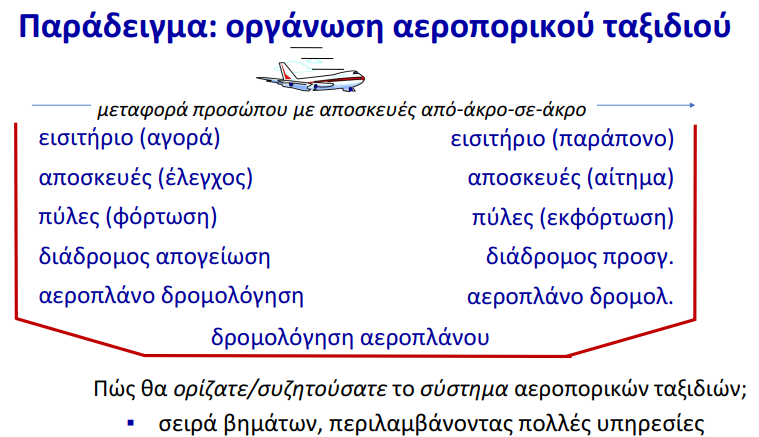
Δύκτια Υπολογιστών 23/10 4ο

Επίπεδα πρωτοκόλλων, μοντέλα υπηρεσιών

“Επίπεδα” πρωτοκόλλων και μοντέλα αναφοράς

*Ερώτηση:* υπάρχει καμία ελπίδα να *οργανώσουμε* την δομή ενός δικτύου;  
… και/ή την *συζήτησή μας για τα δίκτυα;*

Δίκτυα είναι πολύπλοκα, με πολλά “κομμάτια”:  
▪ hosts (Η/Υ)  
▪ δρομολογητές  
▪ ζεύξεις διαφόρων μέσων  
▪ εφαρμογές  
▪ πρωτόκολλα  
▪ υλικό, λογισμικό

Γιατί σε επίπεδα;

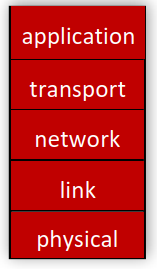
Προσέγγιση για σχεδιασμό/συζήτηση πολύπλοκων συστημάτων:

* ρητή δομή επιτρέπει ταυτοποίηση, σχέση μεταξύ πολύπλοκων κομματιών ενός συστήματος
  + *μοντέλο αναφοράς* με επίπεδα για συζήτηση

Reference model μοντέλο αναφοράς (ο οδηγός )

* αρθρωτή προσέγγιση διευκολύνει συντήρηση, ενημέρωση του συστήματος
  + μεταβολή *υλοποίησης* υπηρεσίας ενός επιπέδου: διάφανη στο υπόλοιπο σύστημα (σαν να μην άλλαξε τίποτα)
  + π.χ., μεταβολή στην διαδικασία πύλης δεν επηρεάζει το υπόλοιπο σύστημα

Στοίβα επιπέδων πρωτοκόλλων Διαδικτύου (γιατί, για μια ολοκληρωμένη στοίβα)

▪ *application (εφαρμογών):*υποστηρίζει δικτυακές εφαρμογές

(πρωτόκολλα επιπέδου εφαρμογών)

• HTTP, IMAP, SMTP, DNS  
▪ *transport (μεταφοράς):*μεταφορά δεδομένων

από διεργασία σε διεργασία (δεν είναι άμεσα

συνεργαζόμενες οι διεργασίες)

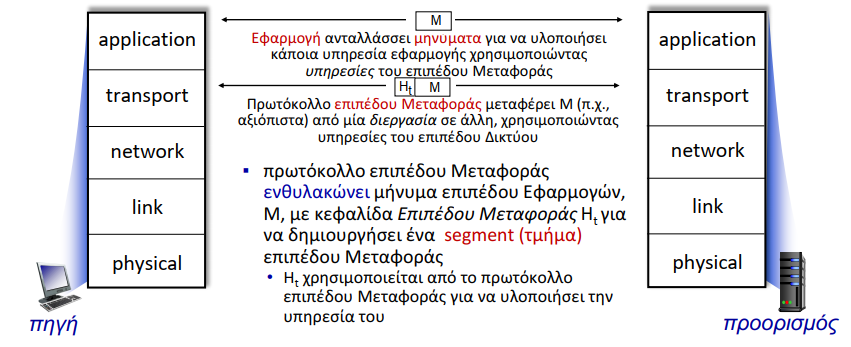
• TCP, UDP κεφ.3  
▪ *network (δικτύου):* δρομολόγηση  
δεδομενογραμμάτων (datagram) από την πηγή στον  
προορισμό (σε οποιαδίποτε συσκευή Η/Υ Α)

