CHAPTER 2:- Application Layer:-

Sec 2.1:- principles of network applications

See Slide 3-4:-

* Creating a network app:-
  - write programs that:-
    - run on different end systems.
    - Communicate over network.

مابنكتب أبليكيشين لل network-core اللي فيها راوتر و switches ، الابلكيشين بتنبي عادةً على end systems على host عند clients

* Application architectures:-

1- Client - Server:- like Google, ritaj

- Server:- more powerfull
  1- Always-on host
  2- Permanent IP address. دايمًا موجود
  3- Data center — شبه ثابت IP

- Clients:-
  1. communicate with server — الكلينت بيلش حكي مع السيرفر
  2- may be intermittently connected — مش دايمًا متصل بالنت
  3- Dynamic IP — مش عند IP الثابت في كل مكان IP اشكل.
  4- Do not communicate directly with each other.
  الكلينت مابقدروا يحكوا مع بعض بشكل مباشر بدون سيرفر.

2- P2P architectures:- peer to peer

- no always-on server — ناس بتحكي مع بعض كيف؟!

مثلًا احنا 10 وكل واحد عنده 100 اجيجا فايل وبدنا نشاركها مع بعض بدله نجوّع معين أو سيرفر ويحكي للكل هي الفايل موجود هناك.

عن طريق P2P :- بخليهم عندي وبصير كلينت وسيرفر بنفس الوقت ، أعطي وآخذ وأحكي للكل كيف يوصلوا على الفايلات وعلى الأقل لازم أعطيهم ليسته

⇒

فيها IP، وشو الملفات الي عندك أو بعضهم link كيف يوصلولهم.

2- حلول p2p عندي سيرفر واحد عليه links للفايلات مش أرفعهم على الدرايف X.

3- عن طريق الهاش تيبل DHT آخر شابتر.

*** process Communicating:-** كيف client وserver يحكوا مع بعض

process:- program runnig within a host

على نفس سيرفين أو host بنعمل Inter-process communication بواسطة OS

اذا ابعاد عن طريق ال messages يعني host مختلفات يكونوا

المعرفة Client process:- process that initiates communication.

Server process:- process that waits to be contacted.

*** Sockets:-**

يختاج IP-Address بس ما بكفي لحاله بدنا socket.

Socket:- لما يفتح مثلًا المتصفح يكون فاتح على أكثر من site، مثلًا youtube و facebook

عندي IP Address واحدة مشبوك عليها حيث لما أفتح كل هدول وأكثر شو الي راح يميزهم

عن بعض هنا sockets:- process sends/receives messages to/from it

بتقدر تفتح أكثر من وحدة لكل وحدة إلها IP. بطلع من جهازي socket للمطرح الي

بدي إياه لما أطلب رابط مثلًا وهتبعه على port 5000 برجعلي socket على port 5000

هسا أنا بحكي مع نت بشكل عام على port 80 ← http بس ما بستخدم port 80

بستخدم socket مع بورت ثاني.

*** Addressing processes:-**

version4 → host has unique 32-bit IP Address $2^{32} = 4G$ = 4مليار — خلصوا — بس ما بكفوا لحالهم

identifier :- IP Add. + port number — عنا البورت تحت.

↓ بحدد وين البكيت راح يروح

port number e.g.:- HTTP: 80 , mail: 25

nowadays using IP Address version 6 :- 128 bit ← 2^{128} رقم كبير، مش الكل

* App-layer protocol defines:-

- types of messages exchanges:- request & response
- message syntax:- fields in message.
- message semantics:- meaning of information in fields.
- rules:- for when and how process send & respond to message.
- Open protocols:- أشياء defined in RFCs Request for comments)
  معرفة عالمياً allows for interoperability HTTP, SMTP
- proprietary protocols:- skype مملوكة لشركات أو لخاصة زي

* What transport service does an app need?

Data integrity: لو صار loss نعرف.
timing:- require low delay to be effective صعبة نحلها
throughput:- // minumum amount of throuput كمان صعب
Security:- لحالها بتجي صعب تجيبها مع كل شي
Slide 14 important.

* Internet transport protocols services:-

A) TCP:- Transmition Control protocol.

بيعتمد عليه • reliable transport:- TCP موجودة اذا فاي بكيت أول شي بدي أعرف اذا وصل أو لا
between sending and receiving وأعرف مين البكيت الي فاي برجع sender بيجيه
لكل بكيت ال IP, port و parity ، كيف راح ال receiver يميز مين الي فاي ؟!
عن طريق (ID) ← sequence number.

موجودة في TCP • flow Control: أنا ببعت للريسيفر بس كمان بديش أثقل ال buffer تاعته
ف TCP بشوف اذا البفر للريسيفر فل بقلل البعت.
reseiver ببعت عشان أعرف كديش ظايل buffer.
وفي كمان Acknowladge تاعت ال packed منها رسيفر بعرف كديش ظايل
reseiver buffer.

ازدحام.

- **Congestion Control:-** موجودة في TCP.
إذا حسيت انه نت ضعيفة لأنه يكون في كثير داتا نبعتت ← network full
TCP بنزل عدد الـ packet اللي بيبعتهم بقلل.

chapter 1 slide 9

- **Connection Oriented:-** موجودة في TCP ← create socket
قبل ما يبلش يبعت الـ packet بستأذن اول سيرفر موجود؟ واذا اه بتفقوا على شغلات
الـ Sengurnization.

- **doesnot provide:-** مش موجود في TCP.
timing, minum throput guarantee, Security.

