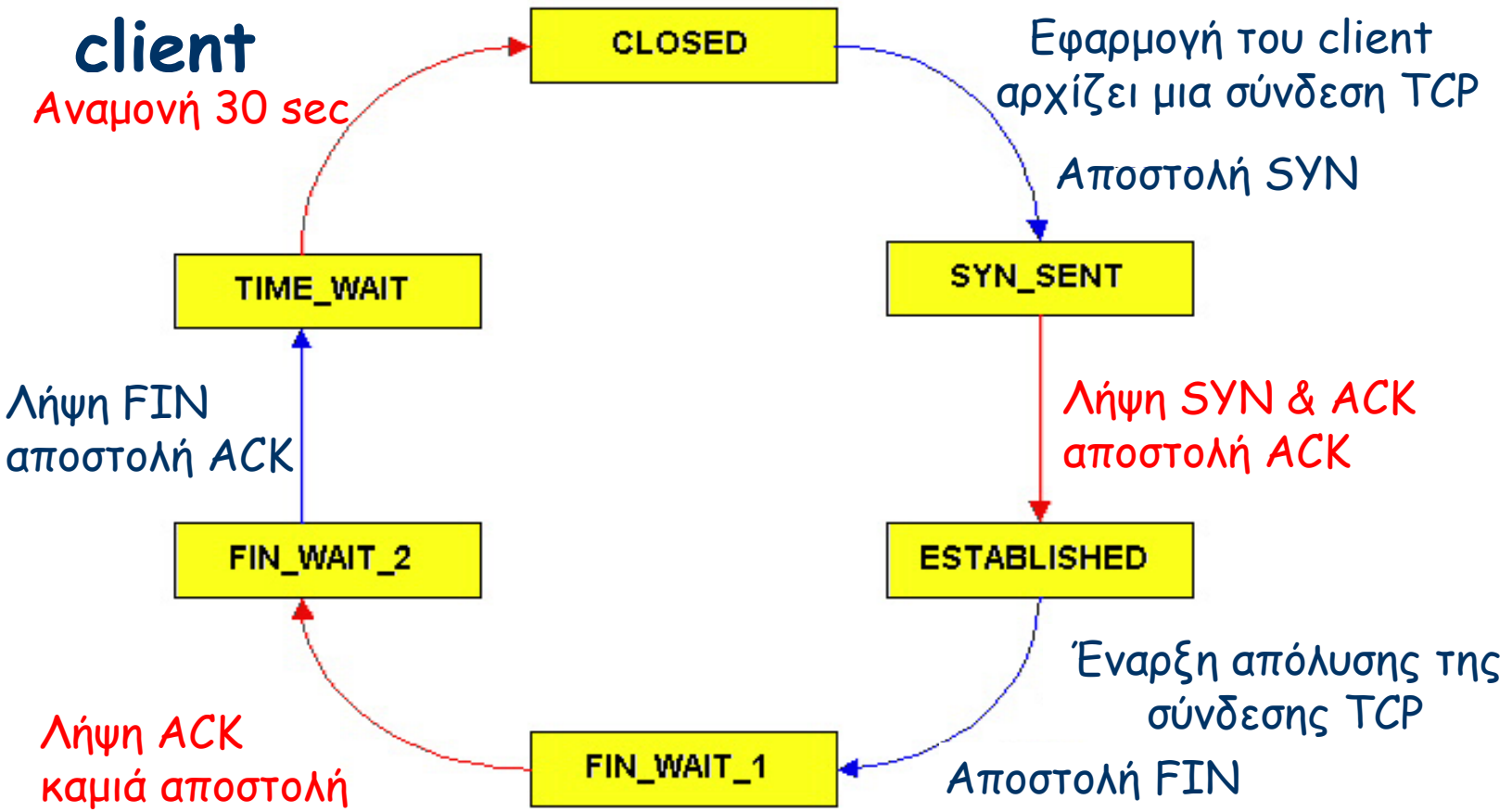
Απόλυση σύνδεσης

- Κάθε άκρο της ροής δεδομένων πρέπει να τερματίσει ανεξάρτητα **("half-close")**
- Αν το ένα άκρο τερματίσει, στέλνει ένα τεμάχιο FIN. Τούτο σημαίνει ότι δεν θα σταλούν άλλα δεδομένα
- Απαιτούνται τέσσερα βήματα:
  - (1) Ο Χ στέλνει ένα FIN στον Υ **(active close)**
  - (2) Ο Υ επαληθεύει το FIN,
    (ταυτόχρονα: Ο Υ μπορεί να στείλει ακόμα δεδομένα στον Χ)
  - (3) και ο Υ στέλνει ένα FIN στον Χ **(passive close)**
  - (4) Ο Χ επαληθεύει το FIN.