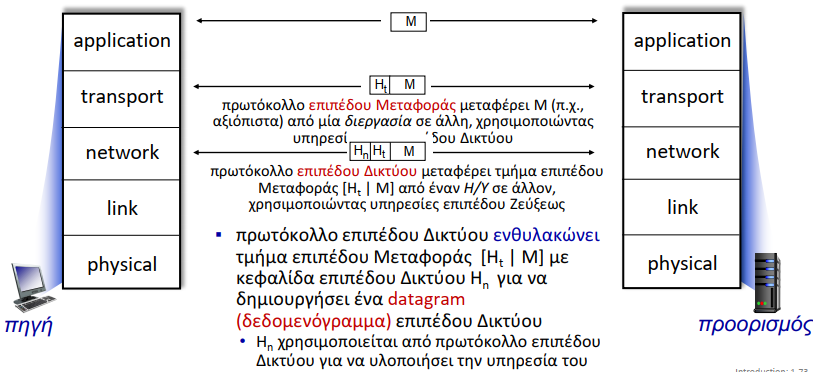
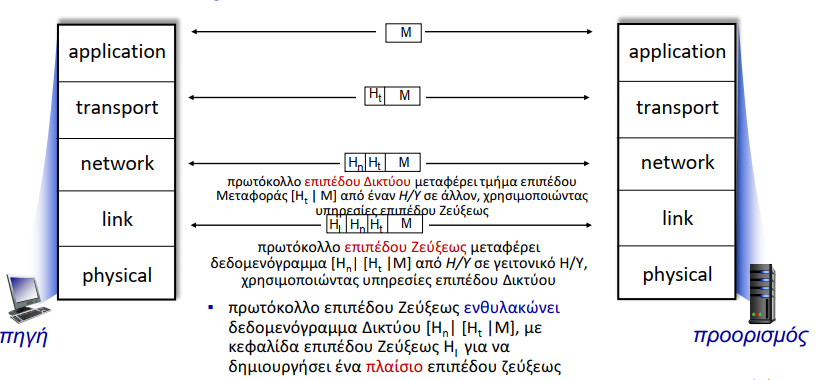
• IP, πρωτόκολλα δρομολόγησης κεφ 4, 5  
▪ *link (ζεύξεως):* μεταφορά δεδομένων μεταξύ

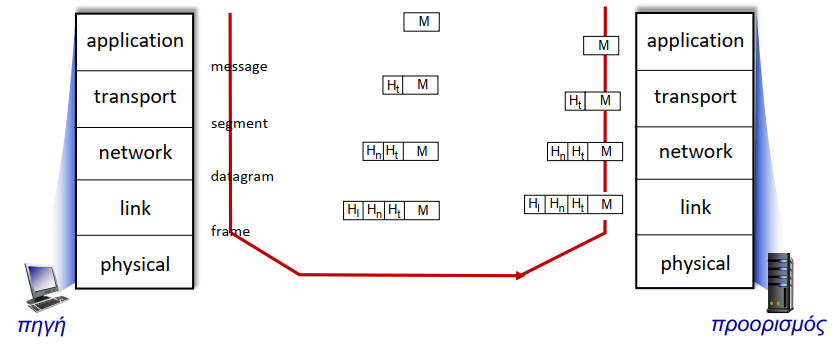
γειτονικών δικτυακών στοιχείων

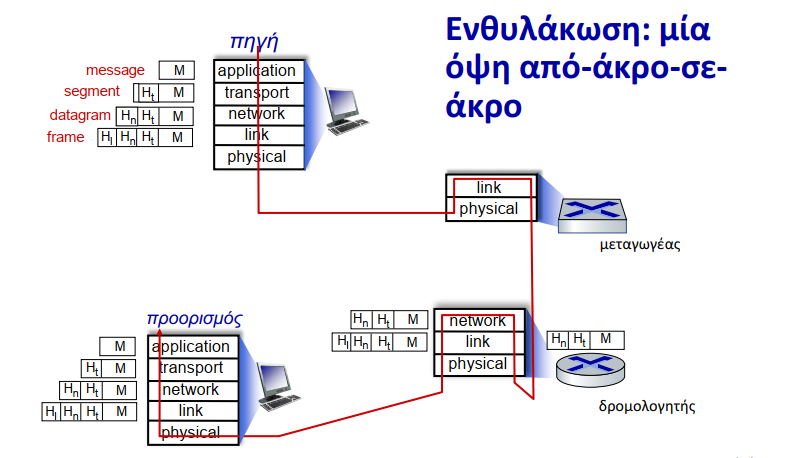
• Ethernet, 802.11 (WiFi), PPP  
▪ *Physical (φυσικό):* bits “στο σύρμα”

