Lektionstillfälle 9

Nätverk i Linux

med Mikael Larsson

NACKADEMIN

Återblick

Förra lektionen gick vi igenom pakethantering och hur man håller systemet uppdaterat.

Vi installerade paket från Ubuntu och andra källor med ”apt”, installerade från nedladdade arkiv och från källkod.

Dagens lektion

Mål: Att konfigurera grundläggande nätverksalternativ och felsöka nätverksanslutningar.

- Protokoll, adresser, portar och tjänster
- Statisk och dynamisk ip
- Routing
- Felsökning (ping, nc, netstat, tcpdump)
- Linux inbyggda brandvägg – iptables
- **Mittkursutvärdering**

Termer och begrepp

Lager och protokoll
MAC-adress
ip-adress
host och domännamn
nfs

Nätverk i Linux

TLCL kapitel 16

Vi kommer att prata om protokollet ”ip”, eftersom det är det som i stort sett allt nätverkande handlar om numer.

Vi kommer också bara prata specifikt om ”ipv4”.

När man pratar om nätverk och protokoll behöver man hålla reda på olika ”lager”.

Ni behöver inte veta vad lagren heter, mest att de finns, och att varje lager bygger på lagret under för att tillföra funktionalitet.

https://en.wikipedia.org/wiki/OSI_model

Nätverkslager

Varje lager har olika familjer av protokoll och tillhörande adressering.

Här är de lager ni behöver känna till, med exempel på protokoll och ”adresser”:

REST, GraphQL	Path, POST - data
http, ssh, dns	url, hostname
tcp / udp / icmp	port
ip	ip-adress
MAC / Ethernet	MAC-adress

Det är de tre nedersta lagren vi kommer titta på idag.

Interface, statisk och dynamisk ip

Närmast ”sladden” sitter nätverksinterface / adapter.
Listas med:

```
$ ip link
```

Alla system har en loopback-adapter, ”lo”.
Övriga adapters har lite olika namn, beroende på drivrutin, teknik, m.m. Ethernet-adaptrar brukar börja på ”en”.

ip-adresser kan sättas statiskt eller dynamiskt med dhcp.

I Ubuntu används ”netplan” för att sätta upp nätverksadaptrar och adresser.

https://ubuntu.com/server/docs/network-configuration

Decimala tal…

Vårt decimala talsystem är ett positionsbaserat system, där varje position kan ha 10 olika värden.

Vi säger att **basen** i talsystemet är 10, och de olika tecken vi använder för att representera de tio olika värdena är 0-9.

Vi vet att positionerna representerar ental, tiotal, hundratal och så vidare från höger till vänster.

Se exemplet för talet 256 i basen 10.

256_{10}

$10^2 = 100$	$10^1 = 10$	$10^0 = 1$	
*	*	*	
2	5	6	
=	=	=	
200	50	6	= 256

binära tal…

Binära tal har istället **basen** 2, och har alltså behov för två olika tecken. Vi använder 0 och 1 som binära siffror, **bitar**.

Positionerna representerar för binära tal ental, tvåtal, fyrtal, åttatal, sextontal, osv.

Se exemplet för talet 101 i basen 2.

101_2

$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	
*	*	*	
1	0	1	
=	=	=	
4	0	1	= 5

…och hexadecimala

Hexadecimala tal har istället **basen** 16, och har alltså behov för sexton olika tecken. Då räcker inte 0-9 till, utan vi använder också A-F.

Positionerna representerar för hexadecimala tal ental, sextontal, 256-tal, 4096-tal, osv.

Se exemplet för talet 12E i basen 16.

$12E_{16}$

$16^2 = 256$	$16^1 = 16$	$16^0 = 1$	
*	*	*	
1	2	E	
=	=	=	
256	32	14	= 302

Hex, binärt och 2-potenser

4 bitar har 16 värden, alltså går det 4 bitar på en hexsiffra:

$$1111_{2} = 15_{10} = F_{16}$$

8 bitar motsvarar då två hex-siffror:

$$11111111_{2} = 255_{10} = FF_{16}$$

I digitala sammanhang kallas en lagringsenhet på 4 bitar en **nibble**, och en på 8 bitar en **byte**.

Ett 32-bitars tal tar 4 bytes att lagra, ett 64-bitars tal 8 bytes.

Bra att kunna / känna igen:

Bitar	1	2	3	4	5	6	7	8
Värden	2	4	8	16	32	64	128	256
Hex	2	4	8	10	20	40	80	100

Bitar	9	10	11	12	13	14	15	16
Värden	512	1024	2048	4096	8192	16536	32768	65536
Hex	200	400	800	1000	2000	4000	8000	10000

Nätmask och subnät

En ip-address är ett 32-bitars tal som vi oftast skriver uppdelat i fyra bytes, alltså fyra tal mellan 0 och 255, med punkter emellan, ”192.168.2.25”.

Med hjälp av nätmask avgränsar man det giltiga adressutrymmet för ett nätverk. Masken låser de högsta bitarna i adressen.

