Δίκτυα Υπολογιστών 27/11 9ο

**TCP round trip time, timeout**

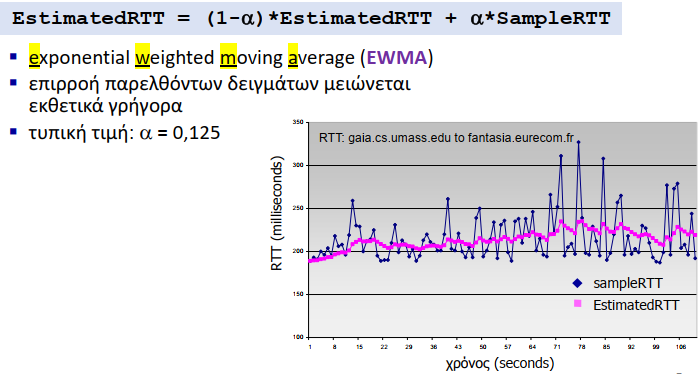
***Ε:***πώς ορίζουμε την τιμή TCP timeout;  
μεγαλύτερη από RTT αλλά RTT ποικίλει!  
▪ ***υπερβολικά μικρή:*** πρόωρο  
timeout, αχρείαστες επαναμεταδόσεις  
▪ ***υπερβολικά μεγάλη:***αργή  
αντίδραση σε απώλεια τμημάτων

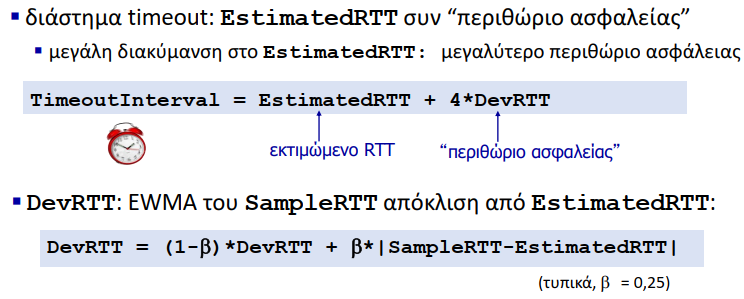
***Ε*:** πώς να εκτιμήσουμε το RTT;  
▪ SampleRTT: μετρηθείς χρόνος από την μετάδοση τμήματος έως λήψη ACK

• επαναμεταδόσεις δεν λαμβάνονται υπ’ όψιν

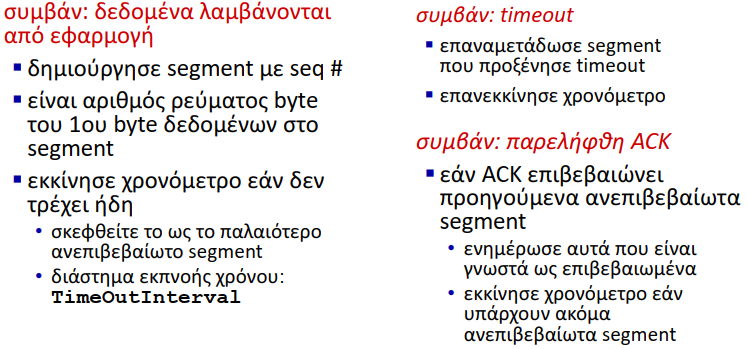
▪ SampleRTT: θα ποικίλει, θέλουμε εκτιμώμενο RTT να είναι “πιο ομαλό”

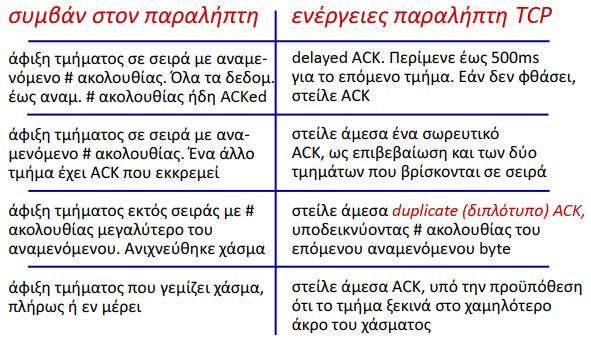
• υπολογίζεται ο Μέσος Όρος αρκετών *προσφάτων* μετρήσεων, όχι μόνον του τρέχοντος SampleRTT