Υπηρεσίες, Οργάνωση σε Επίπεδα και Ενθυλάκωση (H=HEADER)

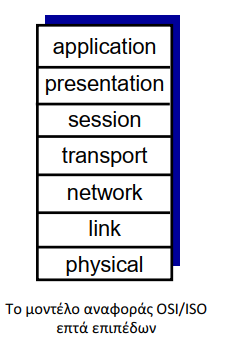








**Μοντέλο αναφοράς ISO/OSI**

Δύο επίπεδα που δεν βρίσκονται στην στοίβα πρωτοκόλλων του Διαδικτύου!  
▪ *presentation:* επιτρέπει σε εφαρμογές να  
διερμηνεύουν το νόημα των δεδομένων, π.χ.,  
κρυπτογράφηση, συμπίεση, συμβάσεις   
συγκεκριμένων μηχανών  
▪ *session:* συγχρονισμός, τοποθέτηση σημείων  
ελέγχου, ανάκτηση δεδομένων (που  
προήλθαν) από ανταλλαγή  
▪ από την στοίβα του Διαδικτύου “λείπουν”  
αυτά τα επίπεδα!

* αυτές οι υπηρεσίες, *εάν χρειάζονται,*πρέπει να υλοποιηθούν στην εφαρμογή
* χρειάζονται;

Chapter 2: Application Layer (Ανήκουν οι διαφορές δικτυακές εφαρμογές)

Αρχές δικτυακών εφαρμογών

Επίπεδο Εφαρμογών: επισκόπηση

οι στόχοι μας:  
▪ εννοιολογικά θέματα *και* θέματα υλοποίησης πρωτοκόλλων δικτυακών εφαρμογών  
• μοντέλα υπηρεσιών επιπέδου μεταφοράς  
• υπόδειγμα client-server (πελάτη-εξυπηρέτη)  
• υπόδειγμα peer-to-peer (ομότιμου-προς-ομότιμο)  
▪ να μάθουμε σχετικά με πρωτόκολλα εξετάζοντας δημοφιλή πρωτόκολλα στο επίπεδο εφαρμογών

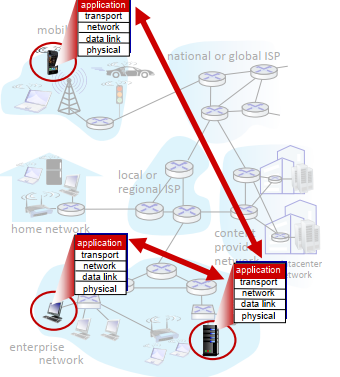
* HTTP
* SMTP, IMAP
* DNS
* συστήματα video streaming, CDNs

▪ προγραμματισμός δικτυακών εφαρμογών

* socket API

Μερικές δικτυακές εφαρμογές

▪ κοινωνική δικτύωση  
▪ Web  
▪ text messaging  
▪ e-mail  
▪ δικτυακά παιχνίδια πολλαπλών χρηστών  
▪ Μερικές δικτυακές εφαρμογές (YouTube, Hulu, Netflix)  
▪ διανομή αρχείων P2P  
▪ voice over IP (π.χ., Skype)  
▪ τηλεσυνεδρίαση πραγματικού χρόνου (π.χ., Zoom)  
▪ Αναζήτηση στο Διαδίκτυο  
▪ remote login  
▪ …  
*Ε: οι δικές σας* αγαπημένες;

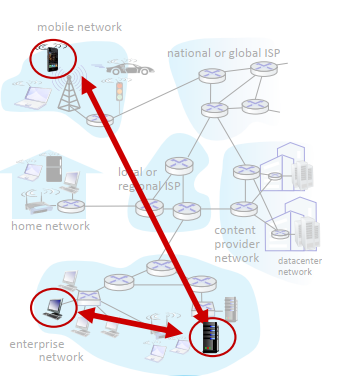
Δημιουργώντας μία δικτυακή εφαρμογή

γράφουμε προγράμματα που:  
▪ τρέχουν σε (διαφορετικά) τερματικά συστήματα  
▪ επικοινωνούν μέσω δικτύου