Exempelvis för 24-bitars-subnätet ”192.168.2.0” är nätmasken ”255.255.255.0”. Ett kortare sätt att beskriva samma nät är ”192.168.2.0/24”.

Ett X-bitars subnät har alltid en mask med X st binära ettor i de högsta bitarna.

Det här gäller ipv4, i ipv6 är adresserna 128 bitar.

Nätadress och broadcast

I ett subnät på x bitar återstår $(32 - x)$ bitar för adressering, eftersom de x högsta bitarna är låsta.

Men – två av adressvärdena är speciella, det med **alla** bitar satta, och det med **inga** bitar satta.

Adressen med alla bitar kallas broadcast-adress. Skickas paket till den adressen når det alla adresser i subnätet.

Adressen med inga bitar är nätets adress och är reserverat.

Exempelvis i nätet ”192.168.2.0/24” är ”192.168.2.255” broadcast och ”192.168.2.0” nätets adress.

Routing

I varje subnät kan man ange en ”gateway” eller ”default route”. Det är via den adressen man kan nå andra subnät, och den router som finns där förmedlar paketen vidare.

Man kan också ha flera routers på ett subnät och ange flera routes förutom ”gateway”.

För att se routing-inställningar:

```
$ ip route
```

alternativt

```
$ netstat -rn
```

Namnuppslagning - DNS

Namnuppslagning mappar hostnamn till ip-adress, så att man kan skriva ”www.dn.se” istället för ip-adressen.

Klassiskt används ”/etc/resolv.conf” för att ange vilka namnservrar man ska använda.

I Ubuntu används ”netplan” även för det.

Dessutom kan man alltid lägga till egna hostnamn i ”/etc/hosts” som har högre prioritet än DNS – uppslagningar.

Grundläggande verktyg

”ip” är grundverktyget för att visa och ändra ip-relaterade inställningar.

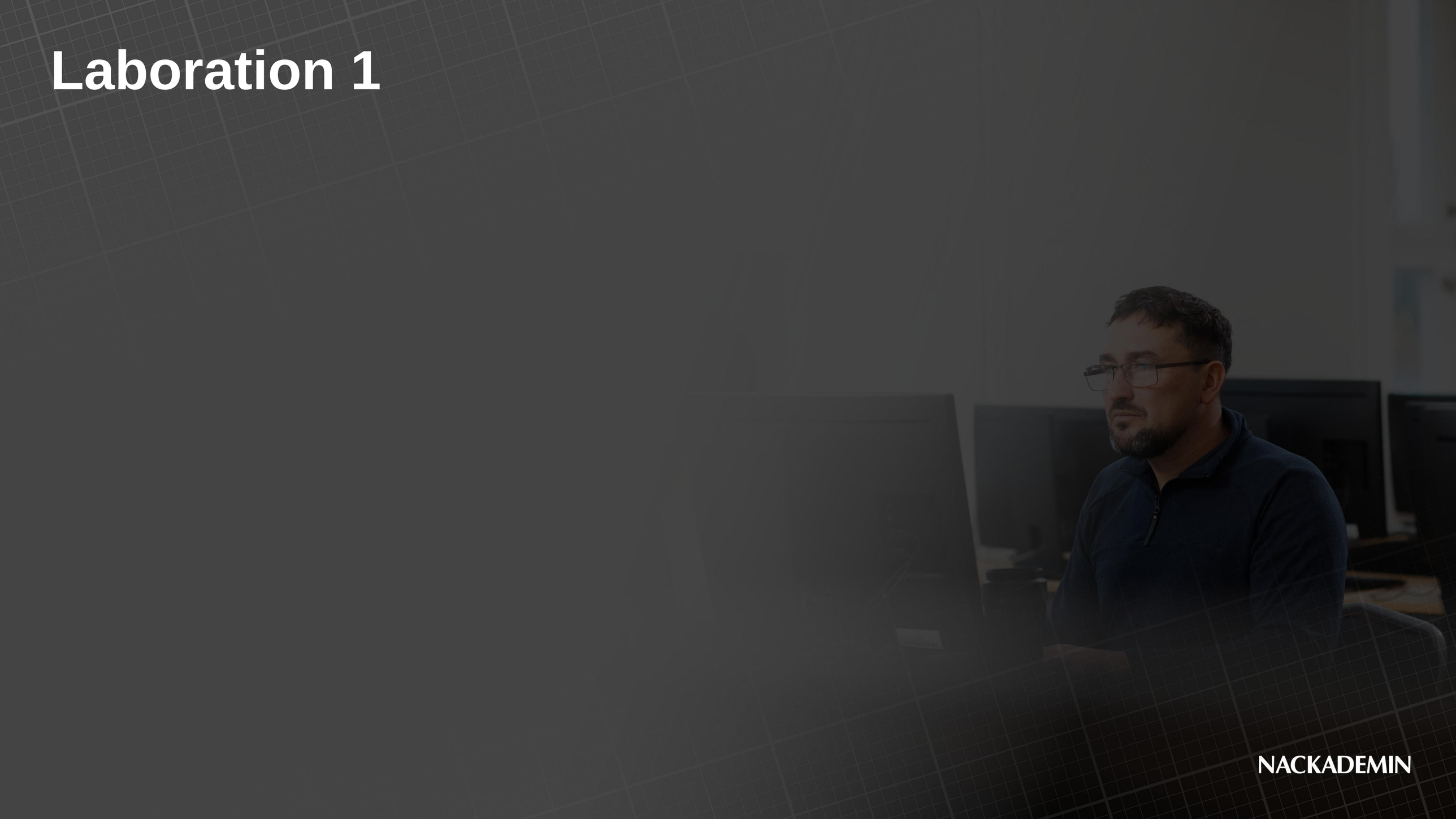
Det har många underkommandon – mest används nog ”ip addr” och ”ip route”.