B) **UDP:- User Data protocol**

- **Unreliable data transfer:-**
عكس TCP ما بستأذن ولا بعرف انه البكيت شاع همه بس بيبعت داتا وبشكل سريع.

- **Does not provide:-** كل TCP
reliability, flow control, Congestion control, timing, throput
Security or connection setup.

See slide 16 important. ↓ TCP more using.
يوتيوب واللي زيها صاروا يقدروا على حلول عشان يستعملوا UDP.

none →

* **Securing TCP:-**
(TCP & UDP not Secure, password send clear text).
الحل أني ابني مدول فوق layer ثانية للسيكيورتي وعادة مبستعمل SSL ↑
SSL هي اللي بتعطي الـ Security ← encrypted, data integrity, end point authentication.
SSL is at app layer: SSL libraries. مش كثير مهمات
SSL Socket.

L6:- Sec 2.2:- Web and HTTP:-

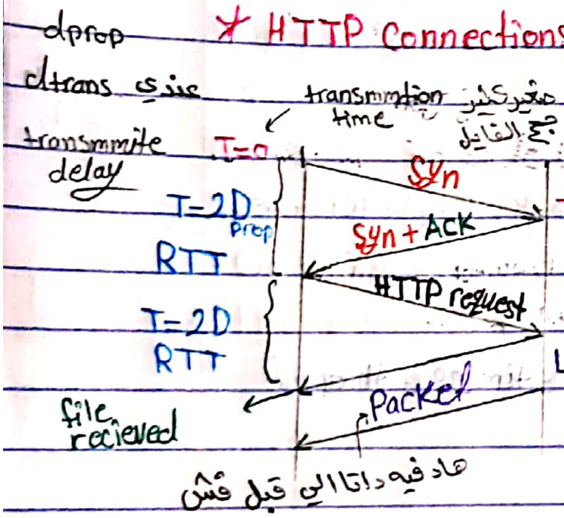
أول اشي بيبعت HTTP request بيجيني HTTP response على أي جهاز كمبيوتر أو تلفون
السيرفر الي بشتغل هو الـ Apache سيرفر المشهور لنت.

HTTP use TCP:-

- client initiates TCP → create socket, port 80
- Server accepts TCP connection.
- HTTP use to send data between client and servier.
- TCP connection close

HTTP نفسه ما بعرف ويه أنا وصلت عاذا أنا حتى ديه وصلت لا هدول البروتوكول stateless
سريعات لأنه استعمل اشي ثاني زي wireshark

★ HTTP Connections:-



بدنا نشوف بالأول كيف HTTP بشتغل:-
١- بده يعمل connection يبعت بكت اسمه Syn
٢- شو بجيه بكت Syn+ACK acknowladge
٣- بنطلب HTTP request
٤- يبعت packet بعد 4D
المسافة / سرعة الضوء D = d/s

أول فايل بنطلبه من سيرفر index.html بعدها بنطلب
الفايلات ثانية

http connection ↑ رجعة الي بالأسود بطلبه

- non-persistent HTTP:-

كل object بده connection لحاله. لو عندي عدة تحميلات كل تحميل بده connection
مثلاً عندي 10 Jpeg صور بدي انزلهم بكون فاتح 11 كونيكشن 10 صور + index.html 1

- persistent HTTP:-

single connection بنزل منه أكثر من object

See slide 22-24

★ كل connection بدنا له 2RTT

RTT: define time for a small packet to trevel from client to server and back.

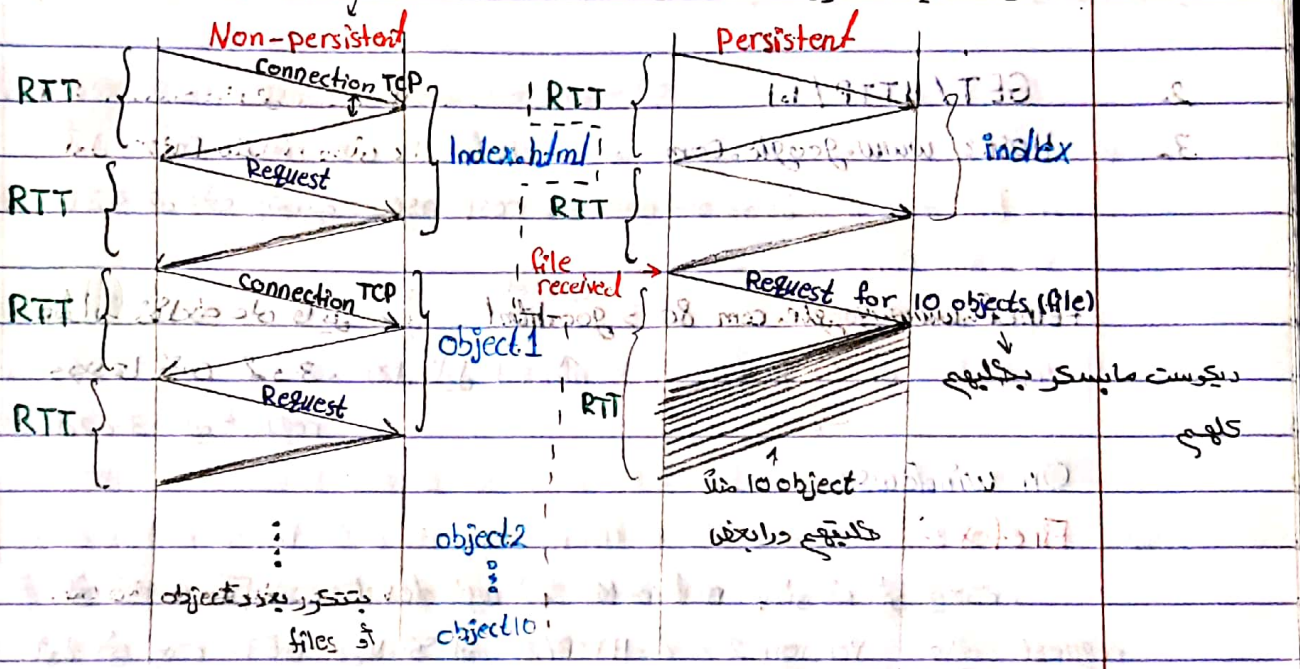
TCP ما بفتح بس الفايل
بيعت كمان
connection