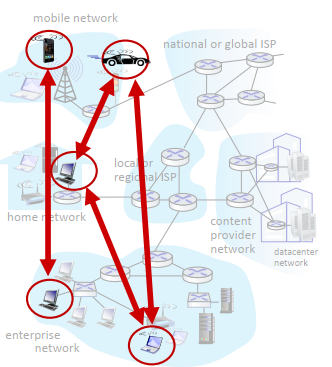
* π.χ., λογισμικό εξυπηρέτη web που

επικοινωνεί με λογισμικό browser

δεν χρειάζεται να γράφουμε λογισμικό  
για συσκευές πυρήνα δικτύου  
▪ τέτοιες συσκευές δεν τρέχουν εφαρμογές  
χρηστών  
▪ εφαρμογές σε τερματικά συστήματα  
επιτρέπουν γρήγορη ανάπτυξη, διάδοση

Υπόδειγμα πελάτη-εξυπηρέτη (client-server)

εξυπηρέτης:  
▪ Η/Υ πάντα ενεργός  
▪ μόνιμη δνση IP   
▪ συχνά σε data centers για κλιμάκωση  
πελάτης:  
▪ έρχονται σε επαφή, επικοινωνούν με  
εξυπηρέτη  
▪ μπορούν να συνδέονται κατά διαστήματα  
▪ μπορούν να έχουν δυναμικές δνσεις IP  
▪ *δεν* επικοινωνούν απ’ ευθείας ο ένας με  
τον άλλον  
▪ παραδείγματα: HTTP, IMAP, FTP

Αρχιτεκτονική P2P(σημ)

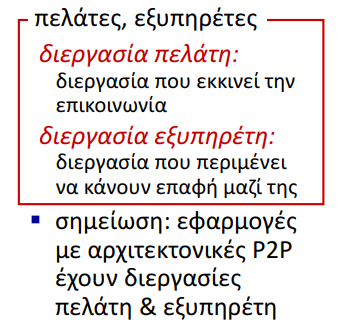
▪ εξυπηρέτης *μη* ενεργός συνεχώς  
▪γιατί, οπιαδήποτε τερματικά συστήματα   
επικοινωνούν απ’ ευθείας μεταξύ τους  
▪ ομότιμοι ζητούν υπηρεσία από άλλους  
ομότιμους, παρέχουν με την σειρά τους  
υπηρεσία σε άλλους ομότιμους (σαν να κατεβάζεις ένα κομμάτι αρχείου ταυτόχρονα δίνω)

• *αυτο-κλιμάκωση(self-scale) – νέοι ομότιμοι φέρνουν νέα χωρητικότητα υπηρεσίας, καθώς και νέες απαιτήσεις υπηρεσίας*

▪ ομότιμοι συνδέονται κατά διαστήματα  
και αλλάζουν δνσεις IP

• πολύπλοκη διαχείριση  
▪ παράδειγμα: διαμοιρασμός αρχείων P2P

Οι παίκτες εδώ ονομάζονται pears γιατί ο καθένας συνδεεται

Επικοινωνούσες διεργασίες

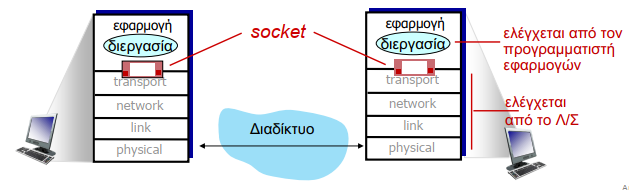
*διεργασία:* πρόγραμμα σε εκτέλεση σε Η/Υ

* γιατί, αυτό είναι ιδιαίτερό;
* Επιδή ο χώρος διεύθυνσης που βρίσκεται στη μνημη είναι κενός
* μέσα στον ίδιο Η/Υ, δύο διεργασίες επικοινωνούν με inter-process communication  
  (ορίζεται από το Λ/Σ)
* διεργασίες σε διαφορετικούς Η/Υ επικοινωνούν  
  ανταλλάσσοντας μηνύματα

Sockets API

▪ διεργασία στέλνει/λαμβάνει μηνύματα προς/από την socket της  
▪ socket ανάλογη με πόρτα