”ping” används för att undersöka om det finns någon som svarar på en viss ip-adress. ”ping” använder icmp, som kan vara blockerat på en viss server, så man kan få ”false negatives”.

”netstat” kan visa route-information men även vilka tjänster som körs och väntar på uppkoppling på en server. Körs på servern där tjänsterna körs.

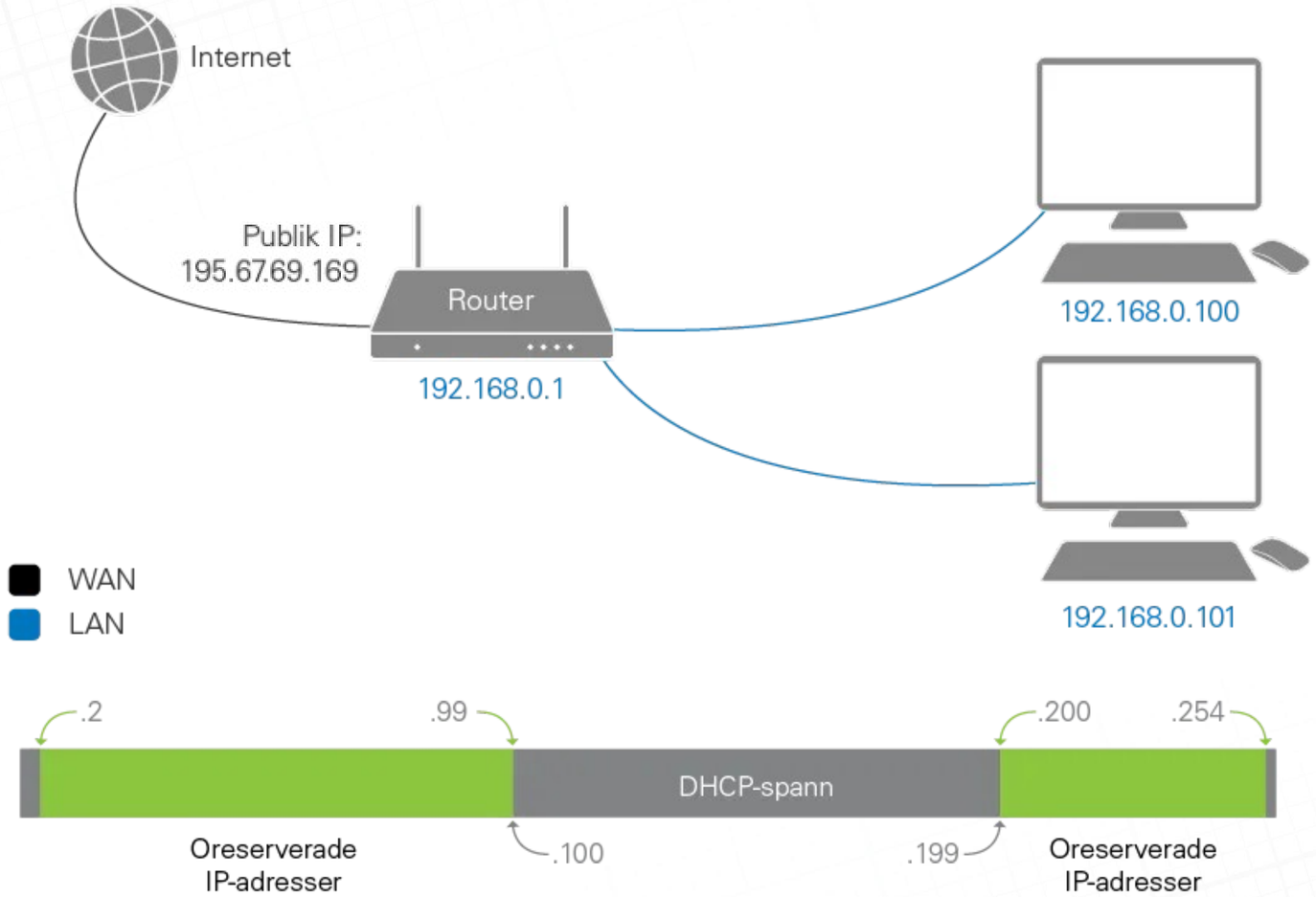
”host” används för att slå upp ett hostnamn.