slide 26-30

- non-persistent issue:-
  1- requires 2 RTTs per object
  2- OS overhead for each TCP connection
  3- browser often open parallel TCP connection to fetch referenced object.

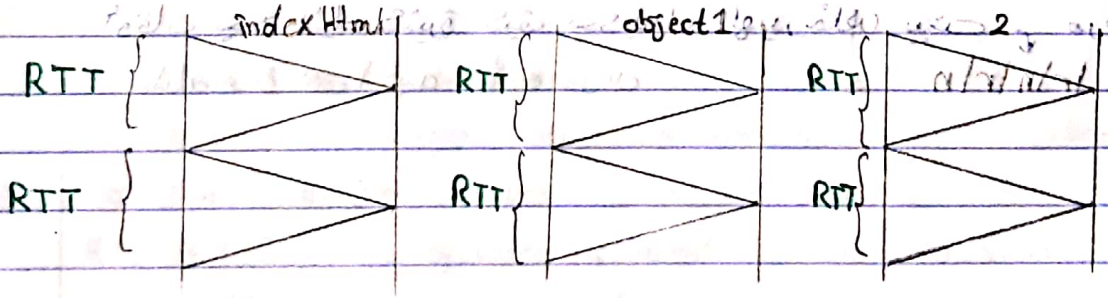
غير مستمر

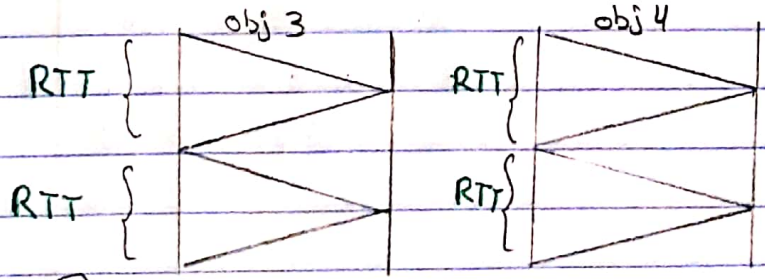
على فرض عندي 10 فايل



10 object → 20RTT + 2RTT index

* parrallel non-persistent





..... > في كمان 6 بشتغلوا كل واحد لحال لكل واحد 2RTT وفي 2RTT ← لل object 10 غير index

* كل 5 بشتغلوا مع بعض

الماكس 5 مع بعض 1 connection بس

Total RTT = 6RTT ← 2RTT ← index و 2RTT ← وال 5 كمان فكلهم 2RTT كلهم

* HTTP Request message:-

version / carriage return character ← شكله

GET / index.html HTTP/1.1\r\n ← line-feed

كبسة Enter

On linux:-

1. ~$:- telnet www.google.com 80 command

هو تول بتبعث فيها Ascii أي سيرفر بدي اياه → port 80 → http.

2. GET / HTTP/1.1

3. Host: www.google.com → أول Enter ملهاش معنى، الـ Enter ثانية هي يلي بتجيب response وبتلقي من جوجل.

telnet www.google.com 80 > goo.html ← عشان نطلعه على فايل

اسم الفايل

ويعدها نفس 2 و 3. بعمل فايل اسمه ↑

1، 2، 3 هي request

On windows:-

Firefox:-

في أشي فيه من ☰ ، web developer ← network بشوف شو بيبعث.

أول أشي بعث GET وبيلاحظ الـ HTTP/2 ← version 2 هاي request

mono:→

أوقات بحط أشياء كثيرة كيف بحطيلو الـ الهيدر خلاص بيبعث مرتين

\r\n\r\n → عشان هيك كنت أكبس enter مرتين

* Uploading from input:-

A) post method:-

في أكواد الـ html بحطها في الـ body.

B) URL method:-

use GET method.

wireshark programe to follow the request ↗ connection

احكيت قبل عنه ارجع حدد.

~o~o~o→

* Method type:-

HTTP/1.0:-	HTTP/1.1:-
GET	GET, POST, HEAD
post	PUT upload file in URL
Head	DELETE delete file in the URL

~o~o~o→

* port number:- هو زي رقم شقة في عمارة. مثلاً لما حدد بروتوكول بدي أكلم TCP server او لما بيني سيرفر بعرف port معين خاص فيه.

~o~o~o→

* HTTP response status codes:-

200 ok :- request succeeded.

301 Moved permantly:- requested move, new location specified latter in this msg.

400 Bad Request:- request msg not understood by server

404 Not Found:- requested document not found on this server

505 HTTP version Not supported.

* Cookies:-

four components:-

1- Cookie header line of HTTP response msg.

2- " " " " " " request ".

3- " file kept on end user. ↓ database بتخزنه في

Unique ID ، وبخزنه ملف عندك على جهاز وكلما أعمل اشي بخزنه في backend database for ID

4- Back end database at website.