* + - αποστέλλουσα διεργασία σπρώχνει μήνυμα έξω από την πόρτα
    - αποστέλλουσα διεργασία βασίζεται στην υποδομή μεταφοράς στην άλλη πλευρά της πόρτας για παράδοση του μηνύματος στην socket της λαμβάνουσας διεργασίας
    - εμπλέκονται δύο sockets: μία σε κάθε πλευρά



Διευθυνσιοδότηση διεργασιών

▪ *identifier* περιλαμβάνει αμφότερες  
δνση IP και αριθμό θύρας που  
σχετίζονται με διεργασία στον Η/Υ  
▪ παράδειγμα αριθμών θύρας:

* εξυπηρέτης HTTP : 80
* εξυπηρέτης mail: 25
* για αποστολή μηνύματος HTTP στον εξυπηρέτη web gaia.cs.umass.edu:  
  • δνση IP: 128.119.245.12  
  • αριθμός θύρας: 80

▪ σύντομα περισσότερα…

▪ Για λήψη μηνυμάτων η  
διεργασία πρέπει να έχει  
*identifier (αναγνωριστικό)*▪ Η/Υ έχει μοναδική δνση IP 32-bit  
▪ *Ε:* δνση IP Η/Υ όπου τρέχει η  
διεργασία, επαρκής για  
ταυτοποίηση της διεργασίας;  
*Α:* όχι, μπορούν να τρέχουν *πολλές*διεργασίες στον ίδιο Η/Υ

Δεν είναι επαρής -> το κάτι παραπάνω

Είναι ο αριθμός θύρας

Ένα πρωτόκολλο επιπέδου Εφαρμογών ορίζει:

* τύπους ανταλλασσομένων μηνυμάτων
  + π.χ., αίτημα, απόκριση
* συντακτικό μηνυμάτων:
  + ποια τα πεδία & πώς οριοθετούνται στα μηνύματα
* σημασιολογία μηνυμάτων
  + νόημα πληροφοριών στα πεδία
* κανόνες για το πότε και πώς διεργασίες στέλνουν & αποκρίνονται σε μηνύματα
* ανοικτά πρωτόκολλα:
  + ορίζονται σε RFC, όλοι έχουν πρόσβαση στους ορισμούς  
    πρωτοκόλλων
  + επιτρέπουν διαλειτουργικότητα
  + π.χ., HTTP, SMTP
* κλειστά πρωτόκολλα:
  + π.χ., Skype, Zoom

Τι υπηρεσία μεταφοράς χρειάζεται μία εφαρμογή;

ακεραιότητα δεδομένων

* μερικές εφαρμογές (π.χ., μεταφορά αρχείων, συναλλαγές web) απαιτούν  
  100% αξιόπιστη μεταφορά δεδομένων
* άλλες εφαρμογές (π.χ., ήχος) μπορούν να ανεχθούν κάποιες απώλειες

χρονισμός

* μερικές εφαρμογές (π.χ., τηλεφωνία, Internet, διαδραστικά παιχνίδια απαιτούν χαμηλή καθυστέρηση γιανα είναι “αποτελεσματικές

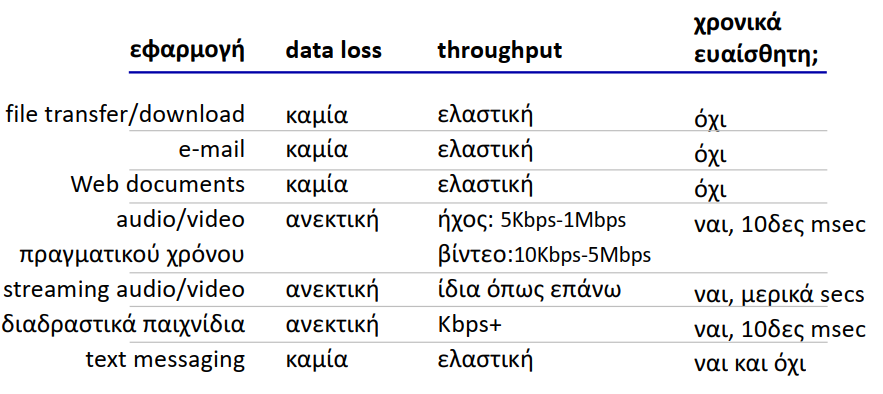
διαμεταγωγή

* μερικές εφαρμογές (π.χ., πολυμέσα) απαιτούν ελάχιστο ποσό διαμεταγωγής για να είναι “αποτελεσματικές”
* άλλες εφαρμογές (“ελαστικές”) χρησιμοποιούν όση διαμεταγωγή βρουν