Laboration 1



NACKADEMIN

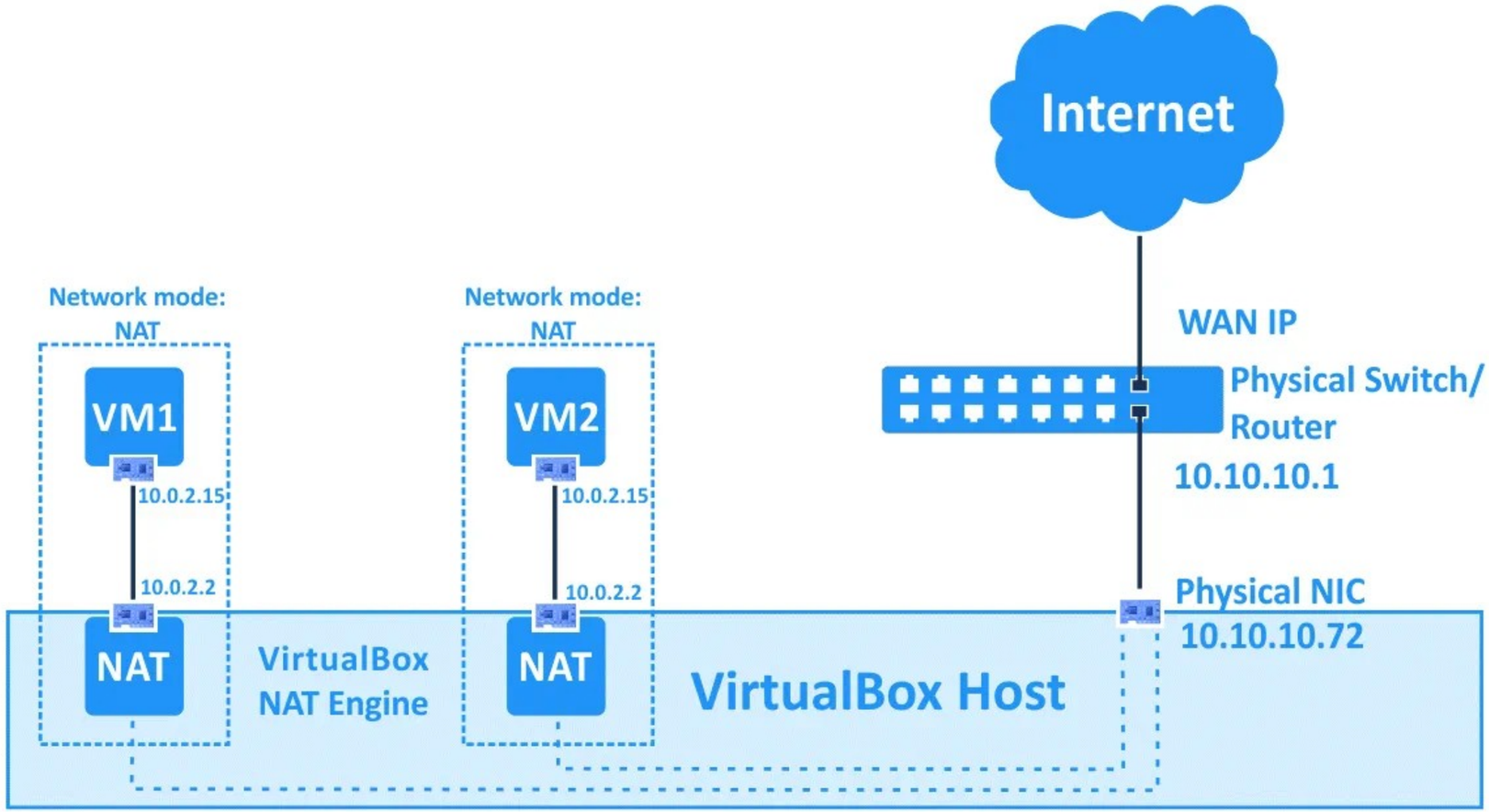
Nätverk - lan, gateway, router, nat, dhcp, internet