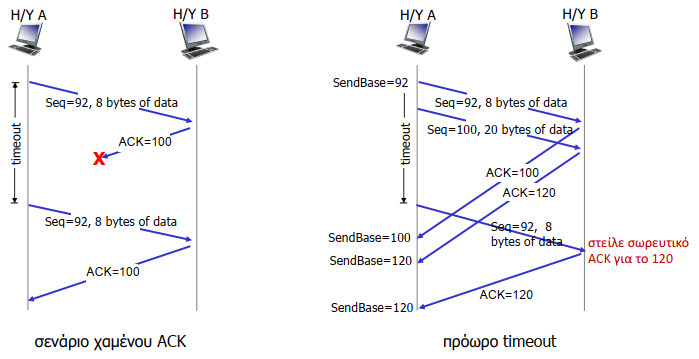


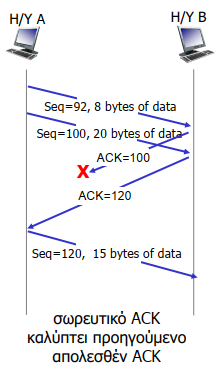


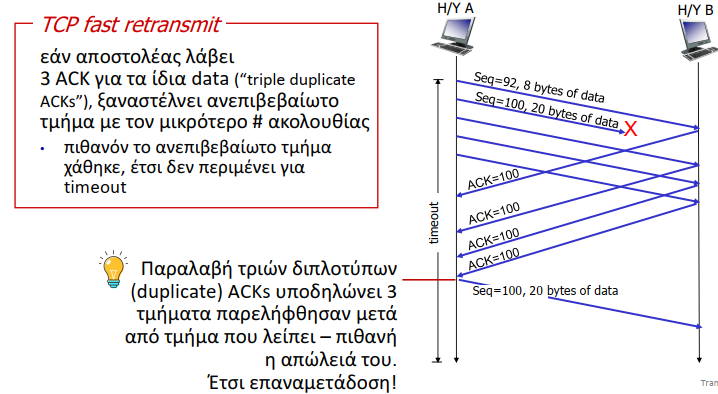
**Αποστολέας TCP (απλουστευμένο)**



**Παραλήπτης TCP: δημιουργία ACK [RFC 5681]**

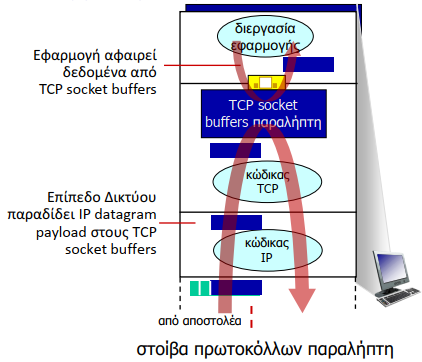
**TCP: σενάρια επαναμετάδοσης**

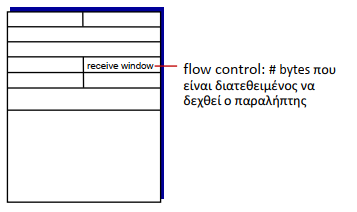


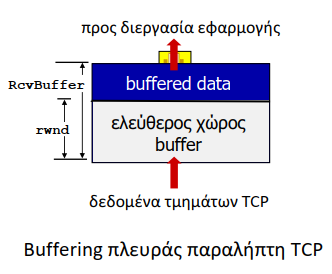
**TCP fast retransmit**

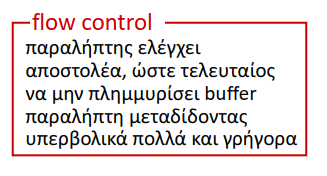
**Συνδεσμική μεταφορά: TCP**

**• Έλεγχος ροής  
• Διαχείριση συνδέσεων**

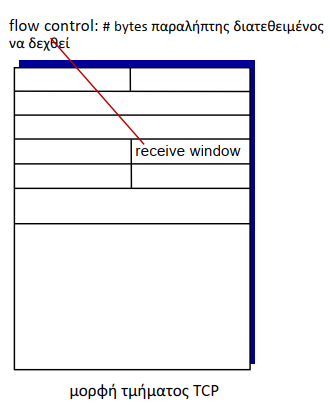
**TCP flow control (έλεγχος ροής)**

**Ε:**Τι συμβαίνει εάν το  
επίπεδο Δικτύου παραδίδει  
δεδομένα ταχύτερα από όσο  
το επίπεδο Εφαρμογών  
αφαιρεί δεδομένα από τους  
socket buffers;



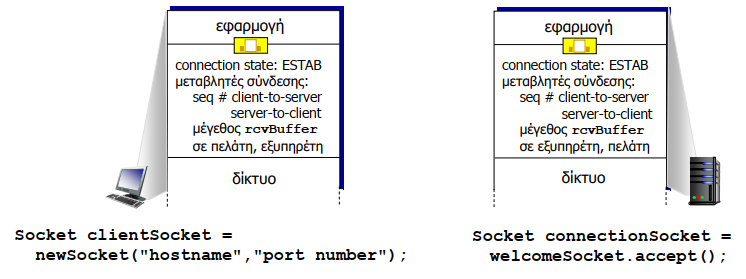


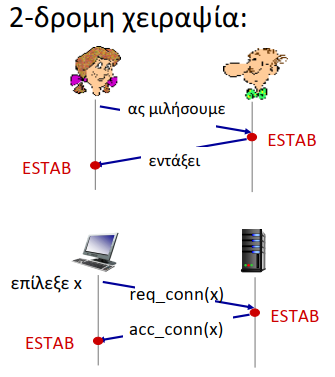
▪ παραλήπτης TCP “διαφημίζει” ελεύθερο χώρο buffer στο πεδίο **rwnd** της κεφαλίδος TCP

• μέγεθος **RcvBuffer** ορίζεται μέσω socket options (τυπική εξ ορισμού τιμή 4096 byte)  
• πολλά Λ/Σ προσαρμόζουν αυτόματα **RcvBuffer**

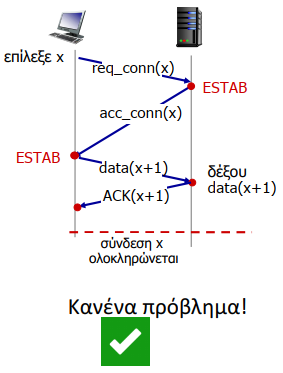
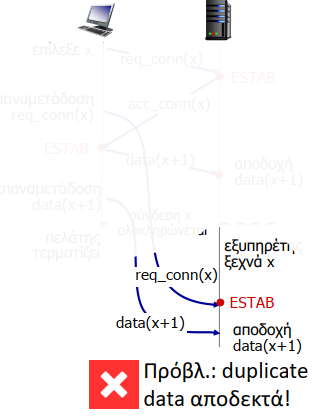
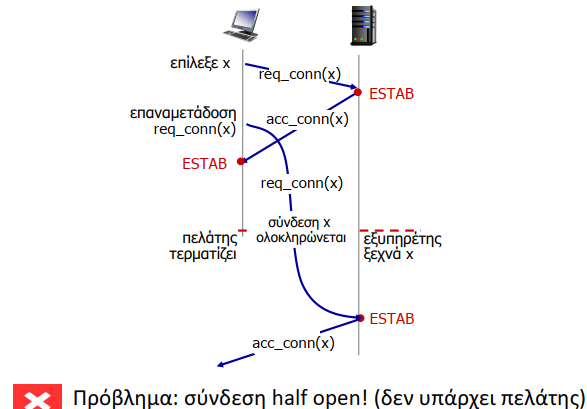
▪ αποστολέας περιορίζει πλήθος ανεπιβεβαίωτων (“σε-πτήση”)  
δεδομένων έως το ληφθέν **rwnd**▪ εγγύηση ότι buffer λήψης δεν θα υπερχειλίσει

