








An abstract graphic featuring two concentric rings. The left ring is blue and the right ring is green, both with a slight gradient and a soft shadow effect. They are set against a light gray background.

Sockets

# Sockets

TCP/IP			OSI Model	Data Unit	Hardware	Protocols
Application	7		<u>Application</u>	Data	Gateway	S/MIME, SMTP, SNMP, HTTP, LPD, FTP, TFTP, Telnet, POP, SMB, NNTP, CDP, GOPHER, NDS, AFP, SAP, NCP, SET
	6		<u>Presentation</u>	Data	Gateway Redirector	No protocols. Service: TIFF, GIF, JPEG, ASCII, MPEG, MIDI, EBCDIC, POSTSCRIPT - Ensures confidentiality, Compression/Encryption
	5		<u>Session</u>	Data	Gateway	SOCKS, RPC, NFS, SQL, NetBIOS, RADIUS, DNS, ASP - Doesn't ensure security CORBA, DCOM, SOAP, .NET
Host-to-host	4		<u>Transport</u>	TCP-Segments UDP-Datagram	Gateway	<u>TCP</u> , <u>UDP</u> , SSL/TLS, SPX, SSH-2, ATP
Internet	3		<u>Network</u>	Packets	Router Router	IP, IPSec, ICMP, IGMP, RIP, OSPF, BGP, IPX, SKIP, SWIPE, NAT, IGRP, EIGRP, BOOTP, DHCP, ISIS, ZIP, DDP
Network access	2		<u>Data Link</u> LLC MAC	Frames	Switch, Bridge, NIC	3thernet 802.3, Token Rin5 802.5, ATM, FDDI 802.4, Wi-Fi 802.11, PPP, L2TP, SLIP, ARP, RARP, 802.1AE MACSec, HDLC
	1		<u>Physical</u>	Bits	Repeater, Hub,NIC, Cables,MAU Multiplexer	ISDN, DSL, SONET, 10BASE-T, 10BASE2, 10BASE5, 100BASE-TX, 100BASE-FX, 100BASE-T, 1000BASE-T, 1000BASE-SX, EIA-x, RS-x

# Transport lagret? (Sender)

Datorer/sensorer  
på ett nätverk

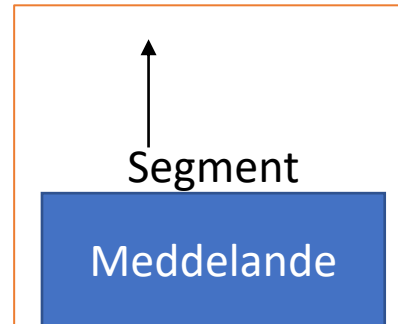
Port 13401

Port 13402

Port 13403

Port 13404

**Källa:** port 13401  
**Destination:** port 80



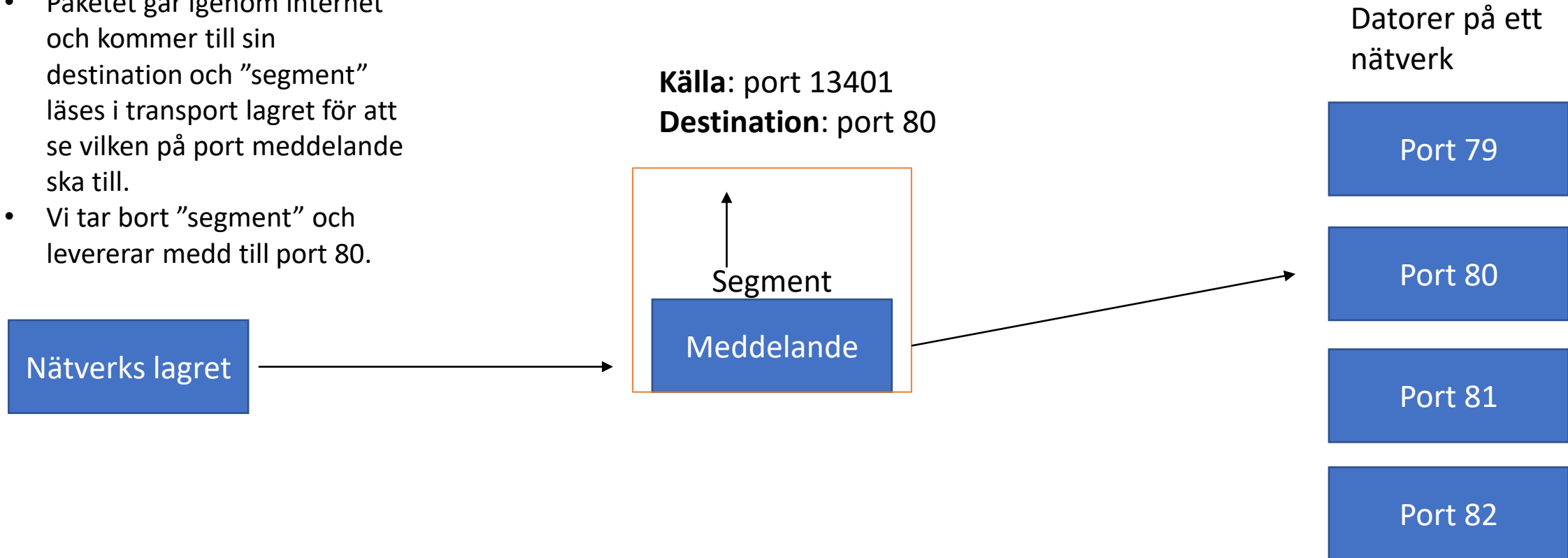
Nätverks lagret



- När vi skickar ett paket i vårt interna nätverk så klär vi den med något som heter "Segment"
- I "Segment" så lägger vi till yttligare info (**Källa & Dest...**)