TCP: Διαχείριση συνδέσεων

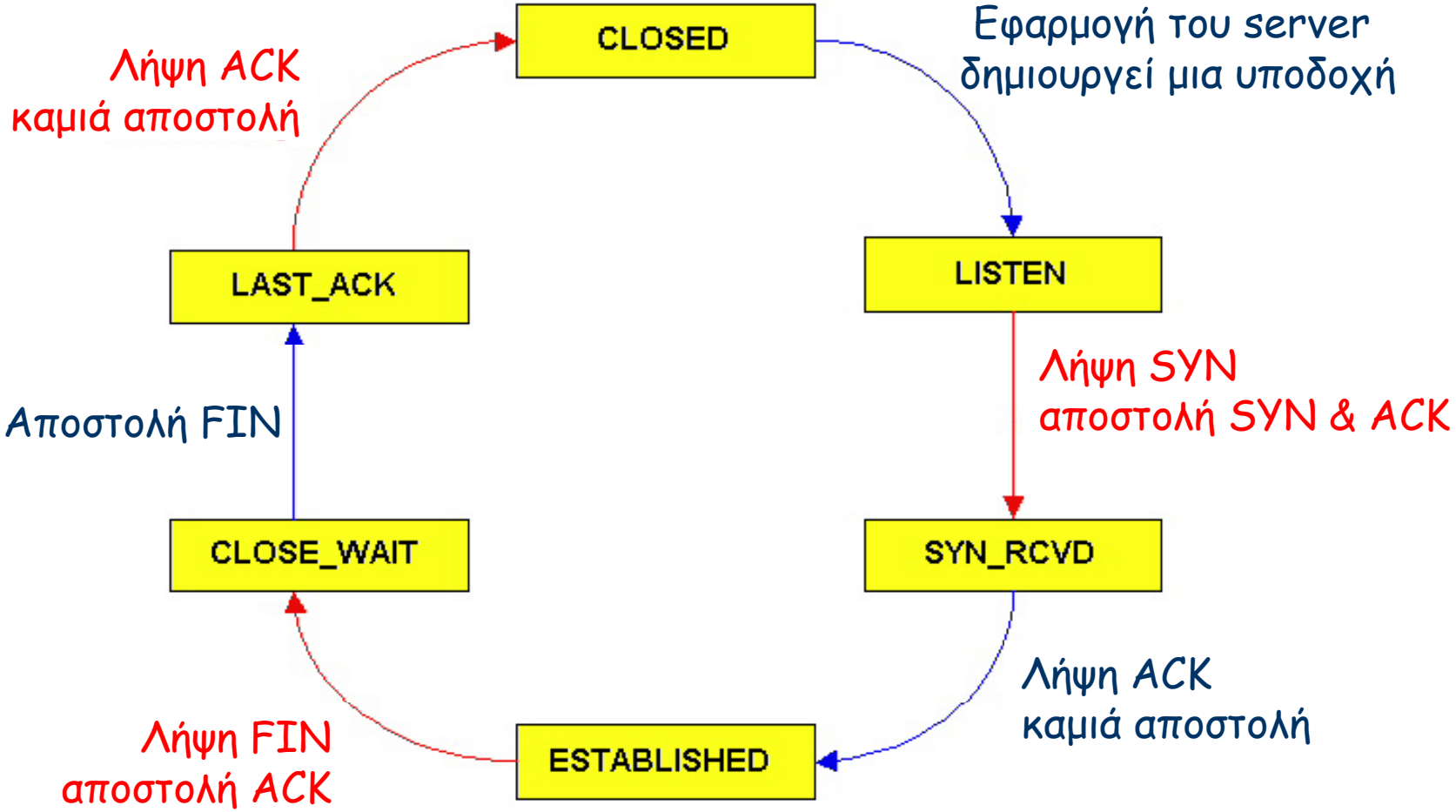
Διάγραμμα μετάβασης καταστάσεων



TCP: Διαχείριση συνδέσεων

Διάγραμμα μετάβασης καταστάσεων

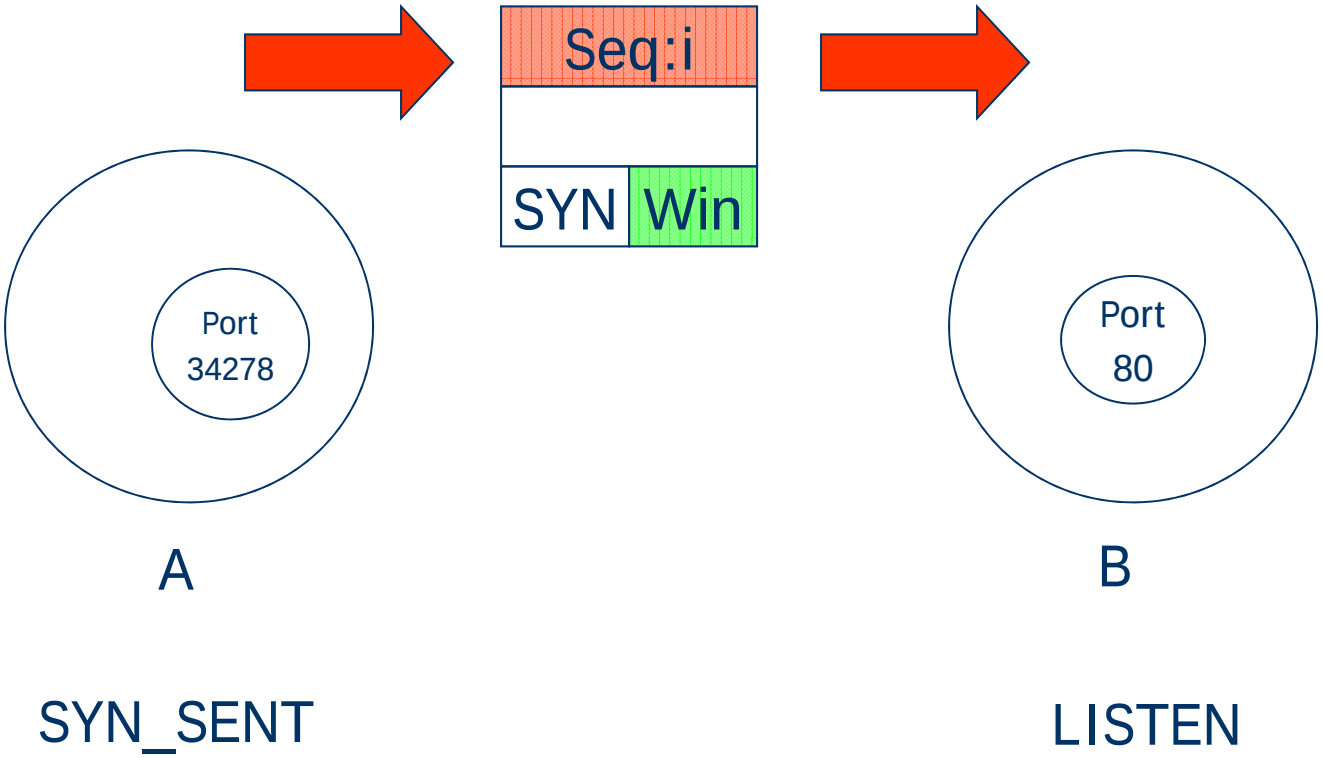
TCP server



TCP: Διαχείριση συνδέσεων

Παράδειγμα εγκατάστασης σύνδεσης

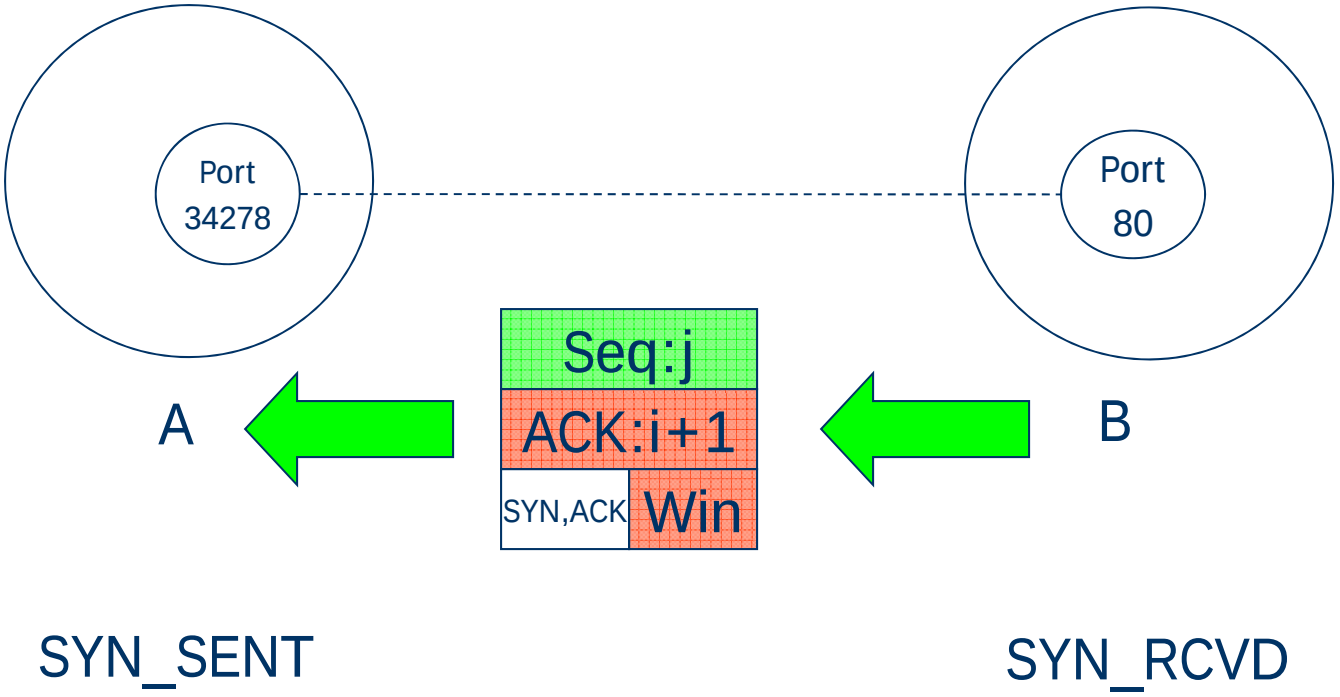
- Εγκατάσταση σύνδεσης: Βήμα 1



TCP: Διαχείριση συνδέσεων