* كمان بتستخدم في برمجة الويب

Cookies used for:-

1- authorization

2- shopping carts.

3- recommendation

4- user session state (web e-mail)

* بتظل بس مشكلة الـ privacy انه المواقع بتعرف عنك معلومات.

* من خلال الـ cookies بنقدر نخلي http ← state : بيصير عنده نوعاً ما معلومات عن اليوزر

شو بتخزنه الشركات أولاً بتبيعهم أو لا برجع الهم

L7 ## * Web Caches:- (proxy server)

الهدف منه أتطيري الـ response for client ويخفف ضغط على الـ origin server

الكلينت بيبعت الـ request بوصل أول اشي للـ cache بفحص اذا طلب موجود بيرجعه اذا لا بيبعت الطلب على origin server وهو بيرجعلي اياه مع الكاش بجيبو وبرجعلي اياه.

- cache acts as both client and server
- cache is installed by ISP.

* why caching?

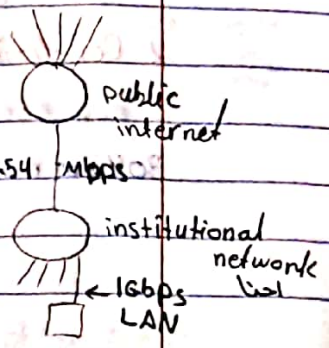
1- reduce time response.

2- reduce traffic on links.

→

3- for poor Content make it easier to request data and effectively deliver content like p2p file.

ex: object size = 100K bit , request rate browser → o.server = 15/sec
data rate to browser = 1.5Mbps , RTT from institutional router to any o.server = 2 sec , access link rate = 1.54 Mbps.
1Gbps LAN



Calculation:-

1- LAN Utilization = $\frac{\text{link rate}}{\text{LAN}} = \frac{1.54M}{1G} = 0.00154\%$

⇒ access link utilization = 97% problem

الحل:- 1- ازيد الباندويث بدل ما خليها 1.54M أخليها مثلاً 150Mbps ، بس المشكلة هاد بكلف كثير مصاري.

2- أني أعمل cache (رخيص نوعاً).

2- total delay = Internet delay + access delay + LAN delay
= 2sec + minutes + μsec micro
≈ 2.xxx sec

1- نفس المثال بس نجرب نحله على الحل الأول 1.54M ← 154Mbps

LAN utilization = $\frac{154M}{1G} = 0.154\%$

access link = 97%

delay ما أثرت على

total delay = 2sec + msecs + μsec ≈ 2.xxx sec بس حل مشكلة ازدحام داتا على link

* الحل ثاني نعمل local cache :-

Suppose cache hit rate = 0.4 → 40% request satisfied at cache , 60% origin

1- access link utilization = 60% request use access link يرجعلي سيرفر الأصلي

2- data rate to browser over access link = 0.6 * 1.54Mbps = 0.924 Mbps

utilization = $\frac{0.924M}{1.54M} = 0.584$ ⇒

total delay = 0.6 * (delay from origin server) +
0.4 * (delay when satisfied at cache)
= 0.6 * (2 sec) + 0.4 * (~msec) = ~1.2 sec

* أنسب حل وأرخص تكلفة

* Conditinal GET:-

الي بجيب أني بيجت request بعمل للكاش قبل ما الكاش يعمل response
بيبعت msg modified للسيرفر وسيرفر اذا زبطه جواب 304 not modified
يعني داتا هي هي الي عندك وديله ايلها.

واذا سيرفر برجع أنه modified يعني داتا تعدلت والكاش بعدل داتا الي عنده
ويرجعلي داتا المعدلة. برجع 200 OK

Δ * Sec 2.3:- electronic mail (SMTP, POP3, IMAP):-

Email:- official communication web.

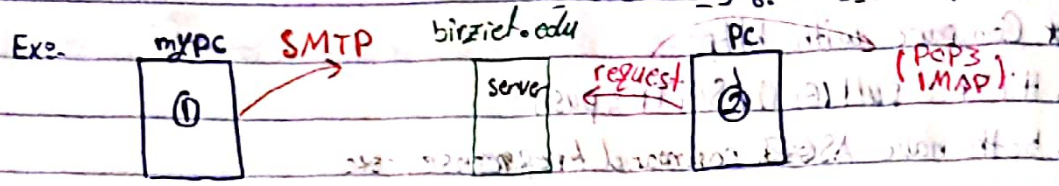
* Email contain 3 major Components:

1. User agent:- thunder bird, outlook زي الأوتلوك والي في الويندوز mail on mac
2. mail server:- mail box, message queue, SMTP protocol.
3. Simple mail transfer protocol (SMTP)

بقدر أرسل وأعدل وأقرأ المسجات عادي زي المتصفح بس الي بفرق عن المتصفحات
أنه لما بفصل نت مابقدر أشوف المسجات بس بال agent عادي. بشتغل offline
هو غابو عليه بعمل كوبي من سيرفر تيجي أو بنزلهم
المتصفحات بخليها المسجات على السيرفر.
فرق ثاني انه لو DNS مثلا خرب ، down ... الخ بضل عندي اكسيس على الايميلات
في المتصفحات لا بس في agent بس لانه بتعامل مع ip

DNS بعمل عملية تحويل من Domain ← ip Add جاينه Sec 2.4

بدي ابعت ايميل من جهازي لحدا ثاني.