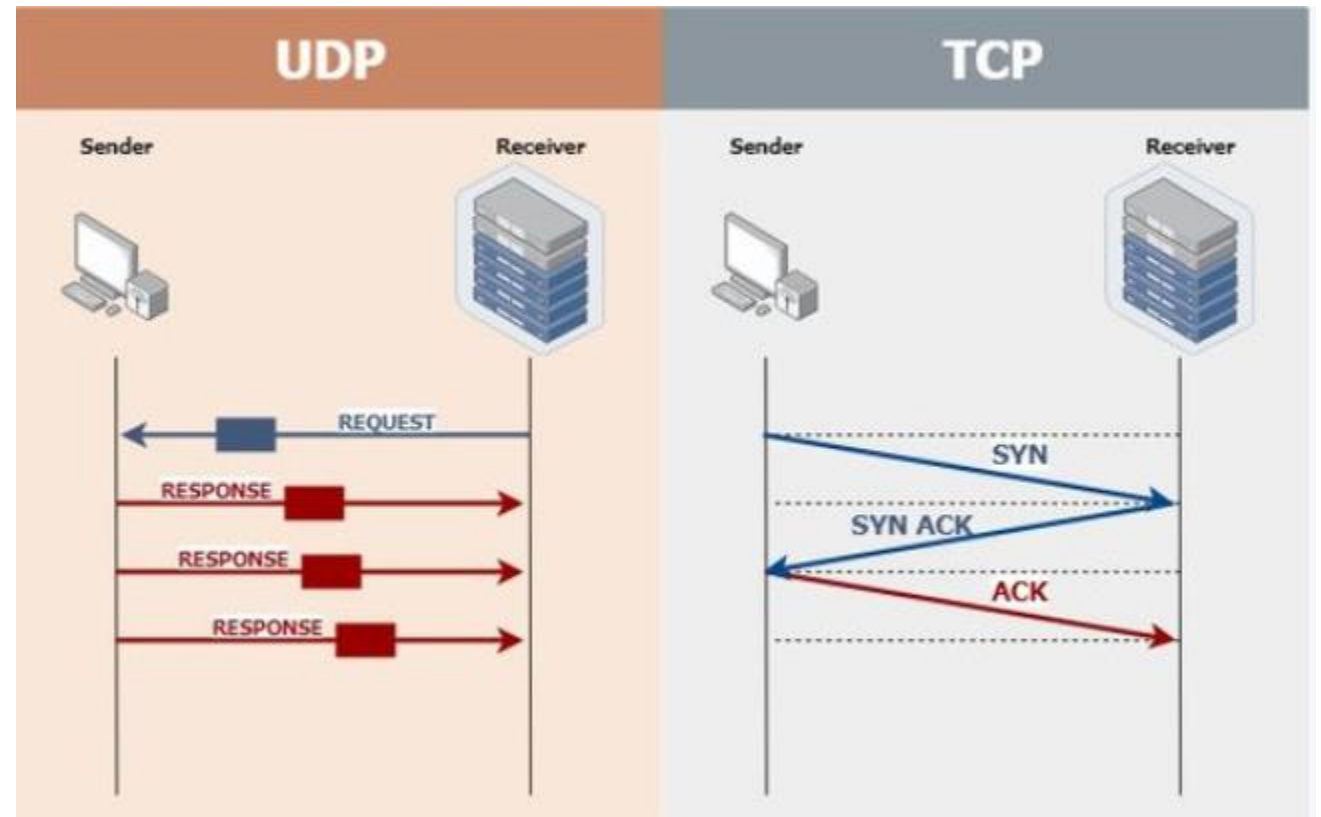
# Transport lagret? (Receiver)

- Paketet går igenom internet och kommer till sin destination och "segment" läses i transport lagret för att se vilken på port meddelande ska till.
- Vi tar bort "segment" och levererar medd till port 80.



# Transport lagret? (UDP & TCP)

- Vi kan alltså sända våra paket i UDP eller TCP.
- Så om ni kommer ihåg från förra kursen så är TCP ett mycket säkrare protokoll. Speciellt om vi ska skicka filer eller meddelande



# Sockets



- En socket är slutpunkt av en två-vägs kommunikation länk som körs på ett nätverk.
- En socket är bunden till port nummer så att TCP lagret kan indentifera applicationen som datat skickas till.
- En slutpunkt(endpoint) är en kombination av IP-address och port nummer
- Så en socket öppnar upp ett lager/layer av kommunikation.

# Socket kommandon i Python

- `s = socket.socket()` **TCP**
- `socket.socket(socket.AF_INET, socket.SOCK_DGRAM)` **UDP**
- `s.bind(host,port)`
- `s.send()`
- `s.listen()`
- `s.recv()`
- `s.close()`

# 1

- `socket.socket()`

Den rad som skapar en  
socket.



## 2

- `s.bind(host,port)`

Sedan måste vi binda IP adress och port till socket.

Host = IP adress

# 3

- `s.send()`

Sedan skickar vi ett  
meddelande med `s.send()`

# 4

- s.listen()

Den andra datorn så använder vi s.listen() metoden.

Betyder att vi "lyssnar" och väntar på ett meddelande

# 5

- `s.recv()`

När vi väll fått ett  
meddelande så måste vi  
decode medd.

# 6

- `s.close()`

Och detta för att stänga  
kommunikation.

# Sockets