**Διαχείριση σύνδεσης TCP**

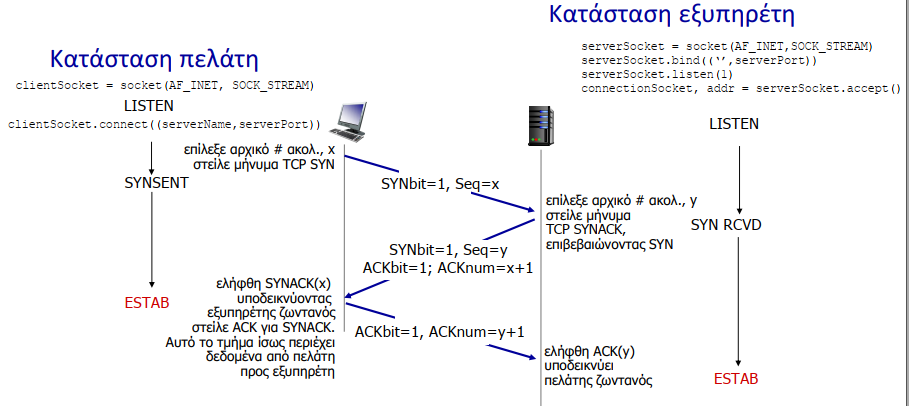
πριν ανταλλάξουν δεδομένα, αποστολέας/παραλήπτης “κάνουν χειραψία”:  
▪ συμφωνούν να δημιουργήσουν σύνδεση (γνωρίζοντας ο κάθε ένας ότι ο άλλος είναι πρόθυμος να δημιουργήσει σύνδεση)  
▪ συμφωνούν στις παραμέτρους σύνδεσης (π.χ., αρχικοί αριθμοί ακολουθίας)

**Συμφωνώντας στην δημιουργία σύνδεσης**

***Ε:*** η 2-δρομη χειραψία θα  
δουλεύει πάντα στα δίκτυα;  
**▪** μεταβλητές καθυστερήσεις **▪** επαναμεταδοθέντα μηνύματα (π.χ., req\_conn(x)) λόγω απωλειών μηνυμάτων **▪** αναδιάταξη μηνυμάτων  
**▪** δεν μπορεί να “δει” την άλλη πλευρά

**Σενάρια 2-δρομης χειραψίας**

**TCP: 3-δρομη χειραψία**



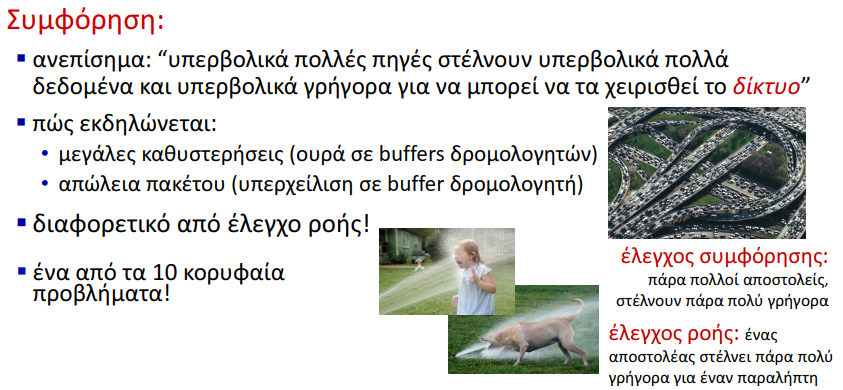
**Τερματίζοντας μία σύνδεση TCP**

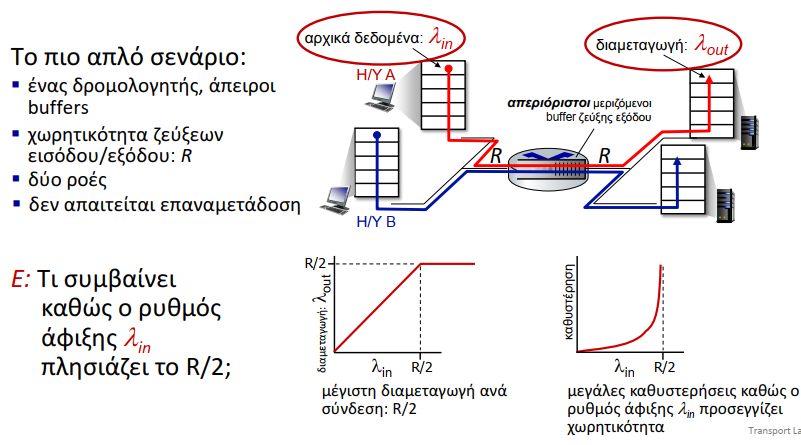
▪ κάθε πελάτης, εξυπηρέτης κλείνει την δική του πλευρά της σύνδεσης TCP

• στέλνει τμήμα TCP με bit FIN = 1  
▪ αποκρίνεται σε ληφθέν FIN με ACK

• με την λήψη FIN, το ACK μπορεί να συνδυασθεί και με δικό του FIN (δλδ. στο ίδιο τμήμα TCP)