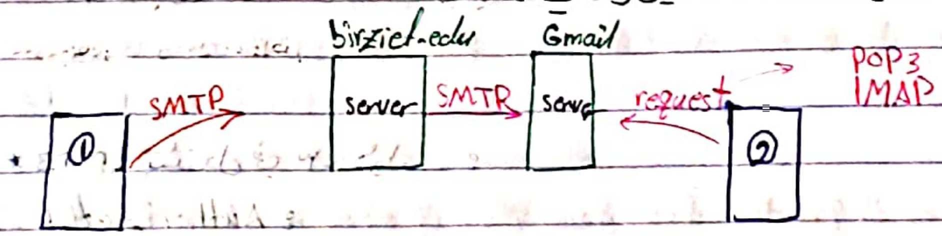


* أول اشي بدي ابعت على نفس server مثلاً birziet.edu

كتبت المسج وبعمل send بيجي دور SMTP بعمل push للايميل من كمبيوتري للسيرفر و SMTP → TCP → port 25 بوصل سيرفر بكون الي على سيرفر اشي اسمه quotta بتخزن فيها فالسيرفر بنقل الايميل من quotta تبعي الي وبحطها بال quotta تاعت الي بعتله المسج لأنا الاثنين على نفس السيرفر، زي كأنه تحويل داخلي، الشخص ثاني بقدر يشوف الايميل لما يعمل request لما يفتح الإيميل هو بعمل request لحاله وبيينه.

storge Area
Gmail 15G
ايميل جامعة
زي ايميل لأنه ايميل

* كل حدا فينا شغال على سيرفر ثاني.



نفس الاشي بعمل send و SMTP بعمل push للمسج وبخزنها في quotta إلي الي هسا BZU server راح يستخدم ال SMTP وبعمل push على سيرفر ثاني Gmail فبتخزن على ال quota تاعت شخص 2 في Gmail ولما يعمل request بشوفها.

SMTP:-

- use TCP to reliably tranfer message , port 25.
- direct transfer: sending server to receiving server.
- message must be in 7-bit ASCII

للمعرفة * to try IMSP

use telnet servername 25

see 220 reply from server

enter HELO , MAIL FORM , RCPT TO , DATA , QUIT.

* Compare with http:-
- HTTP: pull (GET) / SMTP: push
- both have ASCII command / response.
- HTTP: each object encapsulation in its own response message.
- SMTP: multiple objects send in multipart message.

~o~o~ →

1939 ← * POP3:-

pop3 = post office protocal version 3

بستخدمها عشان أشوف الايميلات لما بعمل request والها طريقتين:

1- download: أول ما أفتح نت وبعمل كونيكشن على mail server بيجيب ال pull → transaction

زي pop3 بتروح على quota تاعتي وبشوف شو المسجات الموجودة وبعملها download

على جهازي وبس يخلص وأعمل quit بفضي الكوتا بتنقل الي جواها.
اي الي بتفضى

2- Authorization: بيعطيه الإيميل والباسورد فيعرف أني quotta الي ويجيب

منها المعلومات. هاد بصير لأن سيرفر وبرجعله error أو OK

هسا لو أني مش ع جهازي مثلاً في جامعة أو الشركة وبدي أشوف الايميلات في هاي

الحالة بكون عندي ⇐ Download and Keep

لما بفتح الايميل بعمل داونلود على جهاز وما بحذفهم بتركهم، فبا يضليه أو بصير حذف

بعد فترة كثير طويلة.

→ transaction: list, retr, dele, quit
بطلع وبفضي / delete / بشوف الرسالة بقرأها / شو عندي مسج

* أنا بختار أي واحد بدي ولذا اخترت الاول لازم أتأكد بقدر أعمل backup للداتا لما احفظهم ع جهاز

فالبديل ⇐ 1730 ← * IMAP:

IMAP:- Internet Mail Access protocol

بخزنهم على النت يعني بشتغل على سيرفر، فشو ما كان جهاز هو بس بعمل connect ع سيرفر

بس بيجي أغمق اد ال quota فاضية لأنها اذا فل بيطل يوصلني مسجات.

* عندي مساحة لأسهل IMAP و سيرفر هو بعمل بالاك اب لحاله وبكون كثير منسق و ليته.

ما عندي مساحة بشتغل POP3.

ميزة في IAMP أني بقدر أخليه ينزل بس الـ header للمسجات فهي كثير مفيدة
للي بكون عندهم نت ضعيف.

هنا IAMP لأني بشتغل على سيرفر بسمحلي أني أرتب الفايلات زي ما بدي ولما بدي
انتقل بين الفايلات بكون بشكل state across وسهل أعمل mapping ... الخ .

Slide 50 :- بتورجينا كيف بنطلب وبنستخدم الإيميل على shell - لينوكس
لما نعمل send و request للإيميل.

L8:-

Δ:- Sec 2.4: DNS

DNS:- Domain name system.
مسؤول عن تحويل الأسم مثلاً (www.google.com) لـ ip Address.

- Distributed database:-
implemented in hierarchy of many (name servers).

- Application-layer protocol:-
hosts, name Servers communicate to resolve name (address/name translation).

* DNS services:-
→ hostname to IP Address.
→ host aliasing
→ mail Server aliasing.
→ load distribuation

Δ* why not make one DNS (centralize DNS)? لكل العالم.
1- اذا خرب، صار فيرسات، قطعت الكهربا، حريق ما بقدر أدخل على اي موقع إلا أكتب ip Add
2- عملية الـ update بيصير يلزمني أوقف السيرفر عشان أقدر أعدل.
3- المسافة، فراح يصير عالي كبير delay.
4- ضغط على السيرفر فما راح يقدر يلبي طلبات كل الناس.

