שם: _____ רשתות מחשבים – דף עזר

מודל 5 שכבות

#	שכבה	פרוטוקול	יחידה
5	אפליקציה	HTTP, DNS, SMTP	Message
4	תעבורה	TCP, UDP	Segment
3	רשת	IP, ICMP	Packet
2	קו	Ethernet, ARP	Frame
1	פיזית	Bits	Bit

DNS

היררכיה

Root (.) → **TLD** (.com, .org, .il) → **Domain** (google.com) → **Subdomain** (www.google.com)

שם מלא: `www.google.com.` (נקודה בסוף = root)

סוגי רשומות

סוג	משמעות
A	שם דומיין → IP
PTR	IP → שם דומיין (הפוך)
NS	Name Server של domain
CNAME	alias לשם אחר
MX	Mail server

פעולה

- פורט **53 UDP** עובד מעל DNS
- שרת DNS שאילתה → מחזיר IP
- כל רשומה יש TTL (זמן מטמון)
- כלי: `nslookup www.google.com`

כתובות – IP

מבנה IPv4

4 `192.168.2.5` | דוגמה: 0–255 | כל בית: (32 ביט) בתים

סה"כ כתובות: $2^{32} \approx 4.3$ מיליארד

Subnet Mask

`192.168.0.1/16` → מזהה ישות: `192.168` | מזהה רשת: `0.1`

ניתן לכתוב: `255.255.0.0`

כתובות מיוחדות

כתובת	משמעות
`127.0.0.1`	Loopback / Localhost
`0.0.0.0`	כל כתובת (Any)
`255.255.255.255`	Broadcast

כתובות פרטיות

טווח	כמות
`10.0.0.0/8`	~16.7M
`172.16.0.0/12`	~1M
`192.168.0.0/16`	65,536

כתובות פרטיות **לא ניתנות לניתוב** באינטרנט

NAT (Network Address Translation)

NAT: אזלו. **פתרון** IPv4 **בעיה:** כתובות

תהליך:

1. מחשב פנימי (`192.168.0.1`) שולח חבילה
2. נתב מחליף כתובת מקור → כתובת ציבורית שלו (`1.1.1.1`)
3. שרת חיצוני רואה רק כתובת הנתב ועונה לנתב
4. נתב מחליף כתובת יעד חזרה → מחשב פנימי

תוצאה: כל הרשת משתמשת ב-IP אחד ציבורי

TCP vs UDP

מאפיין	TCP	UDP
קישור	✅ Connection-oriented	❌ Connectionless
אמינות	✅ מובטחת	❌ לא מובטחת
סדר	✅ מובטח	❌ לא מובטח
מהירות	🔴 איטי יותר	🟢 מהיר יותר
Header	20+ בתים	8 בתים
שימוש	HTTP, FTP, SSH	DNS, Streaming

UDP Header שדות:

Source Port | Dest Port | Length | Checksum

TCP 3-Way Handshake

```
Client → Server:  SYN   (seq=x)
Server → Client:  SYN+ACK  (seq=y, ack=x+1)
Client → Server:  ACK   (ack=y+1)
✅ חיבור מוקם!
```

Sequence Numbers: מספור חבילות לסדר נכון

ACK Numbers: אישור קבלה של כל חבילה

שקע (Socket) TCP – Python

שרת:

```
s = socket.socket()
s.bind(('0.0.0.0', PORT))
s.listen(1)
conn, addr = s.accept()
data = conn.recv(1024)
conn.send(reply.encode())
```

לקוח:

```
s = socket.socket()
s.connect(('127.0.0.1', PORT))
s.send('msg'.encode())
data = s.recv(1024)
```

שקע UDP – Python

שרת:

```
s = socket.socket(socket.AF_INET,
                  socket.SOCK_DGRAM)
s.bind(('0.0.0.0', PORT))
data, addr = s.recvfrom(1024)
s.sendto(reply.encode(), addr)
```

לקוח:

```
s = socket.socket(socket.AF_INET,
                  socket.SOCK_DGRAM)
s.sendto('msg'.encode(),
         ('127.0.0.1', PORT))
data, addr = s.recvfrom(1024)
```

מצבי חיבור (netstat)

State	משמעות
LISTENING	ממתין לחיבור
ESTABLISHED	חיבור פעיל

`netstat -a` → כל החיבורים כולל LISTENING

פורטים שמורים (Well-Known Ports 0–1023)