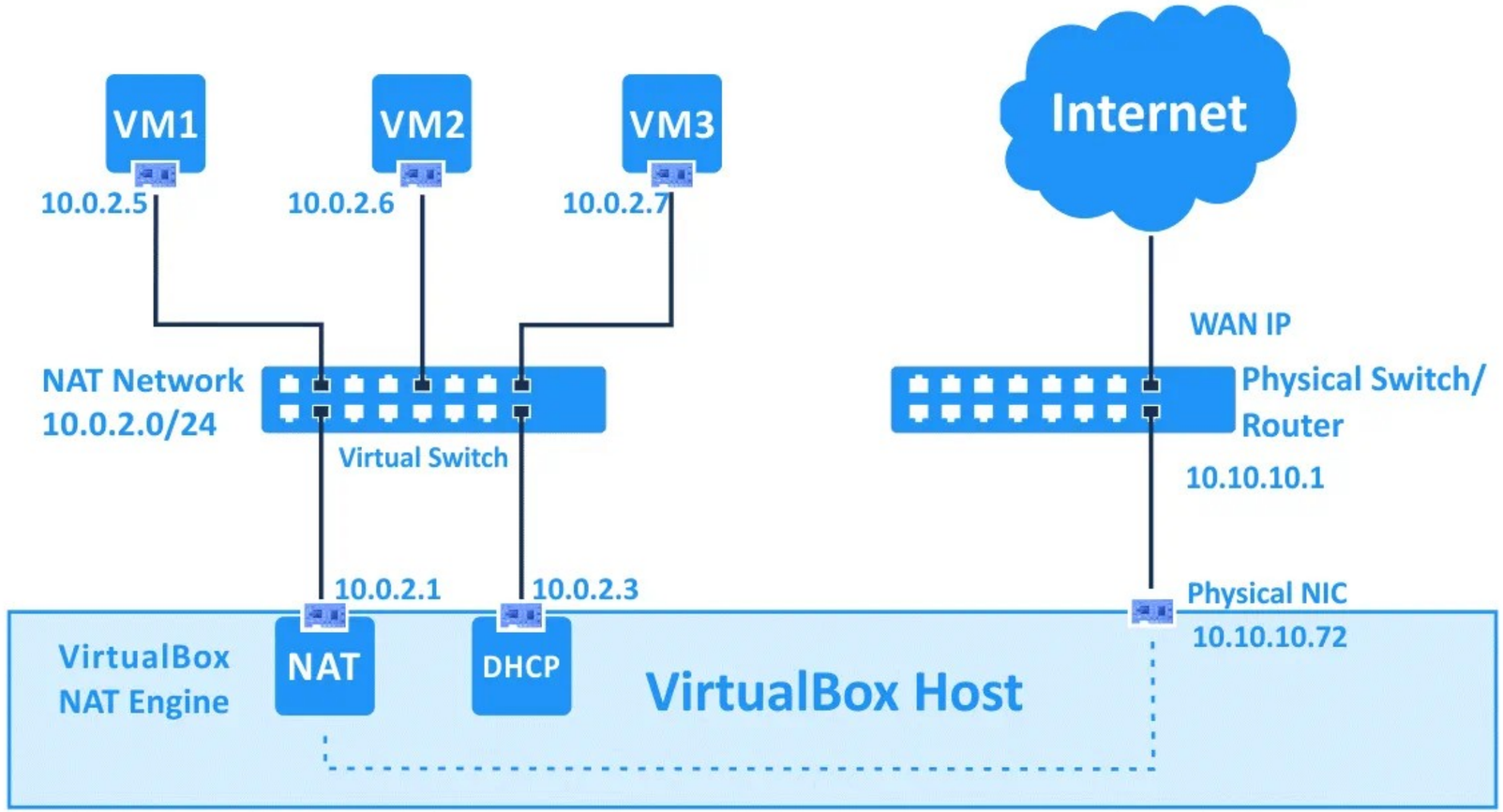
Datorer på samma lan kan nå varandra direkt och via en gateway nå internet.
Datorns ip blir översatt (NAT) innan den når mottagaren.
Mottagaren kan inte nå den enskilda datorn direkt utan en port forward i vår gateway.
Datorn erhåller sitt ip av en DHCP i routern.

Prova **`curl ipecho.net/plain`** *från din terminal så får du se ditt publika ip.*