الحل ←

* الحل لهيك مشكلة قسموا الـ DNS لثلاثة أجزاء:-

top level Domain.

1- DNS Root . 13 root

2- LTD : .com , .org , .ps , .edu ....etc

3- Authoritative. ... المسؤول عن المواقع المعينة زي جوجل، فيس ...

بشكل عام دوما في local DNS ← أنا بطلب موقع مثلاً www.google.com بطلبه من root server يلاقي DNS server الخاص في .com وبعدها ببحث فيه عن google.com لما يلاقيها بطلب يرجعلي الـ ip Address تاع جوجل

* local DNS name server:

سيرفر يكون موجود في الشركات أو جامعات وهو بعمل caching .

How it's work:- بدور على ريتاج

1 iterated query — بيسأل أول اشي الـ local DNS اذا عنده IP Add. برجعلي اياها ما عنده بروح بيسأل DNS root اذا عنده برجعلي اياه بحكيه لل local اذا ما عنده برجعله TLD DNS بيسأل TLD DNS الخاص مثلاً بـ edu. اذا عنده برجعلي اياه بيرجع Authoritative DNS .edu التي هو birzeit ويسأله اذا عنده بيرجع لل local الـ ip Addr تاع الموقع الي بدور عليه rital ولل local برجعلي اياه

(جوال ريتاج ritaj ...)

* تقيل شغل على root ويلاقي DNS عن طريق الـ cache .

2 Recursive query

بسأل local ← local بسأل root ← root بسأل TLD ← TLD بسأل Authoritative

و Authoritative يرد على TLD ← TLD يرد على root ← root يرد على local ← local ← أنا .

Slide 64-65

* طبعاً العمليتين (1 أو 2) بوخذو وقت طويل ويصير delay أو ممكن error

الحل عملية ← caching

* بنحط في cache ← (TTL) ← time to live

* اذا صار out of date مثلاً تغير الـ ip لموقع ما في مشكلة بصير delay

متغير لحد ما يعمل update اذاتا .

* DNS records:-

DNS:- distributed database storing resource records (RR)

RR format: (name, value, type, ttl)

type = A
- name is host name. (google.com, 192.168.11, A)
- Value is IP Address.

type = NS names Server
- name is domain e.g google.com (google.com, NS)
- Value is hostname of authoritative name

type = CNAME canonical
- name is alias name for some real name.
- Value is canonical name

type = MX mail exchange
- Value is name of mailserver associated with name.

(gmail.com, IPmail, MX)

See Slide 68

* DNS → query UDP
query and reply

- Id:- 16 bit for query and 16 bit for reply 2byte
- flag:- query or reply, recursion desired, recursion available 2byte
  reply is authoritative
- # question, # answer RRs 2byte
- # authority RRs, # additional RRs 2byte

2 byte - Question (variable # of questions): name, type fields for a query. بخزنه

//// - answers (// // // RRs): RRs in response to query.

// - authority (// // //): records for authoritative servers.

- additinal info (// //): additional help info that maybe used

See Slide 70-71

* Attacking DNS:-

DDoS attacks:- slide 73

أحكينا قبل عنه انه أكثر من جهاز ببعتوا request (query) كثيرة وبتخلي سيستم يوقع

DoS:- Denial of Server , D:- distributed (الأولى) توزيع

مش كثير مهم

Δ* Sec 2.5:- P2P applications:-

ما عنا سيرفر بوزعل يتكل من جاي حول ، فبنعتمد على تخزينا الملفات فبنخزنهم في مكانه وبنعمل مشاركة بلهي بجير client , server بنفس الوقت .

* من ايجابياتها أنا ما بنحكي مع سيرفر ، دايركت مع شخص ثاني ،

* // سلبياتها أنه ال peers مش دايمًا connected و ال change IP Address

Ex:- torrent , Skype , streaming (Kankan)
file distribution VoIP

* File Distribution:- Client - Server VS P2P

سؤال:- كم بدنا وقت لنطلع الفايلات (size (F)) من server ل peers (يوزرز) N ؟

C-S →

1) • Server transmission:- for every file need F/Us

Us:- upload speed

- time for N copies: NF/Us

2) client: each client must make download.

di/dmin = min client download rate.

min client download time: F/dmin

⇒ time to distribute F to N clients using client-server approach

$$D_{c-s} \geq \max\{NF/u_s, F/d_{min}\}$$

Ex: 1M byte File Size, Us=100 Mbps, D1=10 Mbps, D2=3Mbps

Dmin=100kps, we have 100 user ⇒ find Dc-s?

NF/Us = 100*1M*8/100M = 8 sec

F/ds = 1M*8/100k = 80 sec

$$D_{c-s} \geq \max\{8, 80\} \Rightarrow D_{c-s} = 80 \text{ sec}$$

* File distribution time: P2P

1- Server transmission: time = F/Us

2. client: download file ⇒ time = F/dmin

3- clients: as aggregate must download NF bit

max upload rate is: $u_s + \Sigma u_i$; u_i: client upload

⇒ time to distribute F to N client using P2P approach

$$D_{p2p} \geq \max\{F/u_s, F/d_{min}, NF/(u_s+\Sigma u_i)\}$$

slide 97 درسة

L9:-

* P2P file distribution:- Bit Torrent   حالة Illegal

torrent:- group of peers exchanging chunks of file.

- The idea of bit torrent is divided file into 256Kb chunks.
- peers in torrent send/receive file chunks.