Παράδειγμα εγκατάστασης σύνδεσης

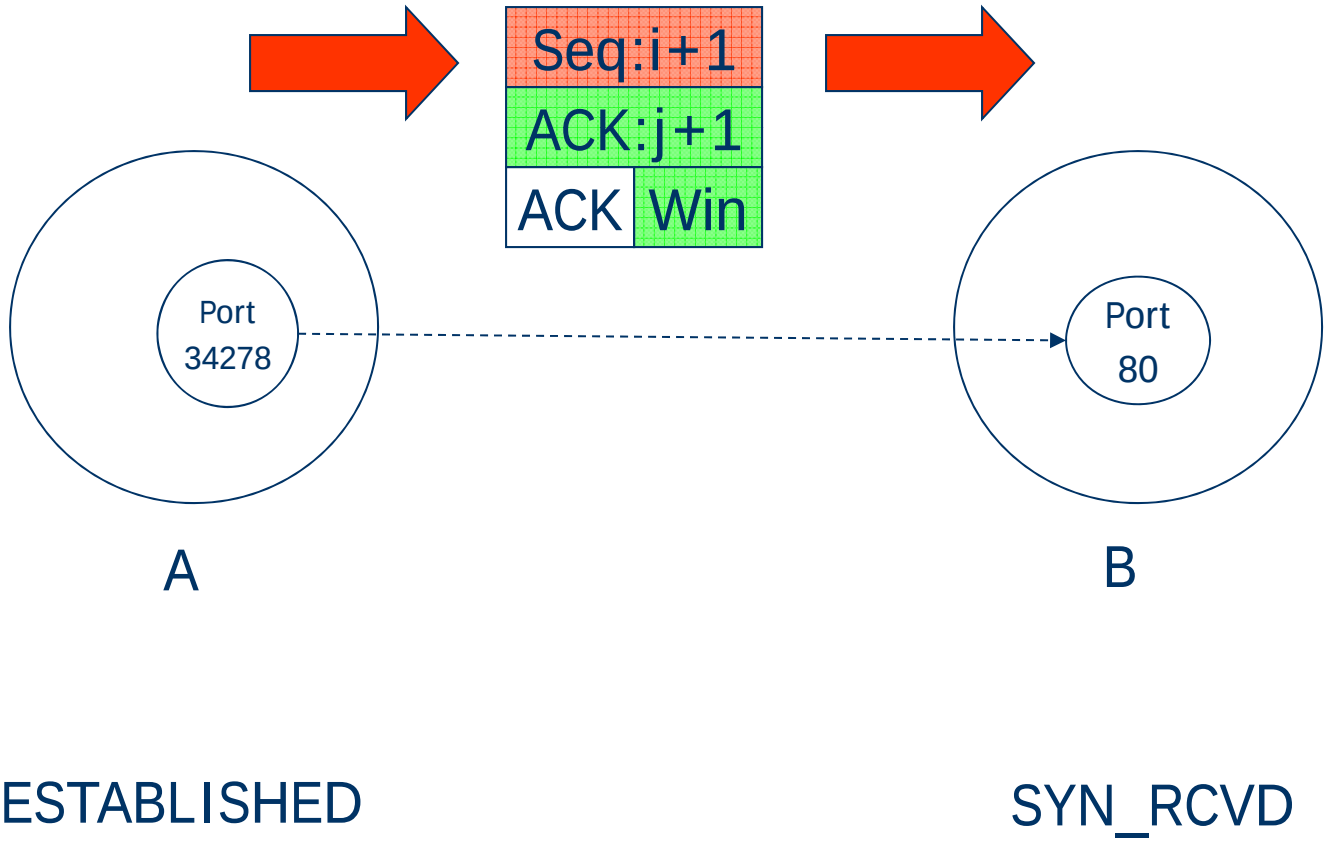
- Εγκατάσταση σύνδεσης: Βήμα 2



TCP: Διαχείριση συνδέσεων

Παράδειγμα εγκατάστασης σύνδεσης

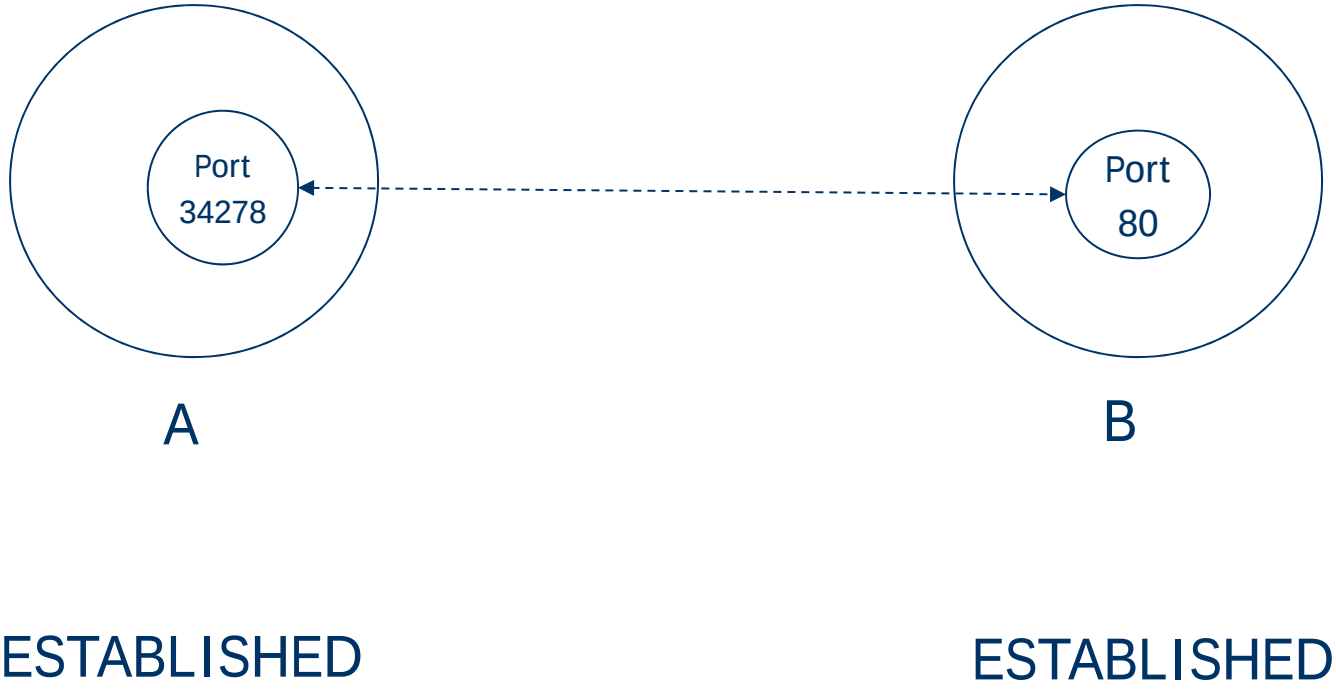
- Εγκατάσταση σύνδεσης: Βήμα 3



TCP: Διαχείριση συνδέσεων

Παράδειγμα εγκατάστασης σύνδεσης

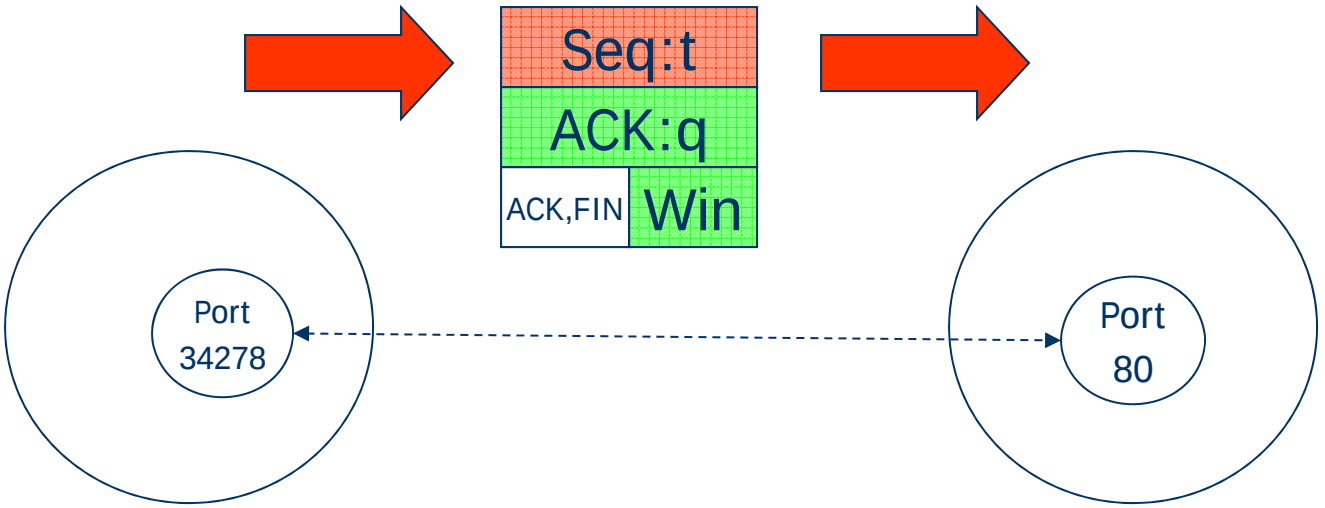
- Και οι δύο στην κατάσταση ESTABLISHED, ανταλλαγή δεδομένων...