▪ μπορεί να χειρισθεί και ταυτόχρονες ανταλλαγές FIN

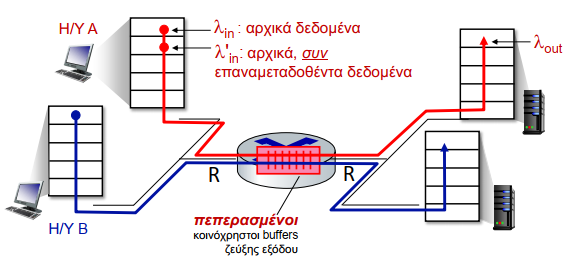
**Αρχές ελέγχου συμφόρησης**

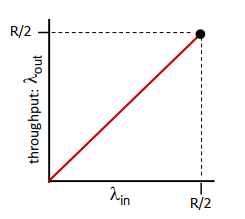
**Αιτίες/κόστη συμφόρησης: σενάριο 1**

**Αιτίες/κόστη συμφόρησης: σενάριο 2**

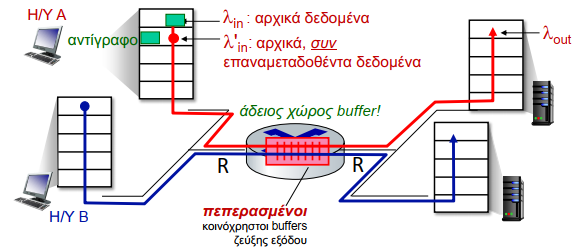
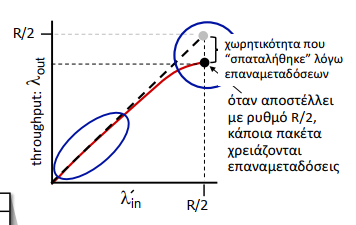
▪ ένας δρομολογητής, *πεπερασμένοι* buffers

▪ αποστολέας επαναμεταδίδει απολεσθέν, εκτός χρόνου packet

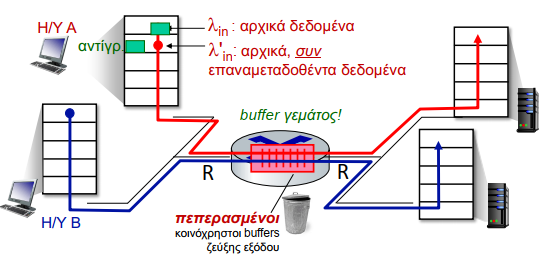
• είσοδος επιπέδου εφαρμογών = έξοδος επιπέδου εφαρμογών: lin = lout  
• είσοδος επιπέδου μεταφοράς περιλαμβάνει επαναμεταδόσεις: l’in ≥ lin

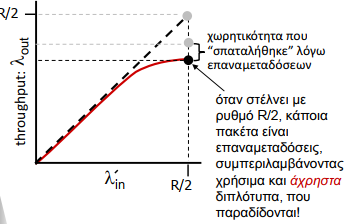
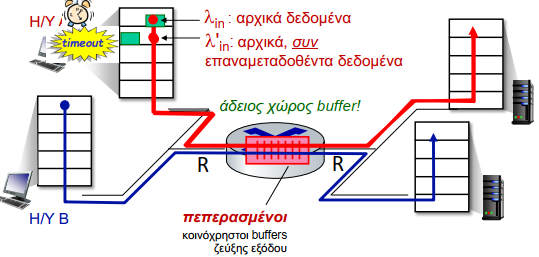


**Εξιδανίκευση: τέλεια γνώση** ▪ αποστολέας στέλνει μόνον όταν buffers δρομολογητών διαθέσιμοι



**Εξιδανίκευση: *κάποια* τέλεια γνώση**▪ πακέτα μπορεί να χαθούν (απορριπτόμενα) στον δρομολογητή) λόγω γεμάτων buffers  
▪ αποστολέας γνωρίζει πότε πακέτο έχει απορριφθεί: ξαναστέλνει μόνον εάν πακέτο *είναι γνωστό* ότι χάθηκε



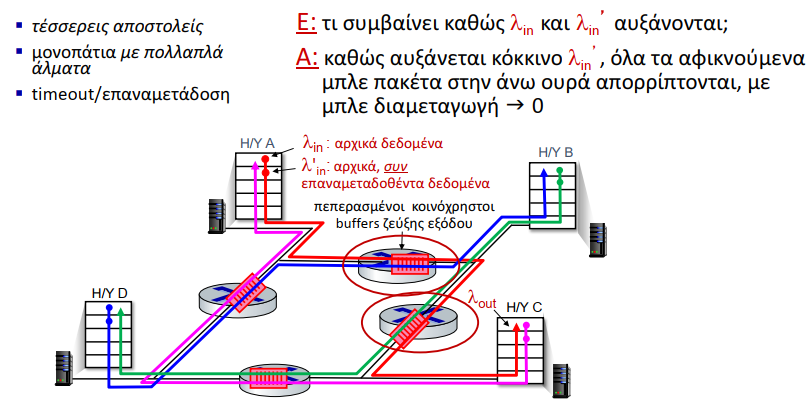
***Ρεαλιστικό σενάριο*: *άχρηστα διπλότυπα πακέτα***▪ πακέτα μπορούν να χαθούν, απορριπτόμενα σε δρομολογητή λόγω γεμάτων buffers, απαιτώντας επαναμεταδόσεις  
▪ αλλά χρόνοι αποστολέα μπορεί να λήξουν πρώιμα, αποστέλλοντας *2* αντίγραφα, που παραδίδονται *αμφότερα*

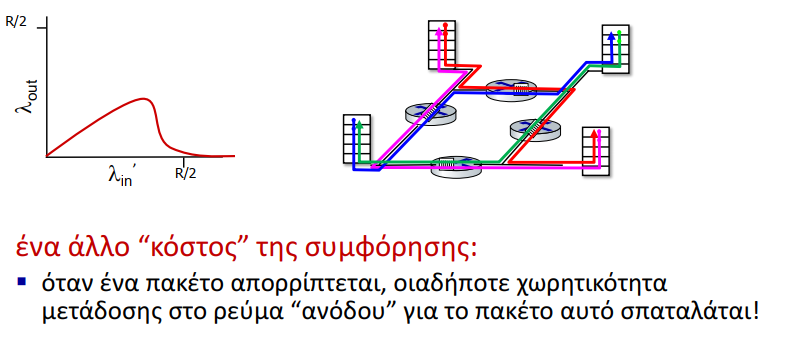
* αλλά χρονόμετρα αποστολέα μπορεί να λήξουν πρώιμα, αποστέλλοντας *δύο* αντίγραφα, που παραδίδονται *και τα δύο*