Ασφάλεια

* κρυπτογράφηση, ακεραιότητα δεδομένων, …

Απαιτήσεις από υπηρεσία μεταφοράς: συνήθεις εφαρμογές

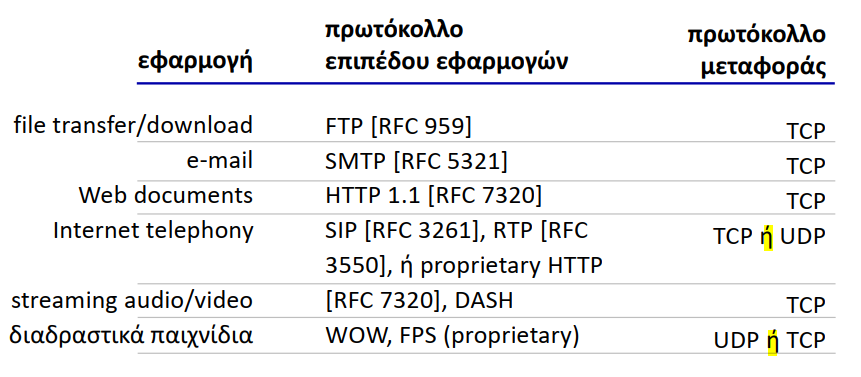


Υπηρεσίες πρωτοκόλλων μεταφοράς στο Διαδίκτυο

*υπηρεσία TCP:*▪ *αξιόπιστη μεταφορά* μεταξύ αποστέλλουσας και λαμβάνουσας διεργασίας  
▪ *έλεγχος ροής:* αποστολέας δεν πλημυρίζει τον παραλήπτη  
▪ *έλεγχος συμφόρησης(contjustion control):* μείωσε ρυθμό αποστολέα όταν δίκτυο υπερφορτωμένο  
▪ *συνδεσμική:* απαιτείται εγκαθίδρυση=καθορισμός μεταξύ διεργασιών πελάτη και εξυπηρέτη  
▪ *δεν παρέχει:* χρονισμό, εγγύηση ελάχιστης διαμεταγωγής, ασφάλεια

*υπηρεσία UDP:*▪ *αναξιόπιστη μεταφορά δεδομένων* μεταξύ αποστέλλουσας και λαμβάνουσας  
διεργασίας  
▪ *δεν παρέχει:* αξιοπιστία, έλεγχο ροής, έλεγχο συμφόρησης, χρονισμό, εγγύηση διαμεταγωγής, ασφάλεια, ή εγκαθίδρυση σύνδεσης  
Ε: γιατί ασχολούμαστε;  
Γιατί υπάρχει το UDP

Εφαρμογές Διαδικτύου & πρωτόκολλα επιπέδου Μεταφοράς



Ασφαλίζοντας το TCP

Συμβατικές socket TCP & UDP:

* μη κρυπτογράφηση
* μη κρυπτογραφημένα password που αποστέλλονται μέσα σε socket διασχίζουν το Διαδίκτυο μη κρυπτογραφημένα (!)

->Ασφάλεια Επιπέδου Μεταφοράς (TLS – Transport Layer Security)

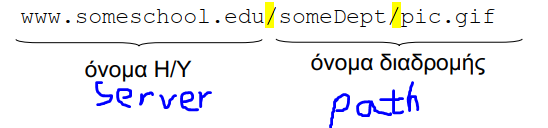
* παρέχει κρυπτογραφημένες συνδέσεις TCP
* ακεραιότητα δεδομένων
* αυθεντικοποίηση στα άκρα

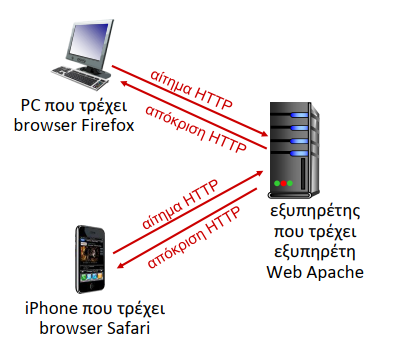
TLS υλοποιείται στο επίπεδο εφαρμογών

* εφαρμογές χρησιμοποιούν βιβλιοθήκες TLS, που με την σειρά τους χρησιμοποιούν το TCP
* μη κρυπτογραφημένο περιεχόμενο αποστέλλεται μέσα σε “socket” διασχίζοντας το Διαδίκτυο κρυπτογραφημένο