* بالمختصر بدل ما أطلب من السيرفر يبعتلي الفايل كامل عشان أنزله، التورنت بقسمه لفايلات صغيرة كل فايل 256K byte.

* يكون في عدة ماشينز (أجهزة) مشبوكين مع بعض بدون سيرفر وراح يصير عملية tracker chunks . هاي العملية اسمها تورنت.

في كل تورنت بكون اسمها tracker فيها ← list of id's for peers

tracker:- tracks peers participating in torrent.

* أنا بدي أنزل فايل بعمل request لل tracker تاع torrent بضيف ip تاعي على الليسته ويبعتلي ال ip's للكمبيوترات الي عندها chunks وجهازي بحاول يعمل connected مع هدول ال ip's اذا نجح بصيروا neighbors.

نوع الكونيكشن بين ال peers أ (TCP)

* لما جهازي يصير عنده عدد chunks كافي بصير يقدر يعمل upload

← لما أخلص عملية داونلود بإمكاني أظل في تورنت أعمل داونلود أو أبلود أو أني أطلع ويقدر أغير كمان ال peers الي شايلة لأنهم بدخلوا وبطلعوا وهدول بنسميهم Churn

* Two Big Decisions:-

1- which chunks should we ask for first?
2- To which peers should we send requested chunks?

عشان أحل المشكلة الاولى بستخدم تكتيك اسمه:-

→ Rarest first:-

بطلب من كل نيبر بير يبعتلي شو عنده chunks فبصير عندي list of chunks لكل واحد من nighbor peer، بعدها بحسب أني ليسته فيها أقل chunks وبطلبها للأول.

→ Tit For Tat:- عشان نحل المشكلة ثانية بنستخدم ←

* أعطيني بحطلك كل واحد حسب الكل واحد من الـ nighbord peer ← rate ← rate(chunk/s)
علشان يبعتولي وبعدها بختار أكثر 4 وبيعتلهم chunks.
الـ 4 الي بختارهم بصير أسمهم Unchoked peers والباقي Choked peers.

استخدام هاد تكنيك بحلي مشكلتين.
1- اذا واحد دخل على تورنت جديد فش حدا راح يحل معاه tit for tat لأنه فش عنده ولا chunks
2- لما أختار 4 راح يبعتولي هاد في السايكل الأولى طيب في ثانية بما أنهم بعتولي راح أبعتلهم وهيك راح نضل لحد ما يخلص الداونلود طيب والنيبر ثانيين ؟!

الحل لـ (2) بنستخدم تكنيك ثاني :- → Random Neighbor Selection
كل 30s بختار واحد من نيبر نبعتله chunks ، مثلاً بنبعت وبنستقبل منه
في حال كان عندي rate عالي راح يحطني في top 4 عنده ونفس الاشي عندي
وبتحل مشكلة اذا واحد دخل جديد لأنه بصير عنده شوي شوي chunks وبصير يقدر يشارك .

←o~o~o~o~o→

A:- Sec 2.6 :- Video streaming and Content distribution network (CDNs):-
Video streaming like youtube, Netflix :- 37% , 16% downstream
داتا رايت عالية عندهم كثير، ويكون في ضغط كبير فراح يكون عنا تحديين
1- Scale :- how to reach 1B users?
عشان يكون ضغط كبير لليوتيوب 1B user و نيتفليكس 75M بلزمني سيرفر قوي يوصل لـ 1 mega وهاد شبه مستحيل فبعملوا distribute سيرفرات بالعنوانات كبيرة ومحدودة أو بالالوف جس صغيرة .

2. Heterogeneity:-
أن كل يوزر بكون اله قدرة نيرعه ذاتي ، يعني عنا مش زي الي في أمريكا .
كل حدا اله datarate وبجد مختلف عن سيرفر ← أوحتى 3g ، 4g ، ethernet
الحل :- low quality video أو application - level infrastructure
أحمل كل شي بدها hd حسب الداتا رايت الي عندي بحطني جودة

* See slide 87 + 88 just read.

• DASH:-

حل لمشكلة الي لما أطلب فيديو من اليوتيوب

Dynamic Adaptive Streaming over HTTP.

فكرتها أن الفيديو نعمله أكثر من نسخة وكل نسخة على bit rate مختلفة كل بت بتعطي كوالتي معينة عشان نخدم كل الكلاينت حسب نت تاعه وكل نسخة من الفيديو بتتخزن على chunks.

Server:-

• divides video file into multiple chunks.

• each chunk stored, encoded at different rates. كلها مخزنة في

• manifest file:- provides URLs for different chunks.

احنا بس بنستدعي ال manifest file وهو من خلاله بيجي على الي بتناسبني والأوقات

وهو بعطينا على حسب المسافة بينا وبين أقرب مسافة لسيرفر rate

ويعطيني عدة rate وأنا بختار.

* Content distribution networks:-

Challenge:- how to stream content (selected from millions of videos) to thousands of simultaneous users?

Option 1:- Single, larg "mega-server"

المشكلة:

1- Single point of failure.

إذا وقع سيرفر كل الفيديوهات بروح ولا أحد بقدر يوصل للفيديوهات.

2- point of network congestion:- عدد اليوزر بوصلوا مليار فبصير تأخير و ديلاي عالي كثير

3- long path to distance client → سيرفر بعيد وأنا

4- multiple copies of video sent over outgoing link.

لو مليون واحد طلب نفس الفيديو بده يضل يحل كوبي كوبي بصير ضغط ويكلف كثير.