פורט	פרוטוקול	שכבת התעבורה
20/21	FTP (data/ctrl)	TCP
22	SSH	TCP
23	Telnet	TCP
25	SMTP (email)	TCP
53	DNS	UDP (בד"כ)
67/68	DHCP	UDP
80	HTTP	TCP
443	HTTPS	TCP
8080	HTTP alt	TCP

מחליטה על פורטים 0-1023. פורט = מזהה אפליקציה (כמו IANA מספר דירה)

Scapy – פקודות

הרצה + ייבוא

```
$ scapy                          # הרצה אינטראקטיבית
from scapy.all import *          # ייבוא בסקריפט
```

הסנפה – sniff()

```
# הסנפת 2 חבילות
packets = sniff(count=2)

# עם פילטר ופונקציה על כל חבילה
packets = sniff(count=10,
    lfilter=filter_func,
    prn=process_func)

# lfilter - פונקציה שמחזירה True/False
def filter_dns(pkt):
    return (DNS in pkt and
            pkt[DNS].opcode == 0 and
            pkt[DNSQR].qtype == 1)

# prn - פונקציה שרצה על כל חבילה
def print_query(pkt):
    print(pkt[DNSQR].qname)
```

יצירת חבילות

```
# IP בסיסי
p = IP()
p = IP(dst='10.1.1.1', ttl=6)

# שכבות - משתמשים ב /
p = IP() / TCP()
p = Ether() / IP() / TCP()
p = IP() / UDP()
p = Ether() / IP() / TCP() / Raw("GET / HTTP/1.0\r\n\r\n")

# עם פרמטרים
p = Ether() / IP(dst='1.2.3.4', ttl=10) / TCP(dport=80)

# Resolving אוטומטי דומיין → IP
p = IP(dst='www.google.com')
```

צפייה בחבילה

```
p           # תצוגה מקוצרת
p.show()    # כל השדות
p.src       # כתובת מקור
p.dst       # כתובת יעד
p[TCP].dport  # שדה ספציפי בשכבה
```

שליחת חבילות

```
send(p)       # שליחה ברמה 3 (IP)
sendp(p)      # שליחה ברמה 2 (Ethernet)
sr1(p)        # שלח + קבל תשובה אחת
```

נפוצות Scapy שכבות

שכבה	Object	שדות חשובים
2 (קו)	`Ether()`	src, dst (MAC)
3 (רשת)	`IP()`	src, dst, ttl
4 TCP	`TCP()`	sport, dport, flags
4 UDP	`UDP()`	sport, dport
3 (ICMP)	`ICMP()`	type, code
5 DNS	`DNS()`	opcode, DNSQR
raw data	`Raw()`	load

בדיקת תוכן חבילה

```
DNS in packet         # האם יש שכבת DNS?
IP in packet          # האם יש שכבת IP?
packet[DNS].opcode    # שדה opcode בDNS
```

ICMP

Internet Control Message Protocol – פרוטוקול ניהול/בדיקת רשת, ב-Layer 3 (מעל IP)

Header: 8 בתים קבועים: Type | Code | Checksum | Data

Type	Code	משמעות
8	0	Echo Request (ping)
0	0	Echo Reply (pong)
11	0	TTL Exceeded (traceroute)
3	*	Destination Unreachable

Ping: Echo Request שולח, Echo Reply מקבל. בודק קישוריות.

Traceroute: שולח חבילות עם TTL=1,2,3... כל נתב מקטין TTL ב-1. כשמגיע ל-0 שולח ICMP TTL Exceeded → גלה כתובת הנתב

IP Header – שדות חשובים

שדה	תיאור
Version	4 (IPv4)
TTL	Time To Live – קטן ב-1 בכל נתב, נזרק ב-0
Protocol	6=TCP, 17=UDP, 1=ICMP
Src IP	כתובת שולח
Dst IP	כתובת יעד
Checksum	לזיהוי שגיאות

שימושיים CLI כלי

פקודה	שימוש
`ipconfig / ifconfig`	כתובת IP מקומית
`ping <host>`	בדיקת קישוריות (ICMP Echo)
`tracert / traceroute <host>`	מסלול חבילה ברשת
`nslookup <domain>`	DNS שאילת
`netstat -a`	כל פורטים פתוחים + חיבורים
`route print`	טבלת ניתוב מקומית

רכיבי רשת

רכיב	שכבה	תפקיד
Hub (רכזת)	1	מעביר לכולם (Broadcast)
Switch (מתג)	2	לומד כתובות MAC, מעביר לפי
Router (נתב)	3	מחבר רשתות IP, מנתב לפי

Default Gateway: נתב שאליו נשלחות חבילות שלא ברשת המקומית (כתובת 0.0.0.0/0 בטבלת ניתוב)