▪ περισσότερα: Κεφάλαιο 8

Web και HTTP

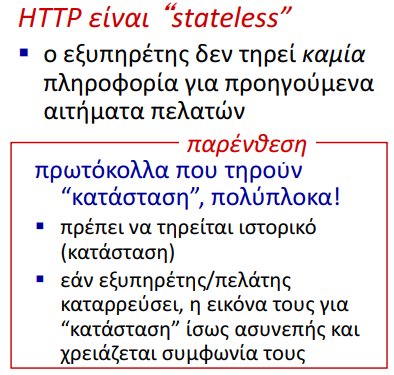
*Πρώτα, μία σύντομη εξέταση…*▪ ιστοσελίδα αποτελείται από *αντικείμενα,* κάθε ένα εκ των οποίων μπορεί να αποθηκεύεται σε διαφορετικούς εξυπηρέτες Web  
▪ αντικείμενο μπορεί να είναι αρχείο HTML, εικόνα JPEG, applet Java, αρχείο ήχου,…  
▪ ιστοσελίδα αποτελείται από *αρχείο base HTML* που περιέχει *αρκετά αναφερόμενα αντικείμενα, με κάθε ένα* να μπορεί να εντοπισθεί μέσω ενός *URL,* π.χ.,

Επισκόπηση HTTP

HTTP: hypertext transfer protocol

* το πρωτόκολλο επιπέδουεφαρμογών του Web
* μοντέλο πελάτη/εξυπηρέτη:
  + *πελάτης:* browser που αιτείται, λαμβάνει, (με χρήση HTTP) και “απεικονίζει” αντικείμενα Web
  + *εξυπηρέτης:* εξυπηρέτης Web αποστέλλει αντικείμενα (με χρήση HTTP) αποκρινόμενος σε αιτήματα

HTTP χρησιμοποιεί TCP:

▪ πελάτης αρχικοποιεί σύνδεση TCP (δημιουργεί socket) προς εξυπηρέτη, θύρα 80

▪ εξυπηρέτης αποδέχεται σύνδεση TCP από πελάτη

▪ μηνύματα HTTP (μηνύματα πρωτοκόλλου επιπέδου εφαρμογών) ανταλλάσσονται μεταξύ browser (πελάτης HTTP) και εξυπηρέτη Web (εξυπηρέτης HTTP)

▪ σύνδεση TCP κλείνει

Συνδέσεις HTTP: δύο τύποι

*persistent HTTP*▪ δημιουργείται σύνδεση TCP σε έναν εξυπηρέτη  
▪ πολλαπλά αντικείμενα μπορούν να σταλούν επάνω από *μοναδική* σύνδεση TCP μεταξύ πελάτη, και αυτού του εξυπηρέτη  
▪ σύνδεση TCP κλείνει

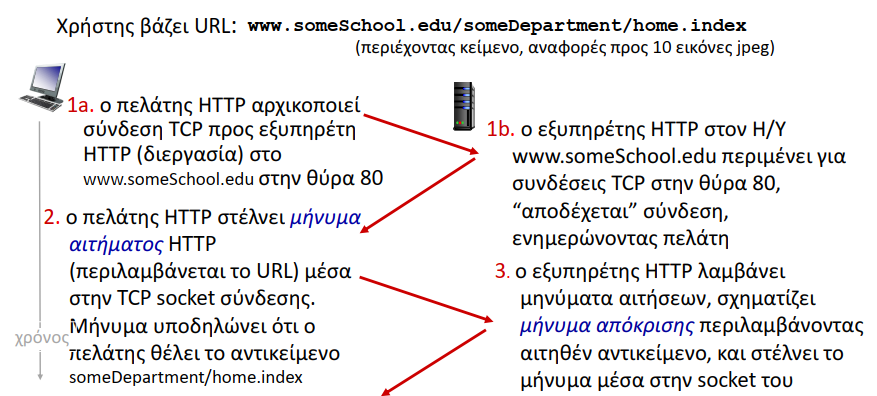
*Non-persistent HTTP*1. δημιουργείται σύνδεση TCP  
2. ένα το πολύ αντικείμενο αποστέλλεται