**“κόστη” της συμφόρησης:**

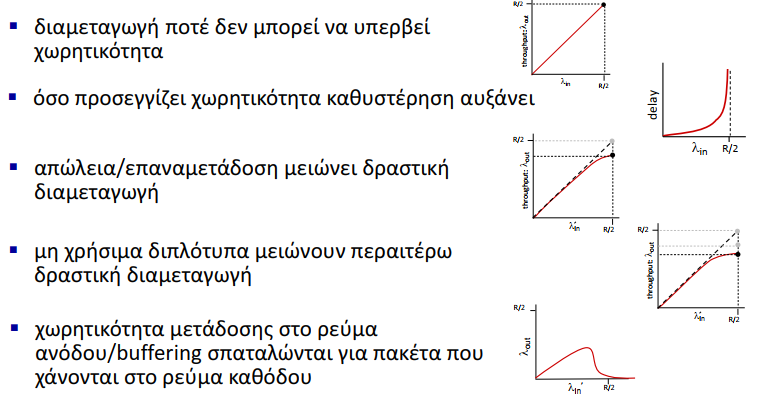
* περισσότερος φόρτος (επαναμεταδόσεις) για δεδομένη διαμεταγωγή παραλήπτη
* αχρείαστες: επαναμεταδόσεις: ζεύξη μεταφέρει πολλαπλά αντίγραφα ενός πακέτου
  + μειώνοντας μέγιστη επιτευκτή διαμεταγωγή

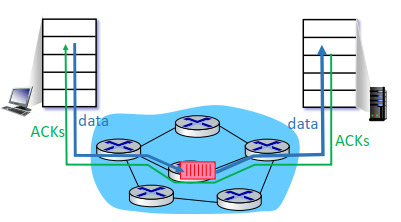
**Αιτίες/κόστη συμφόρησης: σενάριο 3**



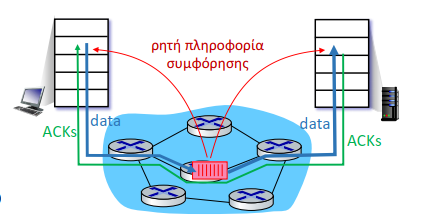


**Αιτίες/κόστη συμφόρησης: διορατικότητα**



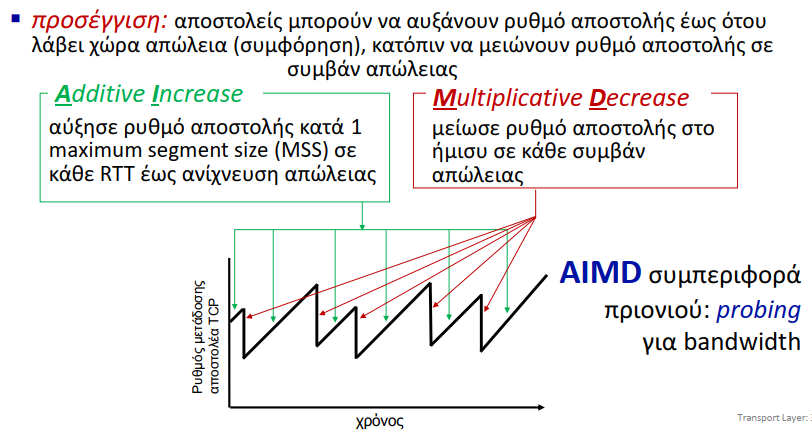
**Προσεγγίσεις στον έλεγχο συμφόρησης  
Έλεγχος συμφόρησης από-άκρο-σε-άκρο:**

▪ καμία ρητή ανάδραση (feedback) από το δίκτυο network  
▪ συμφόρηση *συνάγεται* από παρατηρούμενη απώλεια, καθυστέρηση  
▪ προσέγγιση που ακολουθείται από το TCP

**Έλεγχος συμφόρησης με βοήθεια του δικτύου:** ▪ δρομολογητές παρέχουν *άμεση* ανάδραση στους Η/Υ αποστολείς/παραλήπτες για ροές που περνούν μέσω δρομολογητών με συμφόρηση  
▪ μπορούν να υποδεικνύουν επίπεδο συμφόρησης ή να ρυθμίζουν ρητά ρυθμό αποστολής

▪ πρωτόκολλα TCP ECN, ATM, DECbit

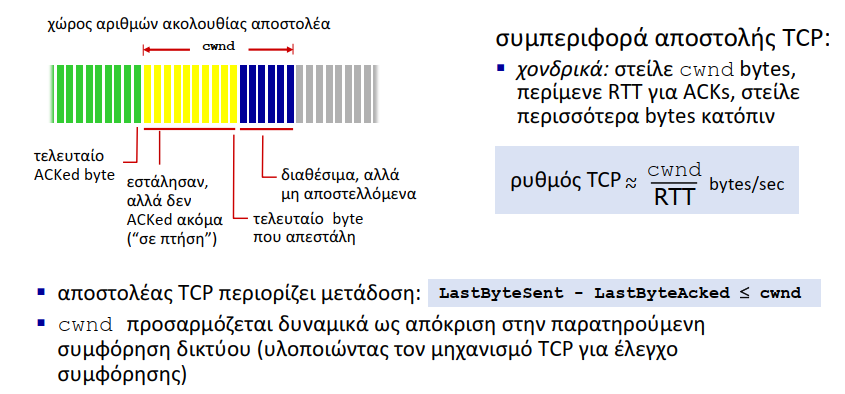
**Έλεγχος συμφόρησης TCP**

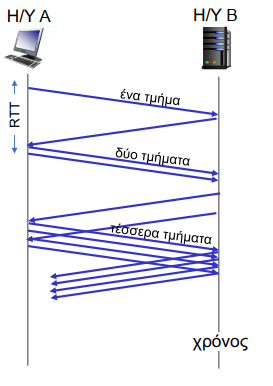
**Έλεγχος συμφόρησης TCP: AIMD**

**TCP AIMD: περισσότερα**

Λεπτομέρειες ***Multiplicative decrease*:** ρυθμός αποστολής γίνεται  
▪ μισός όταν ανιχνεύεται απώλεια από τριπλό διπλότυπο ACK (TCP Reno)  
▪ 1 MSS (maximum segment size) όταν ανιχνεύεται απώλεια από timeout (TCP Tahoe)  
Γιατί A I M D;  
▪ AIMD – κατανεμημένος, ασύγχρονος αλγόριθμος – έχει καταδειχθεί ότι:

• βελτιώνει ρυθμούς ροών με συμφόρηση σε ολόκληρο το δίκτυο!  
• έχει επιθυμητές ιδιότητες σταθερότητας

**Έλεγχος συμφόρησης TCP: λεπτομέρειες**

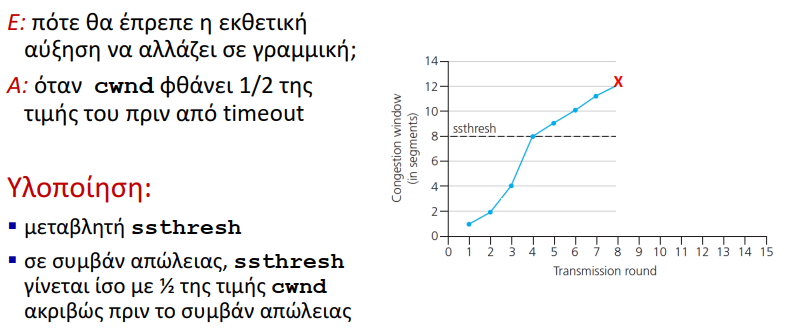
**TCP slow start**

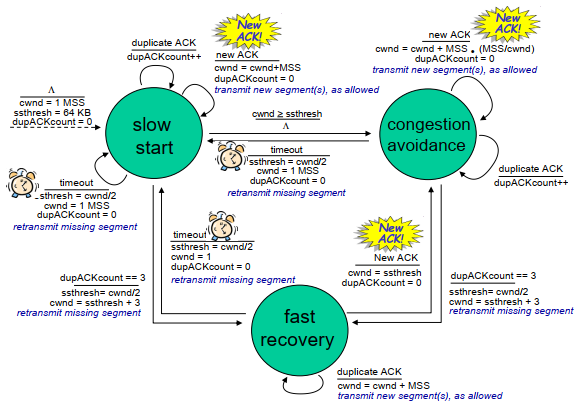
▪ όταν ξεκινά η σύνδεση, αύξησε ρυθμό εκθετικά έως πρώτο συμβάν απώλειας:

• αρχικά cwnd = 1 MSS  
• διπλασίασε cwnd κάθε RTT  
• υλοποιείται αυξάνοντας cwnd για κάθε ACK που λαμβάνεται

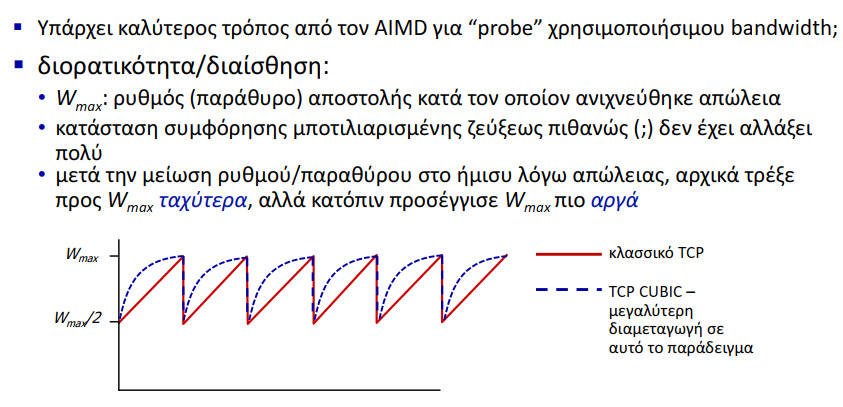
▪ *περίληψη:* αρχικός ρυθμός  
βραδύς, αλλά αυξάνεται  
εκθετικά γρήγορα

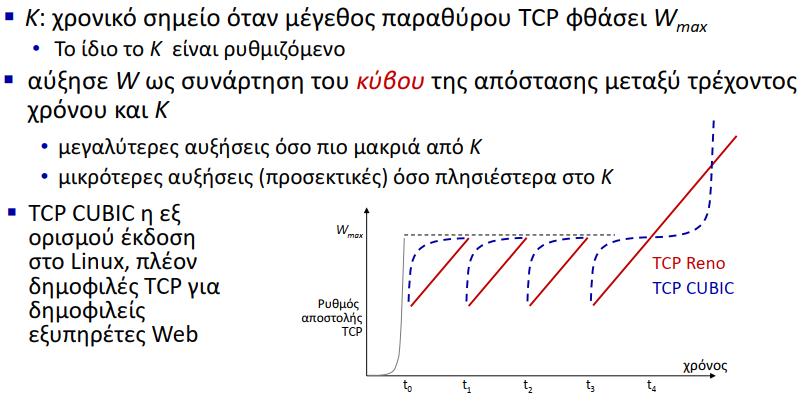
**TCP: από slow start σε congestion avoidance (αποφυγή συμφόρησης )**