TCP: Διαχείριση συνδέσεων

Παράδειγμα απόλυσης σύνδεσης

- Απόλυση σύνδεσης: Βήμα 1



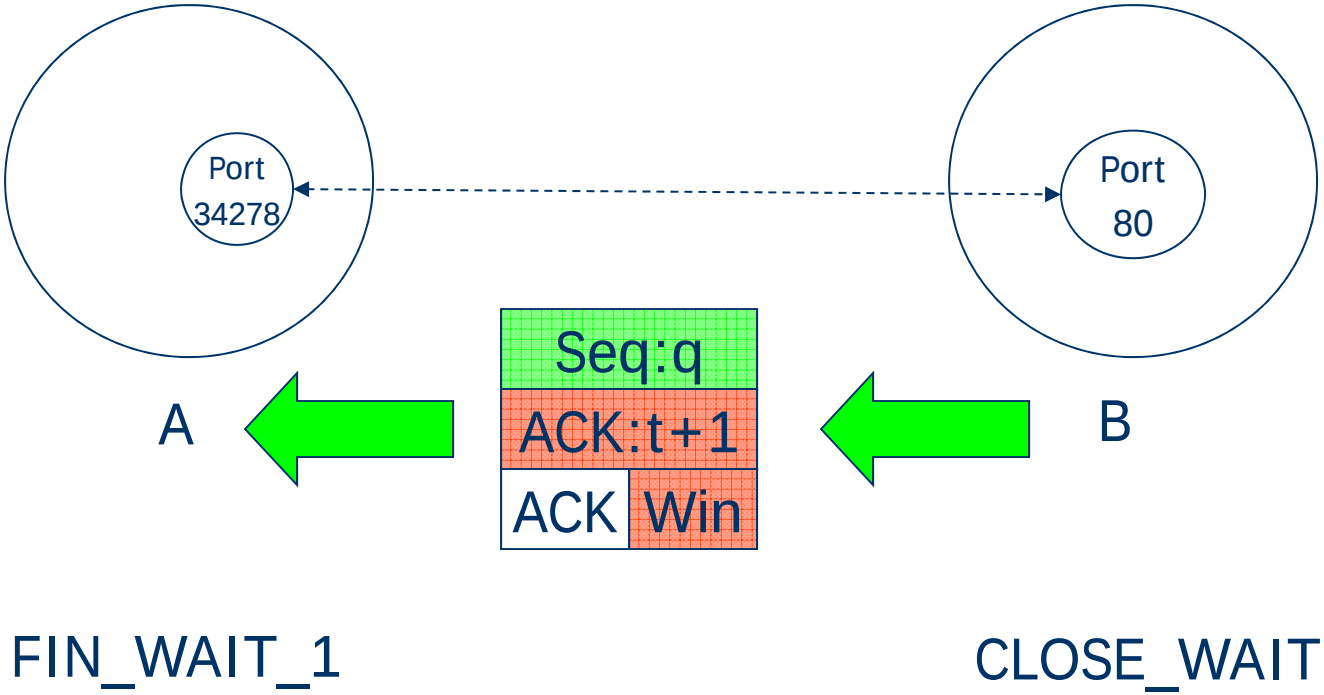
FIN_WAIT_1

ESTABLISHED

TCP: Διαχείριση συνδέσεων

Παράδειγμα απόλυσης σύνδεσης

- Απόλυση σύνδεσης: Βήμα 2



TCP: Διαχείριση συνδέσεων

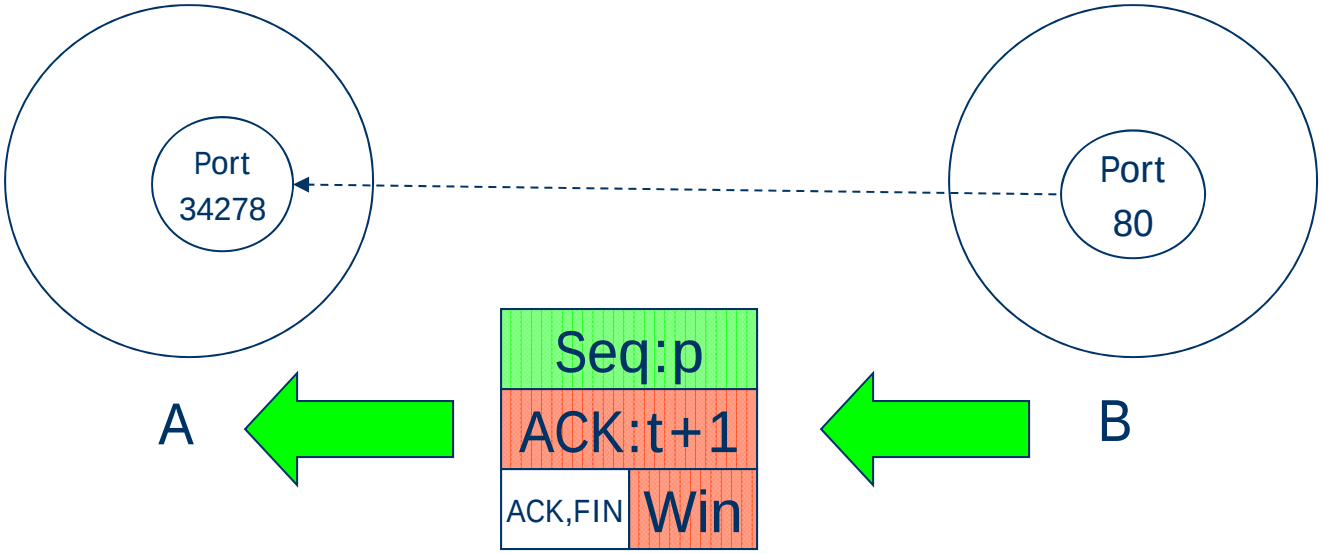
Παράδειγμα απόλυσης σύνδεσης

- Η εφαρμογή στον B πρέπει να κλείσει

TCP: Διαχείριση συνδέσεων

Παράδειγμα απόλυσης σύνδεσης

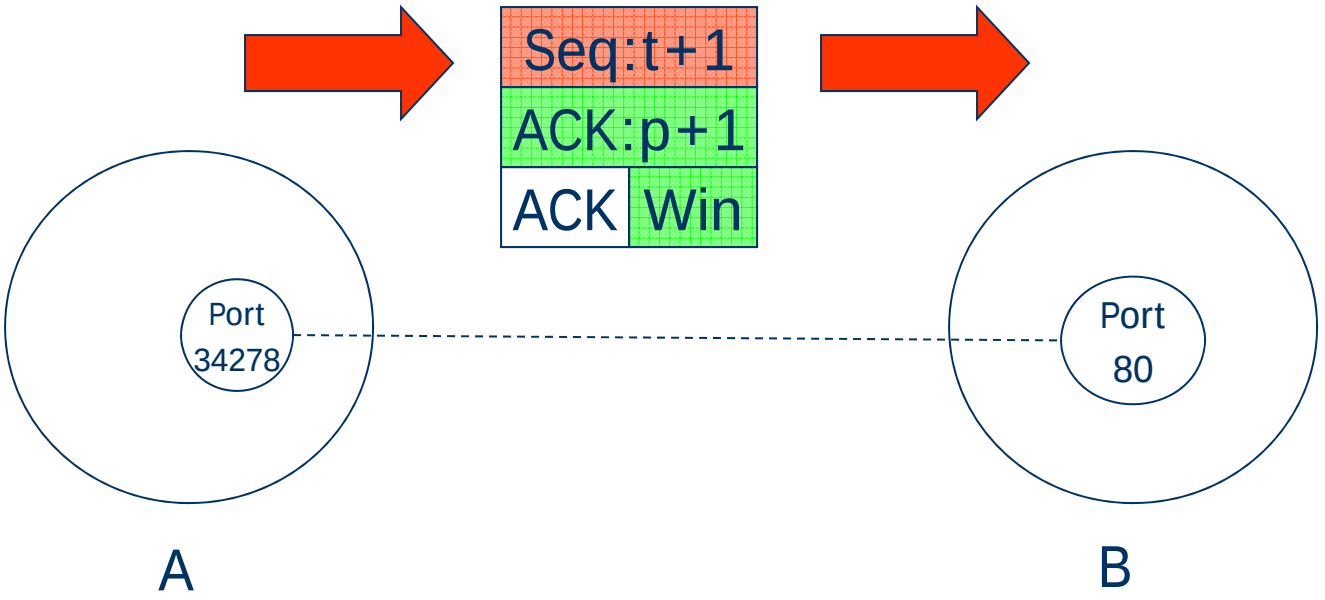
- Απόλυση σύνδεσης: Βήμα 3



TCP: Διαχείριση συνδέσεων

Παράδειγμα απόλυσης σύνδεσης

- Απόλυση σύνδεσης: Βήμα 4



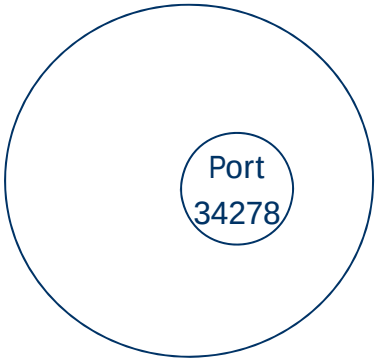
TIME_WAIT

LAST_ACK

TCP: Διαχείριση συνδέσεων

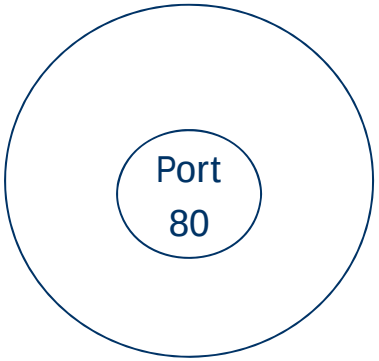
Παράδειγμα απόλυσης σύνδεσης

- Ο Client περιμένει 2*MSL πριν μεταβεί στην κατάσταση CLOSED (ο Β μπορεί να ξαναστείλει ένα FIN)