Legendary Love

⇒ Option 1 doesn't scale

Option 2:- Store / server multiple copies of videos at multiple geographically distributed sites (CDN) → الي بدنا اياه

CDN:- Content Distribution Network.

2 way to make CDN:-

1- enter deep:-

أني بحط سيرفرات قريبة على اليوزرز، زي Akamai عندها 1700 location، بنشر سيرفرات بالعالم. عددها بالألوف في نفس البلد

2- bring home:-

عددها بال (10's) أني بحط سيرفر قوي كثير في نقطة معينة والكل الي عندها مسافة معينة بشبك عليها زينا لما بنشبك على سيرفرات الي في ألمانيا، وهاي أوفر بس أبطأ

~o~o~o→

مثلاً بدي أحضر فلم على النت فليكس، أنا بسأل نت فليكس هو بعطيني ال manifest file بشوف مين أقرب سيرفر الي وبجيب منه الفلم

★ Over the top:

أنه بوزع الفلم على سيرفرات، وهيك، راح نحكي عنه بعدين

~o~o→

تحليه بشكل أقرب بس بدي فلم بسأل الـ DNS الي عنده شو الـ ip Add للفيديو فهو بروح بسأل الـ DNS تاع الشركة نفسها، الـ DNS تاع شركة برجعلي ال ip تاع DNS الي فيه الفيديو، فبروح عليه وبنهاية datatransfer بتصير بين جهازي وبين الفلم.

أو CDN ... بعطيني وين ال ip تاعه يوديني في CDN

*كل يوزر راح يجيه موقع غير عن الثاني حتى لو طلبوا نفس الفلم حسب المنطقة وشو الـ DNS الشركة بعطيني.

Δ→ Sec 2.7:- Socket programming with UDP and TCP:-

Δ:- Socket programming:-

نفرض في عندي يوزر بدو يفتح web مع جوجل، جوجل الـ web server تاعو على port 80
اليوزر شو البورت تاعه مش 80 بكون رقم كبير زي 5000 ، 16000 ... الخ
وكلما أفتح newtap بفتح socket مختلفة عن الثانية، حتى لو فتحت جوجل مرتين
كل وحدة غير لما أعملت الـ request.

* كل socket فيه رقم مميز:-
ip source, ip distination, port for source, port for distination.
destination

Socket:- door between application process and end-end transport protocol.
عبارة عن انترفيس بين الـ application layer و الـ transport layer.
أو هو برنامج بوخذ داتا بودّيها عن طريق TCP أو UDP.

الهدف منه أنه: build client/server applications that communicate using Socket.

UDP:- unreliable datagram.
TCP:- reliable, byte stream-oriented.

* Application Example: (تطبيق)
1- Client reads a line of characters (data) from its keyboard and sends data to server.
2- Server receives the data and converts characters to uppercase.
3- Server sends modified data to client.
4- client receives modified data and displays line on its screen.

* Socket programming with UDP:-

ما بتعمل كونيكشن مع السيرفر لما (handshaking) ما بعمله أي أفحص الكونيكشن وأتأكد من أني اتصلت مع جهاز ثاني.

فبعت البكيت بكون بلزمني IP source و IP destination و port #

لما بوصل رسيفر بشوف هدول وبعمل extract وبجيب البكيت ... الخ

note: UDP transmitted data maybe lost or received out of order

ممكن داتا تضيع أو توصلني مخربطة.

* كيف بتبرمج:-

الكلينت بعمل Socket والسيرفر بعمل Socket ، الكلينت ببعث البكيت بوصل للسيرفر ← سيرفر بعمله receive للبكيت لما يبعتها الكلينت وتوصله وبعدها بعمل converging وبرجع يبعتها للكلينت

Code:- UDP client

```
from Socket import *
Servername = "localhost"
Serverport = 12000
client Socket = Socket(AF_INET, SOCK_DGRAM)
```

AF_INET ← Address Family internet (كان في أشياء غير IP)

SOCK_DGRAM ← لأنها UDP

```
message = raw_input("Input lowercase Sentence:")
clientSocket.Sendto(message.encode(),(serverName, serverport))
modifed Message, Server Address = clientSocket.recvfrom(2048)
print modifedMessage.decode()
ClientSocket.close()
```

لو علقنهم كومنت وشغلت مارح يعطي ايرور ولا يتأكد لأنه بس يبعت

* اذا عملت رن ودخلت اي اشي بحوله لـ Uppercase.

لو عملت print serverAddress ← ('127.0.0.1', 12000)

* كود سيرفر نفس الاشي ما أختلف See Slide 103

* لازم الكلينت والسيرفر الاثنين نفس البورت serverport

* من 10 ← 8000 محجوزات الـ ports ما بقدر استخدمهم .

* Socket programming with TCP:-

* أول أشي لازم سيرفر يكون شغال وبعدها لازم سيرفر يفتح باب (socket) عشان يعمل كونيكشن ، بعدها الـ client بعمل socket وبحدد IP Address و port # of server

وبعدها الكلينت بشبك مع socket of server وبعمل Data transfer

Code:- TCP client

ما اختلف عن الي قبل بس سطر (4) حطينا SOCK_STREAM يدل على TCP

بدل كلمة SOCK_DGRAM للـ UDP

لو جيت أشغل الكود ما راح يزبط لازم أكون مشغل السيرفر قبل .

ونفس تغير عن UDP ↑

See slide 102, 103 UDP codes

See slide 106, 107 TCP codes

السلايدات "..." chapter 2 V7.02

أو ← 99, 100 / 103-104

End chapter 2