**Περίληψη: έλεγχος συμφόρησης TCP**

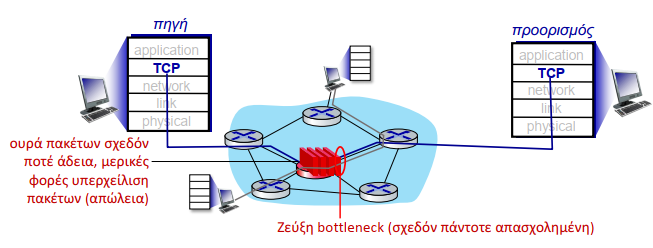
**TCP CUBIC**

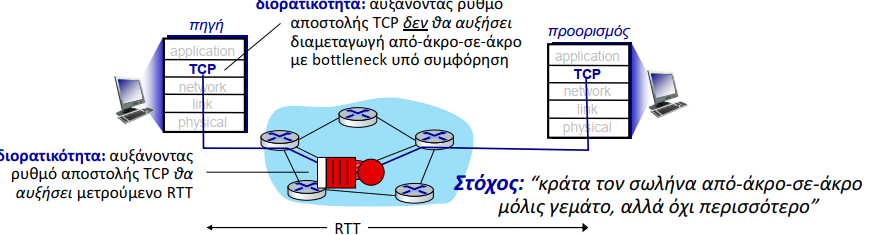


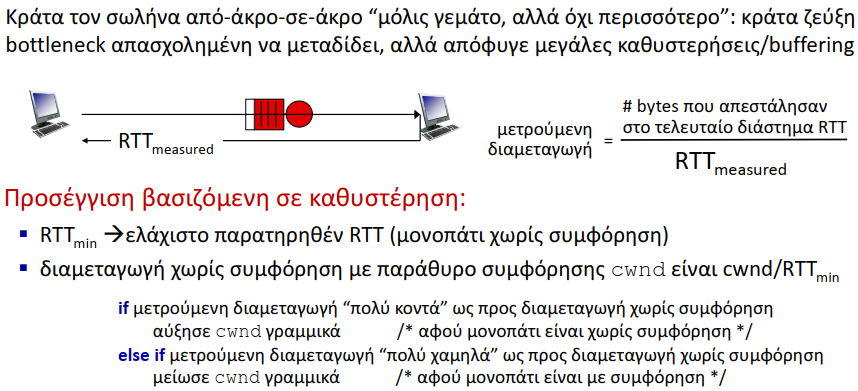


**TCP και η υπό συμφόρηση “ζεύξη λαιμού φιάλης”**

**▪ TCP (κλασσικό, CUBIC)** αυξάνει ρυθμό αποστολής του TCP έως ότου  
λάβει χώρα απώλεια πακέτου στην έξοδο κάποιου δρομολογητή: στην  
***ζεύξη λαιμού φιάλης (bottleneck link****)*

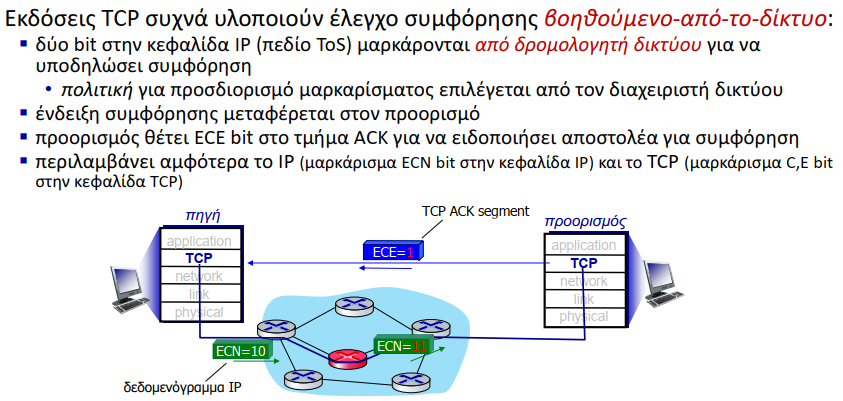
*▪ κατανοώντας την συμφόρηση: χρήσιμη η εστίαση στην υπό συμφόρηση  
ζεύξη bottleneck*



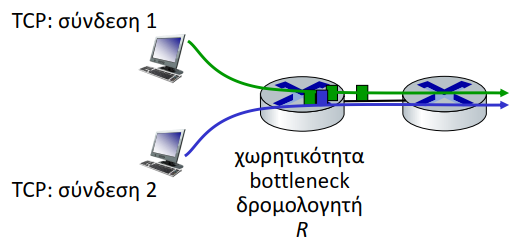
**Έλεγχος συμφόρησης TCP βασιζόμενος-σε-καθυστέρηση**

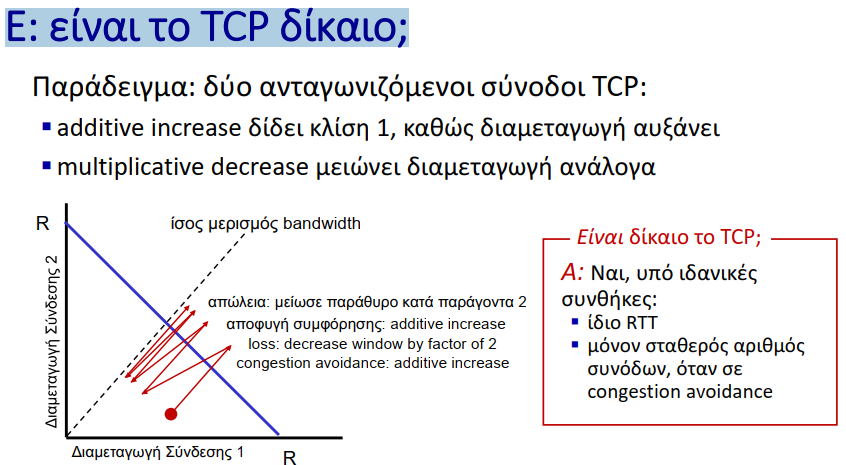
▪ έλεγχος συμφόρησης χωρίς να επιφέρει/επιβάλει απώλεια  
▪ μεγιστοποιεί διαμεταγωγή (“κρατώντας τον σωλήνα μόλις γεμάτο…”) ενώ διατηρεί χαμηλή την καθυστέρηση (“…αλλά όχι πιο γεμάτο”)  
▪ ένας αριθμός από εγκατεστημένα TCP ακολουθεί προσέγγιση βασισμένη- σε-καθυστέρηση