A

TIME_WAIT



B

CLOSED

TCP: Διαχείριση συνδέσεων

Κατάσταση αναμονής 2MSL = TIME_WAIT

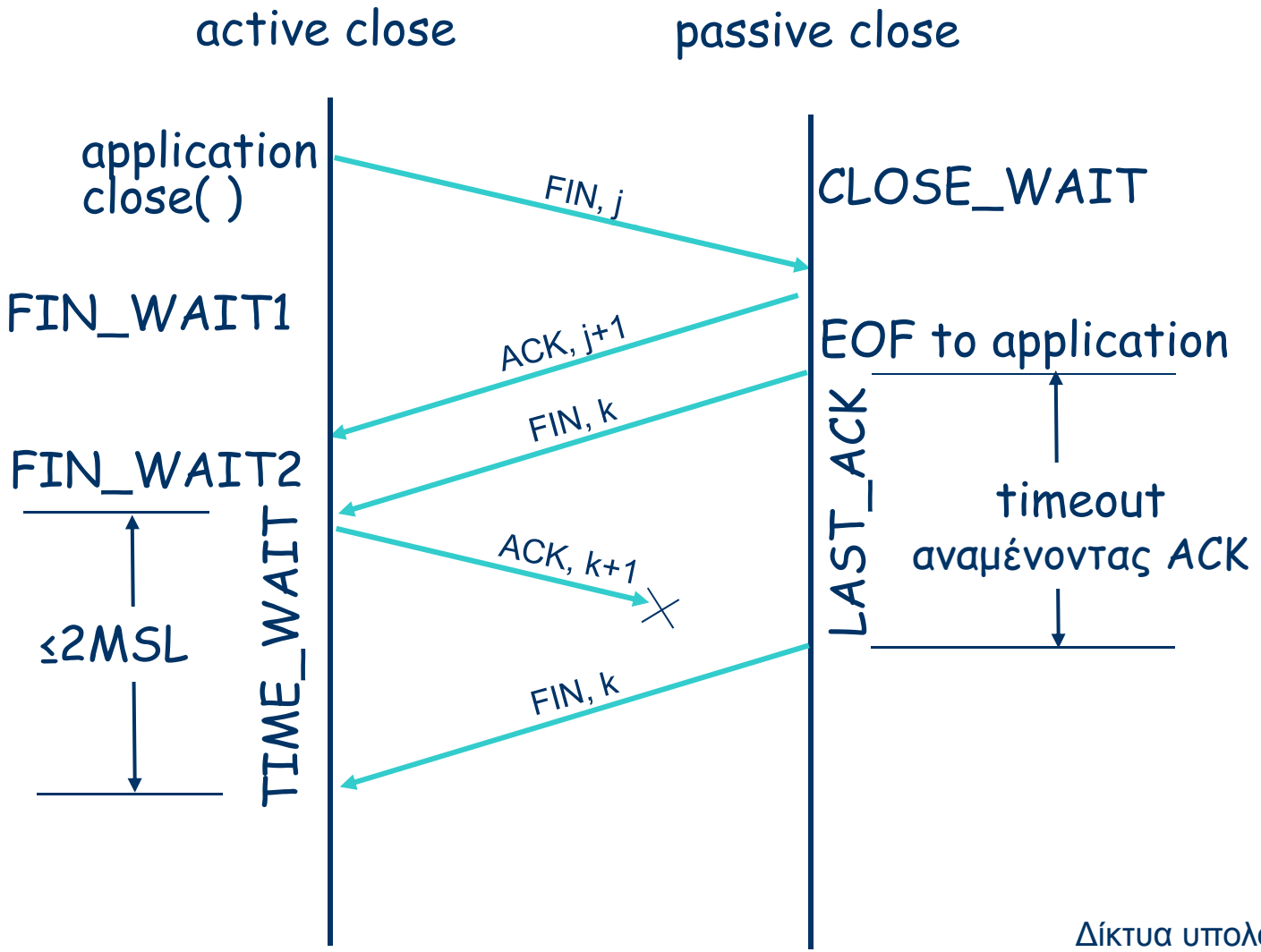
- Όταν το TCP κάνει active close, και στέλνει την τελική ACK, η σύνδεση πρέπει να παραμείνει στην κατάσταση **TIME_WAIT** για διπλάσιο χρόνο από τη μέγιστη ζωή τεμαχίων (maximum segment lifetime, MSL).

  2MSL= 2 * Maximum Segment Lifetime

- Γιατί;
  Δίνεται στον TCP client μια ευκαιρία επαναποστολής της τελικής ACK. (Ο server θα κάνει timeout αφού στείλει το τεμάχιο FIN και ξαναστείλει το FIN)
- Το MSL τίθεται στα 2 min ή 1 min ή 30 sec.

TCP: Διαχείριση συνδέσεων

TIME_WAIT



TCP: Διαχείριση συνδέσεων

Επαναφορά συνδέσεων

- Η επαναφορά (reset) συνδέσεων γίνεται με την ενεργοποίηση της σημαίας RST
- **Πότε ενεργοποιείται η σημαία RST;**
  - Όταν φθάνει η αίτηση σύνδεσης και δεν αναμένει καμία διαδικασία server στο σημείο προορισμού
  - Όταν αποσταλεί τεμάχιο που δεν αναμένεται καθόλου σε υπάρχουσα σύνδεση, π.χ. ο αύξων αριθμός είναι εκτός περιοχής
- Η επαναφορά μιας σύνδεσης αναγκάζει τον δέκτη του RST να πετάξει τα αποθηκευμένα δεδομένα. Ο δέκτης δεν επαληθεύει το τεμάχιο RST

TCP: Μεταφορά δεδομένων

Η μεταφορά δεδομένων μέσω TCP μπορεί να χαρακτηριστεί ως:

αλληλοδραστική μεταφορά - telnet, rlogin
μαζική μεταφορά - ftp, mail, http

Το TCP έχει ευριστικές μεθόδους για να χειρίζεται αυτούς τους τύπους μεταφοράς δεδομένων.

Για αλληλοδραστική μεταφορά δεδομένων:

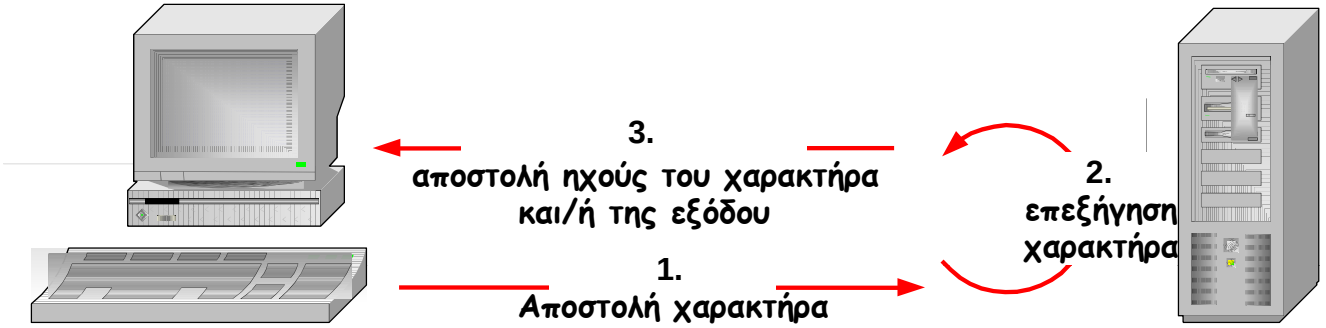
- Επιχειρεί να περιορίσει τον αριθμό των πακέτων

Για μαζική μεταφορά δεδομένων:

- Επιχειρεί να ελέγξει τη ροή δεδομένων

TCP: Μεταφορά δεδομένων

Αλληλοδραστικές εφαρμογές: Telnet



Οι εφαρμογές απομακρυσμένου τερματικού (π.χ., Telnet) στέλνουν χαρακτήρες σε έναν server. Ο server επεξεργάζεται τον χαρακτήρα και στέλνει την έξοδό του στον client.