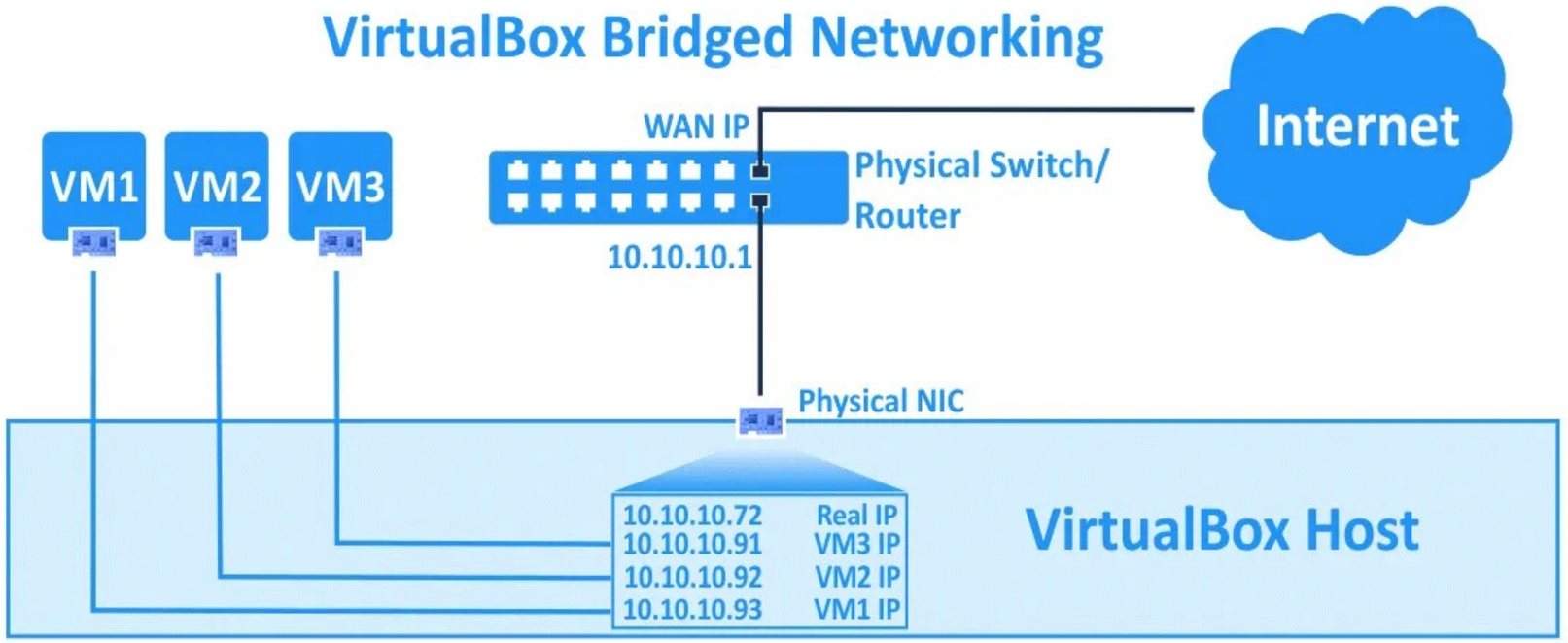
VirtualBox NAT



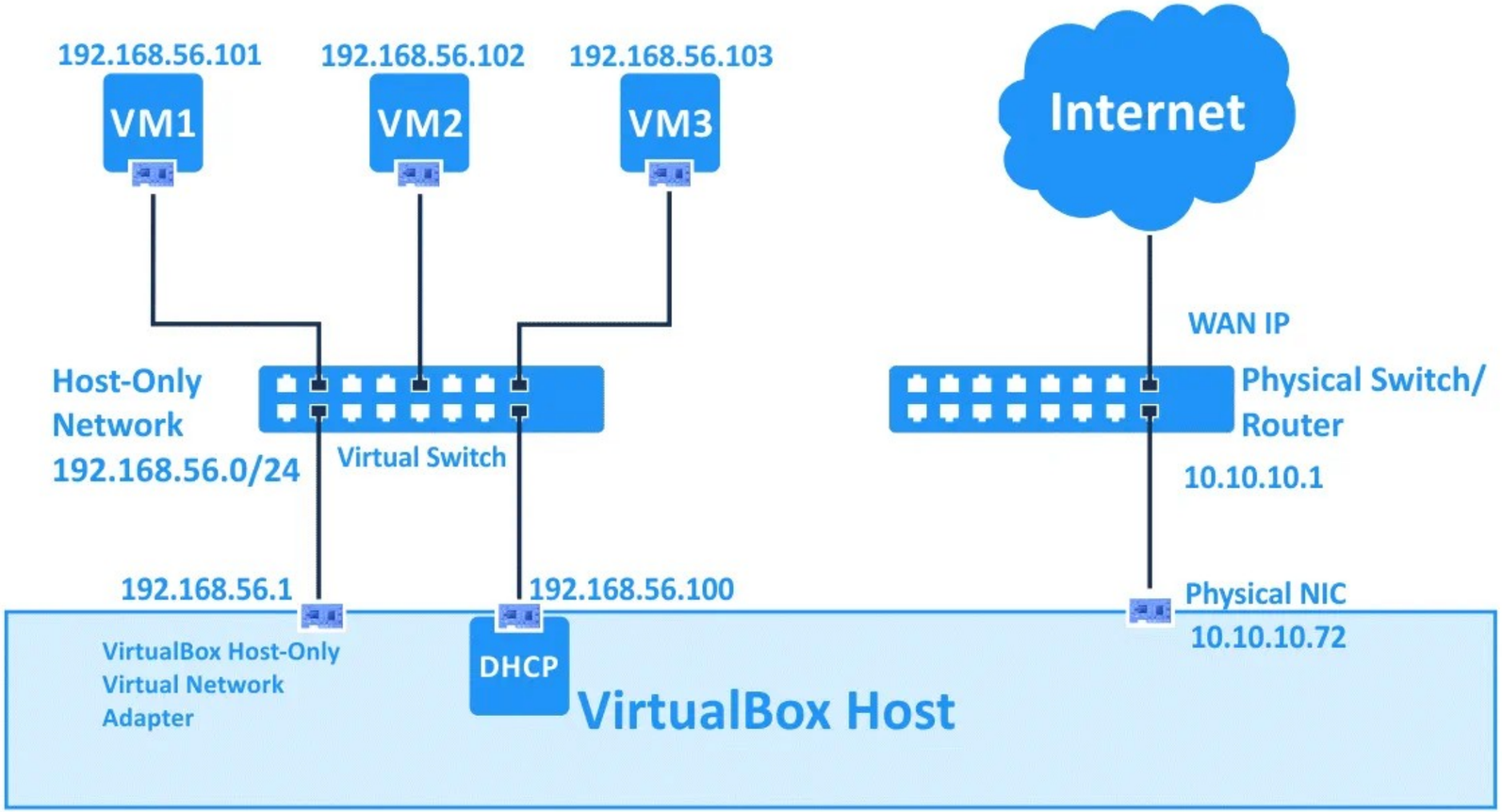
VirtualBox NAT-network



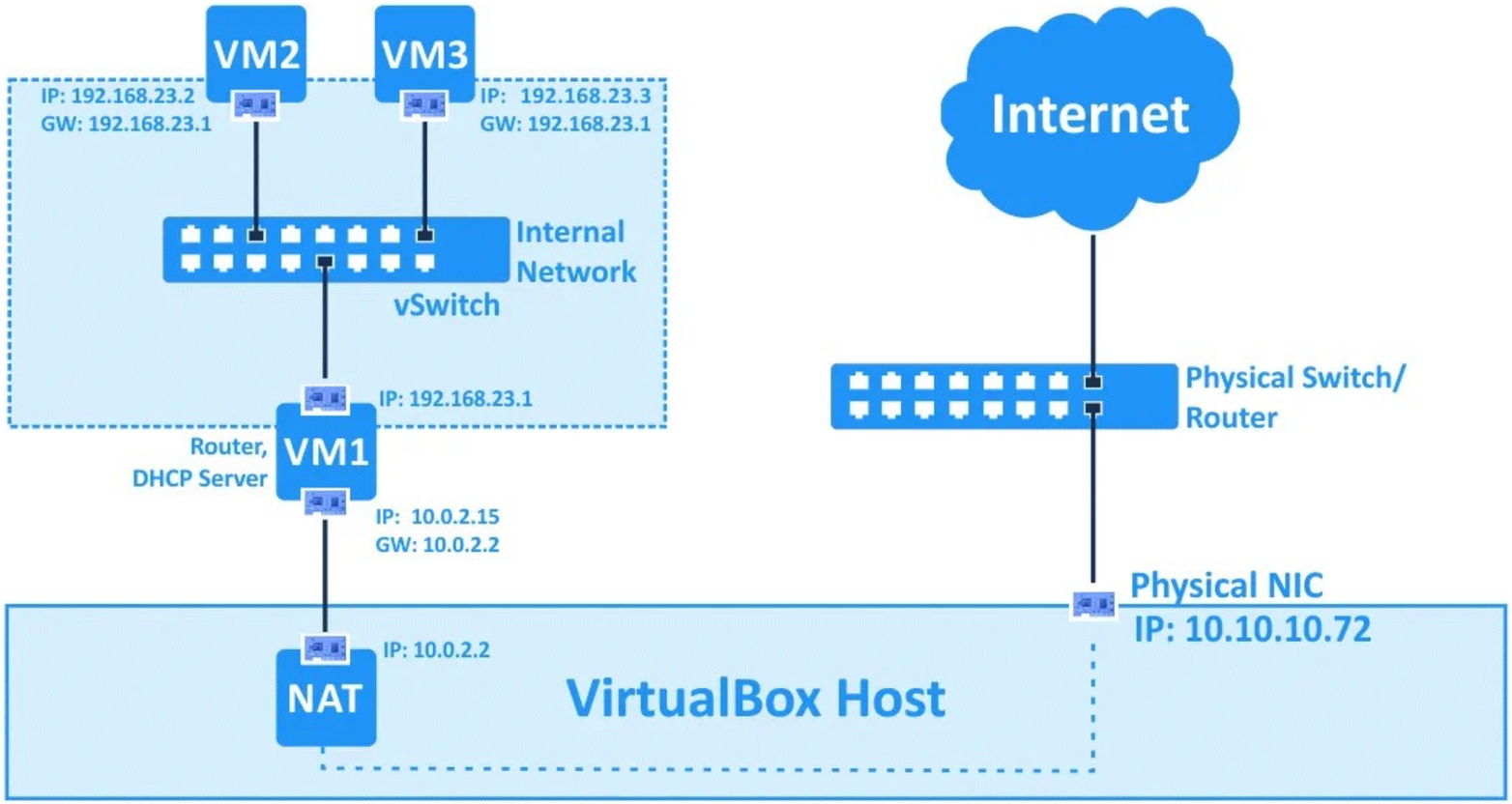
VirtualBox Bridged



VirtualBox Host-Only



VirtualBox internal-network



VirtualBox nätverksjämförelse

	VM ↔ VM	VM → Host	VM ← Host	VM → LAN	VM ← LAN
Not attached	–	–	–	–	–
NAT	–	+	Port Forward	+	Port Forward
NAT Network	+	+	Port Forward	+	Port Forward
Bridged	+	+	+	+	+
Internal Network	+	–	–	–	–
Host-only	+	+	+	–	–

Felsökning - uppkoppling

- ip link
- ip addr
- ping
- ip route / netstat -rn
- host
- nc -kl 5678 och telnet <ip> 5678

Felsökning – tjänster

På server-sidan:

- netstat -utl

-u = udp, -t = tcp, -l = listen

På klient-sidan:

- nc -z -w 1 <host> <port> && echo ”Yes!”
- telnet <host> <port>

Felsökning – att titta på trafik

”tcpdump” – dumpar trafik med filtrering

Exempel:

```
$ tcpdump port 80
```

Att dumpa trafik kan ibland vara ovärderligt för att spåra fel och lista ut vad som egentligen händer.