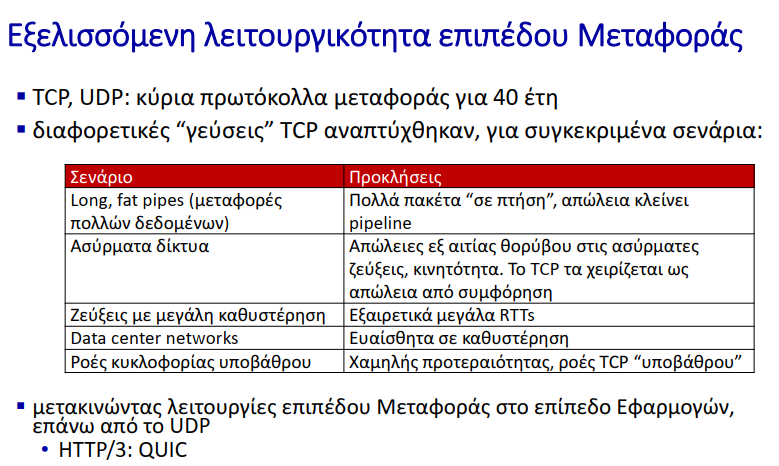
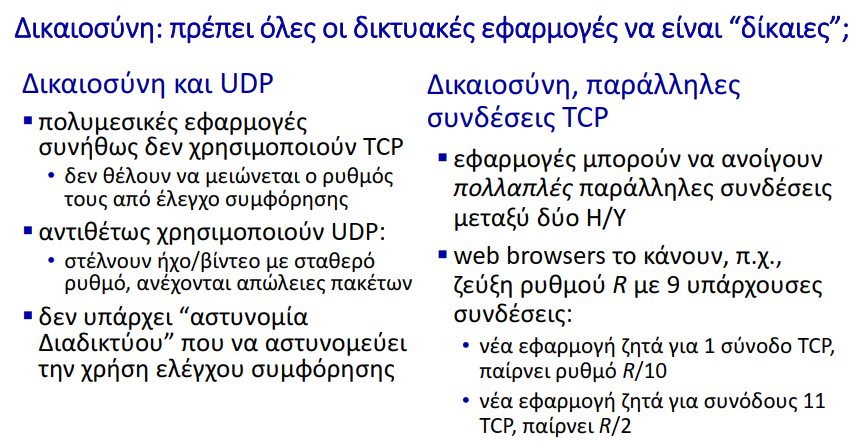
▪ TCP BBR εγκατεστημένο στο δίκτυο ραχοκοκαλιάς της Google (εσωτερικό)

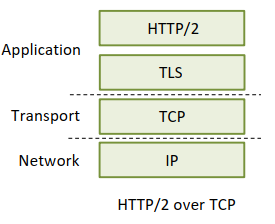
**Ρητή ειδοποίηση συμφόρησης (ECN)**

**Δικαιοσύνη TCP**

**Στόχος δικαιοσύνης:** εάν *K* σύνοδοι TCP μοιράζονται την ίδια ζεύξη λαιμού φιάλης bandwidth *R*, κάθε μία θα πρέπει να έχει μέσο ρυθμό *R/K*



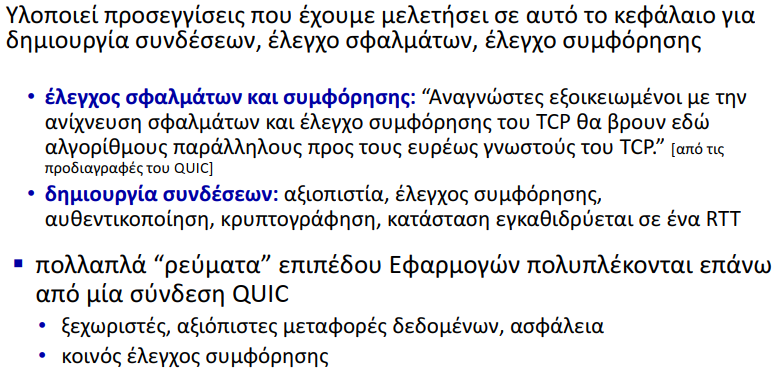
**Εξέλιξη λειτουργικότητας επιπέδου μεταφορών**

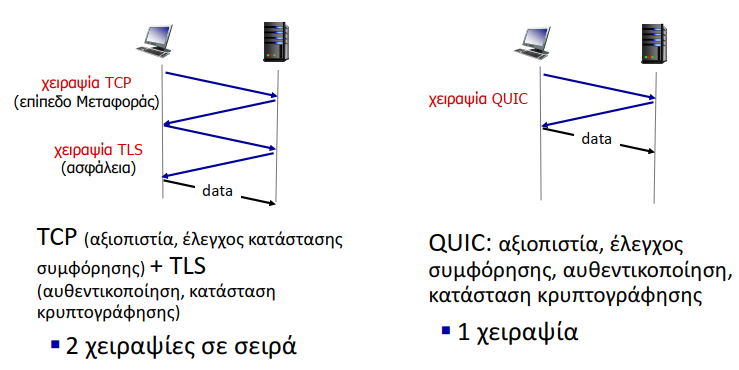


**QUIC: Quick UDP Internet Connections**

▪ πρωτόκολλο επιπέδου εφαρμογών, επάνω από UDP

• αυξάνει απόδοση HTTP   
• εγκατεστημένο σε πολλούς εξυπηρέτες της Google, εφαρμογές (Chrome, mobile YouTube app)



**QUIC: εγκαθίδρυση σύνδεσης**

**QUIC: ρεύματα: παραλληλισμός, μη HOL blocking**