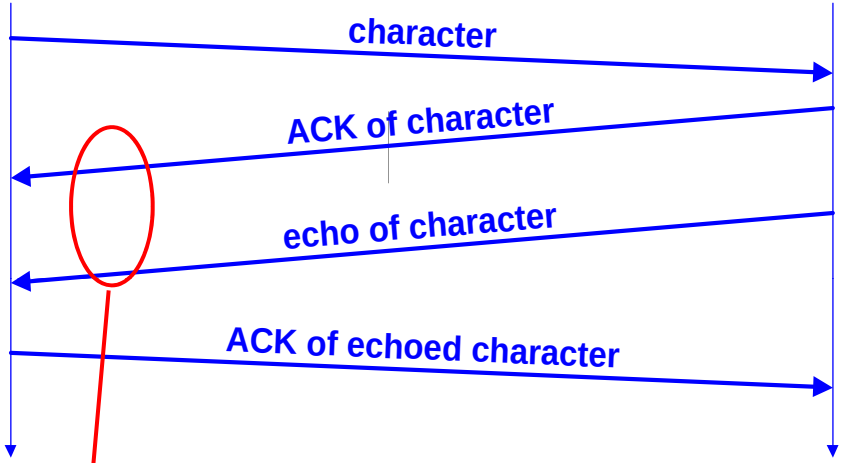
Για κάθε πληκτρολογούμενο χαρακτήρα, βλέπουμε 3 πακέτα:

1. **Client → Server:** Αποστολή πληκτρολογούμενου χαρακτήρα
2. **Server → Client:** Ηχώ του χαρακτήρα και επαλήθευση του πρώτου πακέτου
3. **Client → Server:** Επαλήθευση του δεύτερου πακέτου

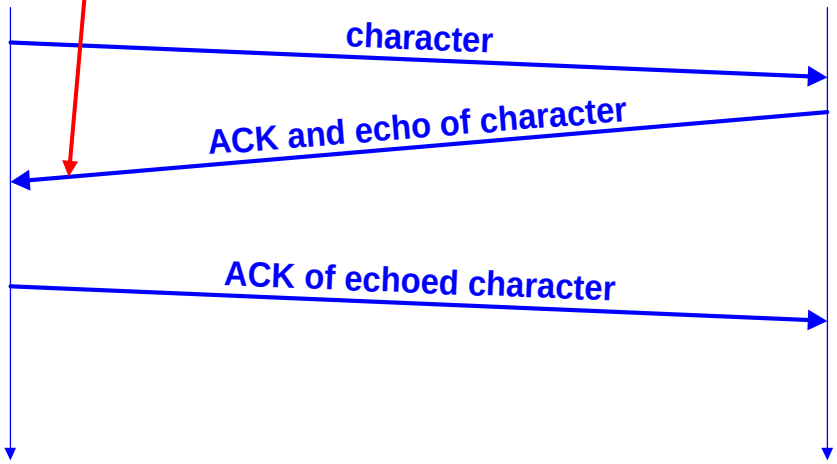
TCP: Μεταφορά δεδομένων

Αλληλοδραστικές εφαρμογές: Telnet

- Θα αναμέναμε 4 πακέτα ανά χαρακτήρα:



- Ωστόσο, εμφανίζεται η παραπλεύρως απεικονιζόμενη μορφή:

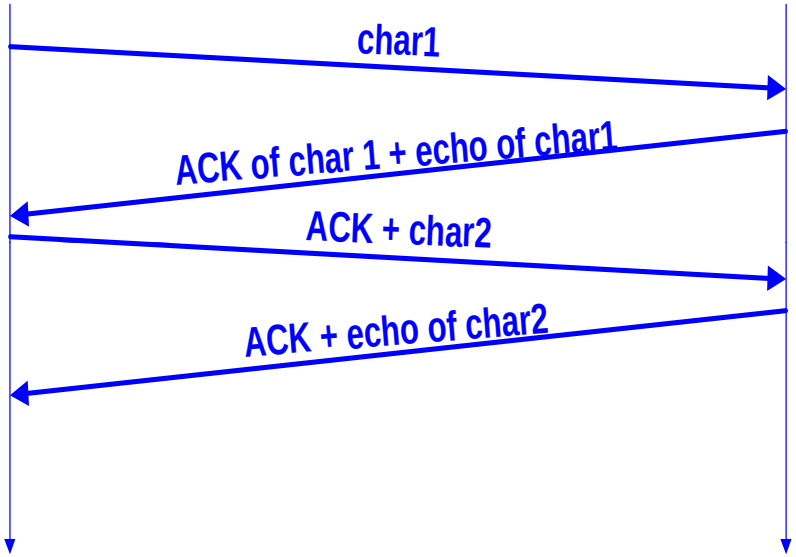


- Το TCP έχει καθυστερήσει την αποστολή μιας ACK

TCP: Μεταφορά δεδομένων

Σύνοδος telnet σε απομακρυσμένον host

- **Παρατήρηση:** Η μετάδοση των τεμαχίων ακολουθεί διαφορετικό τρόπο, δηλ., υπάρχουν μόνο δύο πακέτα ανά πληκτρολογούμενο χαρακτήρα



- Η καθυστερημένη επαλήθευση δεν εμφανίζεται
- Ο λόγος είναι ότι υπάρχουν πάντα δεδομένα έτοιμα προς αποστολή, όταν φθάνει η ACK