”tcpdump” måste köras med root-rättigheter. Varför?

Om du har tillgång till GUI – använd hellre wireshark!

iptables

Linux har en inbyggd brandvägg, iptables, som kan användas för att exakt styra hur ip-trafik flyter eller blockeras.

Om man är på ett system och tycker att trafik inte kommer fram som den ska kan man lista alla iptables-regler:

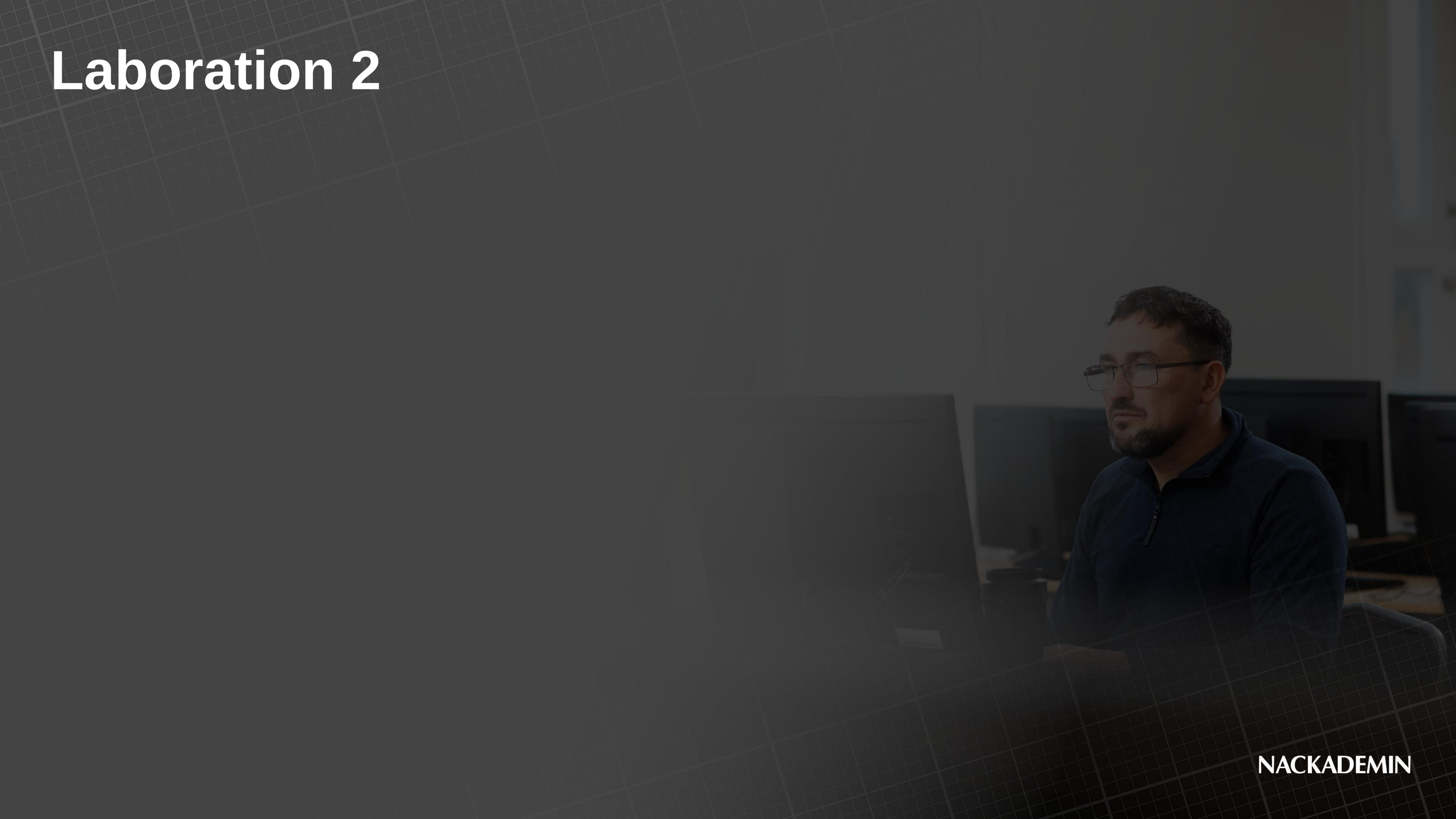
```
$ sudo iptables -L
```

Om man till lägga till en regel som blockerar all trafik från en viss ip kan man:

```
$ sudo iptables –A INPUT –i en01 –s 10.0.1.11 –j DROP
```

https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands

Laboration 2



NACKADEMIN

Summering

Idag har vi pratat om och experimenterat med olika nätverkskommandon och inställningar.

Vi har satt upp NFS för att dela filer och tittat på trafik mellan olika datorer.

Nästa gång

Mål: Kunna skapa och köra skript som tar argument. Använda villkor och tester.

- Exekverbara skript
- Variabler och argument
- Villkor med if
- test och ”modern test”

Stort tack!

NACKADEMIN