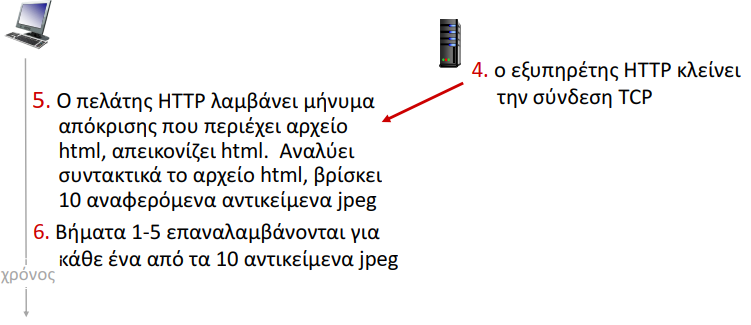
επάνω από σύνδεση TCP  
3. σύνδεση TCP κλείνει

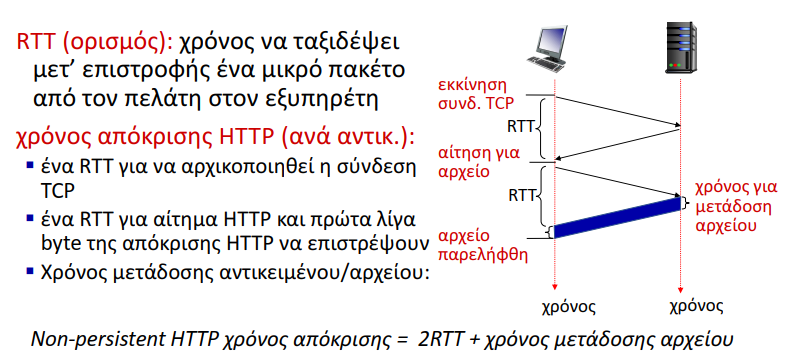
για κατέβασμα πολλαπλών αντικειμένων

απαιτούνται πολλαπλές συνδέσεις

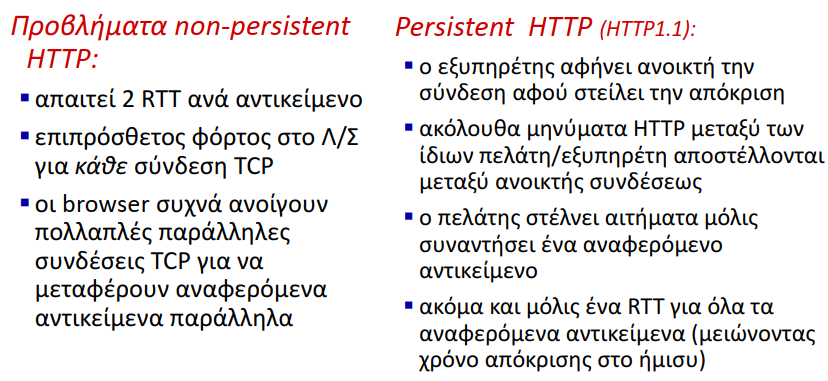
Non-persistent HTTP: παράδειγμα





Non-persistent HTTP: χρόνος απόκρισης

Persistent HTTP (HTTP 1.1)

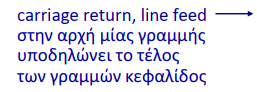


Μήνυμα αιτήματος HTTP

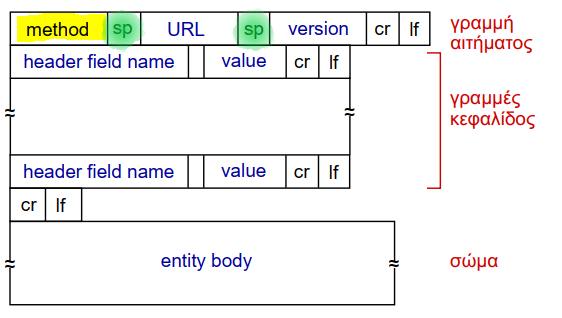
▪ δύο τύποι μηνυμάτων HTTP: *request*, *response*▪ μήνυμα αιτήματος HTTP:

* ASCII (μορφή αναγνώσιμη από ανθρώπους)



Μήνυμα αιτήματος HTTP: γενική μορφή

πεδία

συμβολοσειρές

μορφοποιημένες

σε διαφορετικες

γραμμές

cr, if = τηλέτυπα γραφόμηχανές