TCP: Μεταφορά δεδομένων

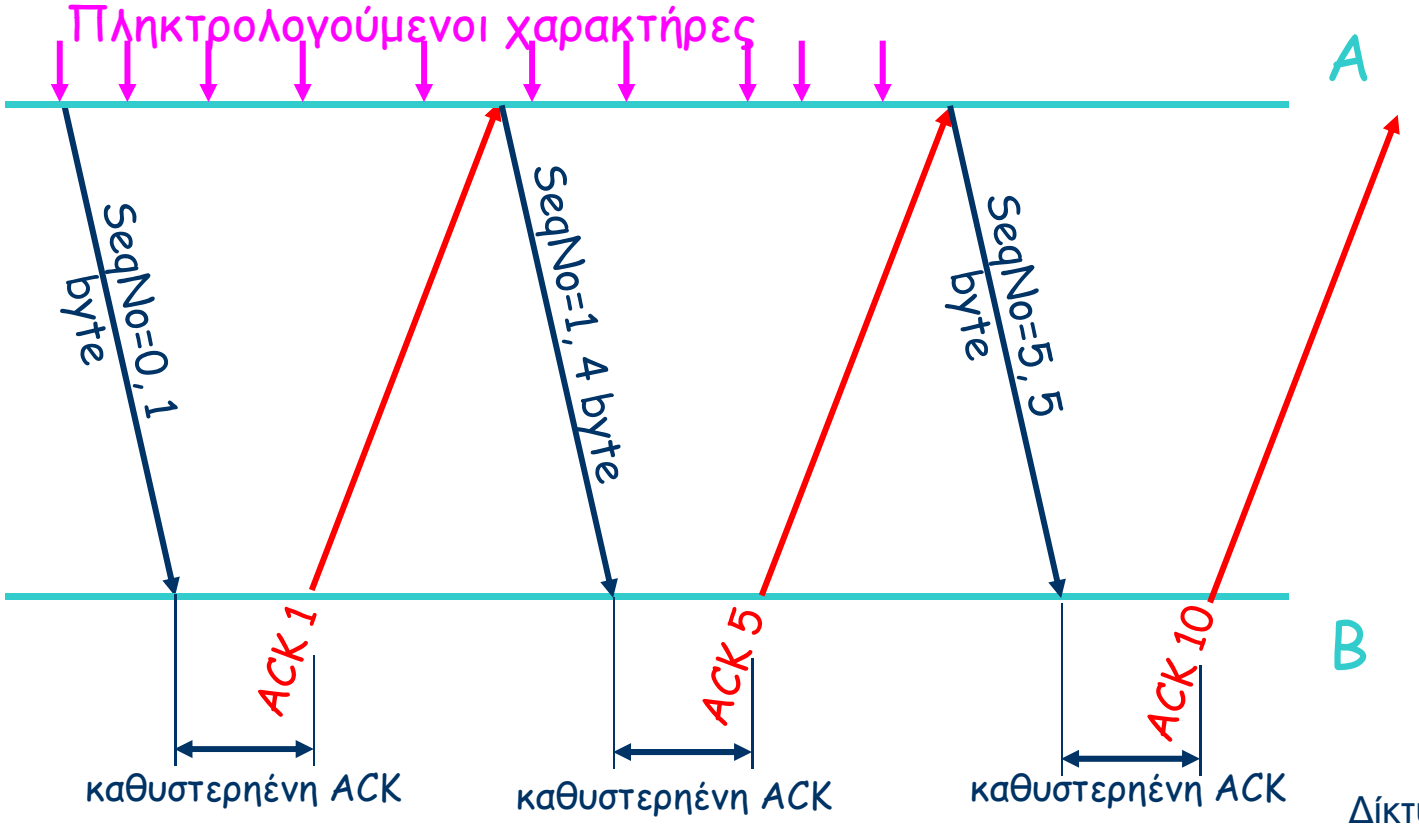
Αλγόριθμος του Nagle

- Έχει ως στόχο την αποφυγή της μη αποτελεσματικής χρήσης του εύρους ζώνης
- Πομπός:
  - αποθηκεύει προσωρινά όλα τα δεδομένα χρήστη, αν εκκρεμούν ανεπαλήθευτα δεδομένα
  - στέλνει, όταν όλα έχουν επαληθευτεί ή έχει δεδομένα που συμπληρώνουν ένα τεμάχιο MSS
- Δέκτης:
  - δίνει εντολή αποστολής, μόνο όταν μπορεί να αυξήσει επαρκώς το παράθυρο λήψης

TCP: Μεταφορά δεδομένων

Αλγόριθμος του Nagle

- Μόνο ένα τεμάχιο ενός byte μπορεί να μεταδίδεται (Επειδή δεν υπάρχουν δεδομένα προς απόστολή από τον B προς A έχουμε καθυστερημένες ACK)



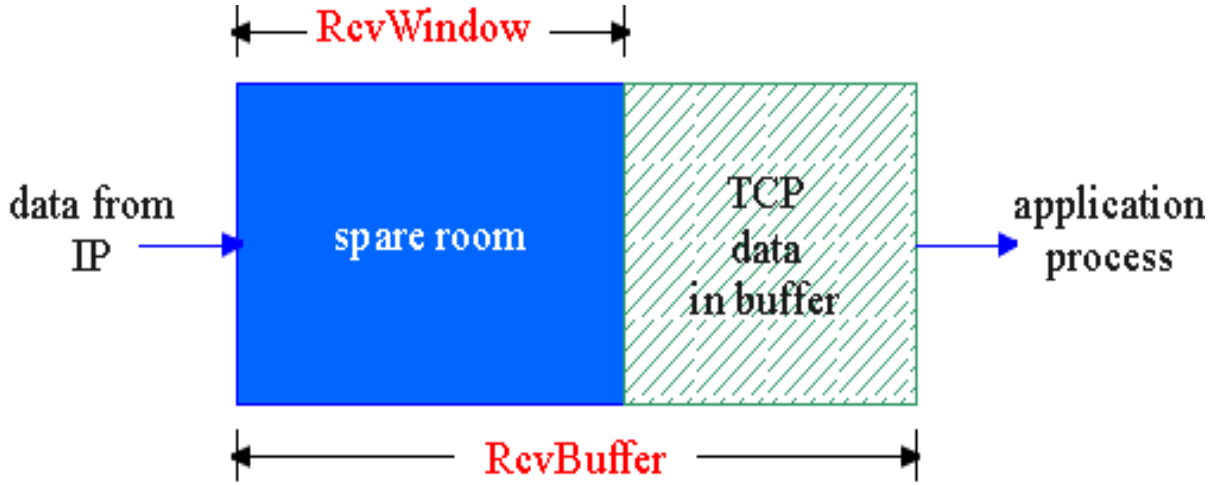
TCP: Μεταφορά δεδομένων

Ιδιότητες του αλγορίθμου του Nagle

- Εφαρμόζεται μόνο σε μικρά πακέτα. Στις μεταφορές μεγάλων αρχείων υπάρχουν πάντα πλήρη MSS για αποστολή
- Ο αλγόριθμος είναι αυτοχρονιζόμενος:
  - βασικά εφαρμόζει Stop & Wait για μικρά τεμάχια
  - σε LAN, το μικρό RTT δεν εισάγει μεγάλη αναμονή, οπότε ο αλγόριθμος δεν είναι αποτελεσματικός
  - σε WAN, το μεγάλο RTT εισάγει περισσότερη αναμονή, αλλά ο αλγόριθμος είναι πιο αποτελεσματικός σε μακριές ζεύξεις
- Όταν απαιτείται μικρή καθυστέρηση, ο αλγόριθμος προκαλεί ανεπιθύμητες καθυστερήσεις
- Οι εφαρμογές μπορεί να απενεργοποιήσουν τον αλγόριθμο

TCP: Έλεγχος ροής

- η πλευρά λήψης της σύνδεσης TCP έχει έναν καταχωρητή λήψης:



- η διαδικασία εφαρμογής μπορεί να αργεί να διαβάσει από τον καταχωρητή λήψης
- έλεγχος ροής: ο πομπός δεν πρέπει να υπερχειλίσει τον καταχωρητή του δέκτη μεταδίδοντας πολλά, πολύ γρήγορα
- προσαρμογή του ρυθμού αποστολής δεδομένων στον ρυθμό ανάγνωσης της λαμβάνουσας εφαρμογής

TCP: Έλεγχος ροής

- Το TCP χρησιμοποιεί έλεγχο ροής με ολισθαίνον παράθυρο:
  - δεν χρησιμοποιεί NACK
  - μόνο συσσωρευτικές ACK
- Οι επαληθεύσεις δεν προκαλούν αυτόματα αλλαγές στο μέγεθος παραθύρου του πομπού
- Ο δέκτης επιστρέφει δύο παραμέτρους στον πομπό

AckNo	window size (win)
32 bits	16 bits

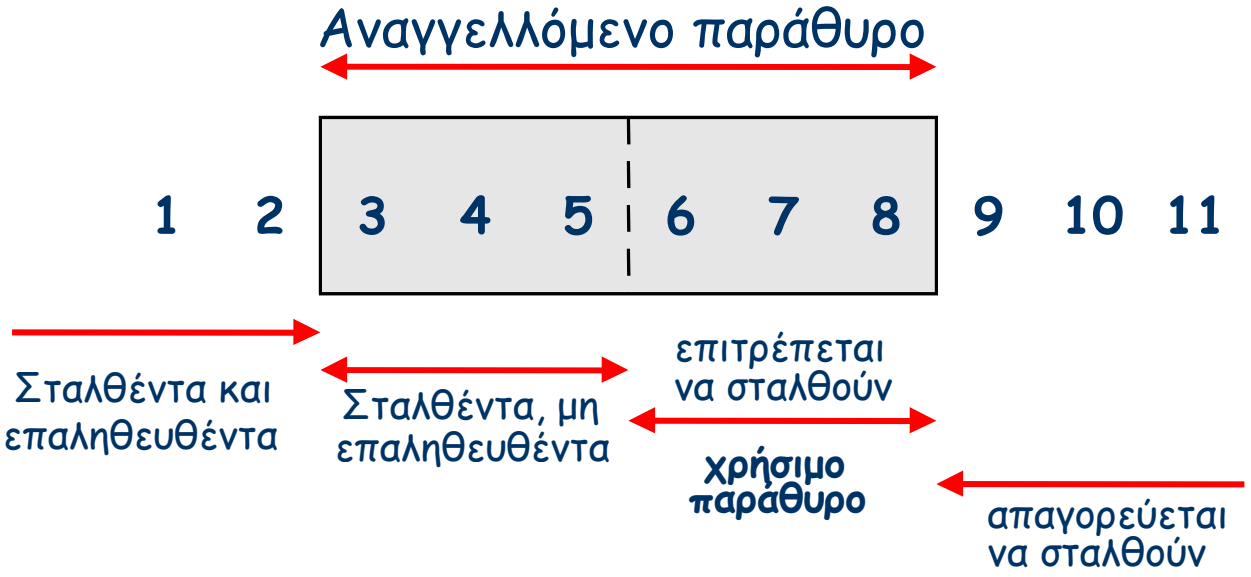
- Οπομπός μπορεί να στείλει δεδομένα μέχρι το διαφημιζόμενο παράθυρο, δηλαδή, τα byte

  AckNo, AckNo+1, …, AckNo + win -1
- Ο δέκτης μπορεί να επαληθεύσει χωρίς να αλλάξει το παράθυρο
- Ο δέκτης μπορεί να αλλάξει το παράθυρο χωρίς να επαληθεύσει

TCP: Έλεγχος ροής

Ολισθαίνον παράθυρο

Το πρωτόκολλο ολισθαίνοντος παραθύρου λειτουργεί σε επίπεδο byte:

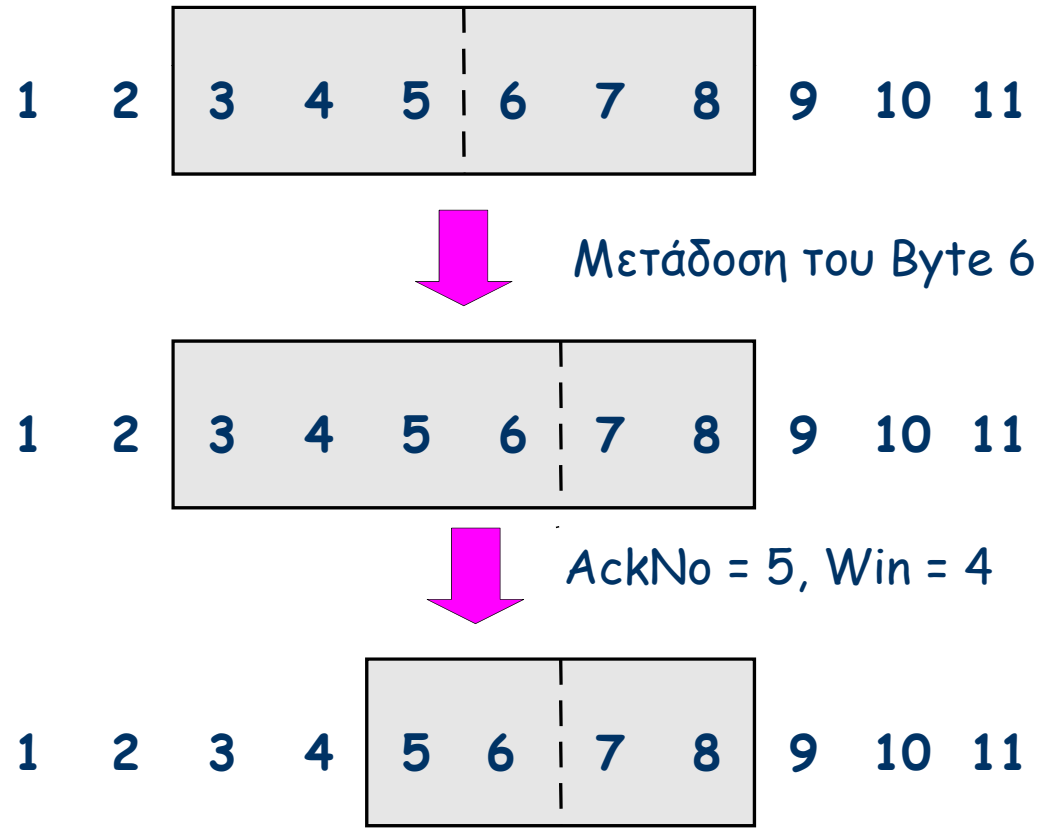


Ο πομπός μπορεί να στείλει μόνο τους αύξοντες αριθμούς 6,7,8.

TCP: Έλεγχος ροής

Ολισθαίνον παράθυρο: κλείσιμο

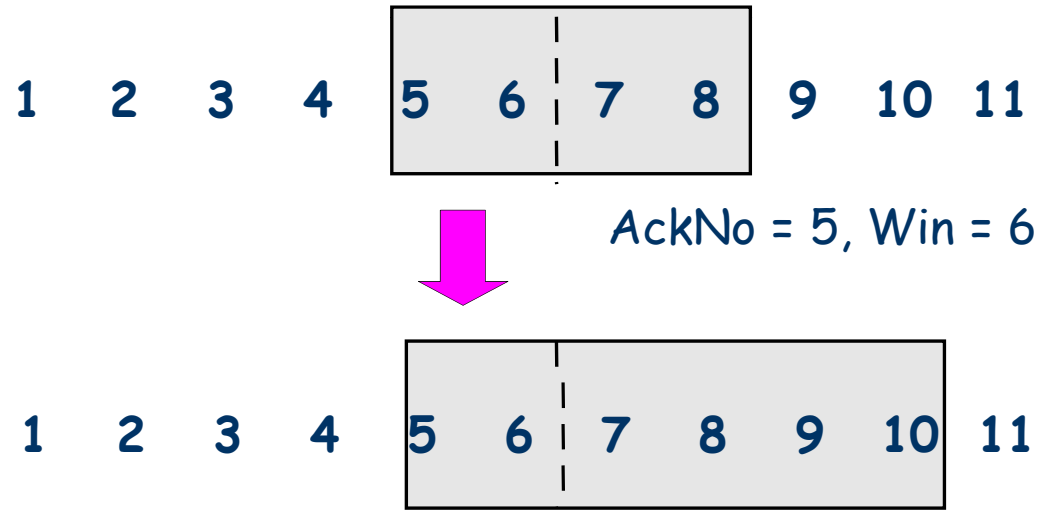
Αποστολή ενός byte (με SeqNo = 6) και λήψη της επαλήθευσης (AckNo = 5, Win=4):



TCP: Έλεγχος ροής

Ολισθαίνον παράθυρο: άνοιγμα

Λήψη επαλήθευσης που μεγαλώνει το παράθυρο προς τα δεξιά (AckNo = 5, Win=6):

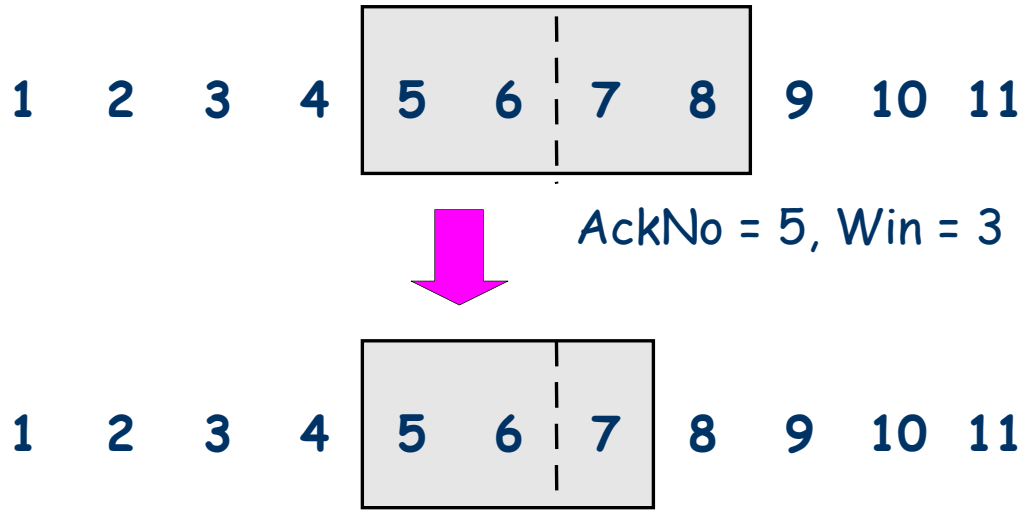


Ο δέκτης ανοίγει το παράθυρο όταν ο καταχωρητής TCP αδειάζει (εννοώντας ότι τα δεδομένα παραδίδονται στην εφαρμογή).

TCP: Έλεγχος ροής

Ολισθαίνον παράθυρο: συρρίκνωση

Λήψη επαλήθευσης που περιορίζει το παράθυρο από δεξιά (AckNo = 5, Win=3):



Η συρρίκνωση παραθύρου δεν πρέπει να χρησιμοποιείται

TCP: Έλεγχος ροής

Ολισθαίνον παράθυρο: παράδειγμα