

Отчёт по лабораторной работе №1

Университет ИТМО

Дисциплина Телекоммуникационные системы и технологии

Консольные утилиты настройки сетевых компонентов Linux

Студент Киселев Илья Александрович

Группа М3305

Содержание

Часть 1. Базовые сетевые утилиты Linux	3
part1_net_utils.sh	3
Вывод выполнения	5
Часть 2. Виртуальные интерфейсы через nmcli	7
part2_nmcli_commands.sh	7
Результат	7
Часть 3. Реальные интерфейсы через netplan	9
part3_netplan_config.yaml	9
Результат	9
Часть 4. Объединение интерфейсов (bonding)	11
part4_bonding_setup.sh	11
Конфигурация	11
/proc/net/bonding/bond007	12
part4_stats_script.sh	12
Анализ (п.9)	13
Вопросы и задания	14

Часть 1. Базовые сетевые утилиты Linux

part1_net_utils.sh

```
#!/bin/bash
```

```
set -u
```

```
STATIC_IP="10.100.0.2"
```

```
STATIC_MASK="24"
```

```
STATIC_GATE="10.100.0.1"
```

```
STATIC_DNS="8.8.8.8"
```

```
require_root() {
```

```
    if [[ $EUID -ne 0 ]]; then
```

```
        echo "Скрипт нужно запускать от root (sudo)." >&2
```

```
        exit 1
```

```
    fi
```

```
}
```

```
pick_iface() {
```

```
    read -rp "Введите имя сетевого интерфейса: " IFACE
```

```
    if [[ ! -d "/sys/class/net/$IFACE" ]]; then
```

```
        echo "Интерфейс $IFACE не найден." >&2
```

```
        IFACE=""
```

```
    fi
```

```
}
```

```
show_link_info() {
```

```
    [[ -z "${IFACE:-}" ]] && pick_iface
```

```
    [[ -z "${IFACE:-}" ]] && return
```

```
    echo "--- Драйвер / шина ---"
```

```
    ethtool -i "$IFACE" 2>/dev/null
```

```
    BUS_INFO=$(ethtool -i "$IFACE" 2>/dev/null | awk -F': ' '{print $2}')
    if [[ -n "$BUS_INFO" ]]; then
```

```
        echo "--- Модель карты ---"
```

```
        lspci -s "$BUS_INFO"
```

```
    fi
```

```
    echo "--- Скорость / duplex / link ---"
```

```
    ethtool "$IFACE" 2>/dev/null | grep -E "Speed|Duplex|Link detected"
```

```
    echo "--- MAC-адрес ---"
```

```
    ip -brief link show "$IFACE"
```

```
}
```

```
show_ipv4_info() {
```

```
    [[ -z "${IFACE:-}" ]] && pick_iface
```

```
    [[ -z "${IFACE:-}" ]] && return
```

```

    echo "--- IPv4 адрес и маска ---"
    ip -4 addr show dev "$IFACE"

    echo "--- Gateway по умолчанию ---"
    ip route show default

    echo "--- DNS (/etc/resolv.conf) ---"
    cat /etc/resolv.conf
}

configure_static() {
    [[ -z "${IFACE:-}" ]] && pick_iface
    [[ -z "${IFACE:-}" ]] && return

    ip addr flush dev "$IFACE"
    ip addr add "${STATIC_IP}/${STATIC_MASK}" dev "$IFACE"
    ip link set "$IFACE" up
    ip route replace default via "$STATIC_GATE" dev "$IFACE"
    echo "nameserver $STATIC_DNS" > /etc/resolv.conf

    echo "Сценарий #1 применён: $STATIC_IP/$STATIC_MASK, gw $STATIC_GATE, dns
$STATIC_DNS"
}

configure_dhcp() {
    [[ -z "${IFACE:-}" ]] && pick_iface
    [[ -z "${IFACE:-}" ]] && return

    dhclient -r "$IFACE" 2>/dev/null
    ip addr flush dev "$IFACE"
    dhclient "$IFACE"

    echo "Сценарий #2 применён (DHCP) на $IFACE"
}

menu() {
    while true; do
        cat <<EOF

-- Настройка сети (интерфейс: ${IFACE:-не выбран}) --
1) Модель карты / скорость / duplex / link / MAC
2) Текущая конфигурация IPv4 (ip, mask, gate, dns)
3) Настроить по сценарию #1 (статика)
4) Настроить по сценарию #2 (DHCP)
5) Сменить интерфейс
0) Выход
EOF

        read -rp "Выбор: " choice
        case "$choice" in
            1) show_link_info ;;
            2) show_ipv4_info ;;
            3) configure_static ;;

```

```

        4) configure_dhcp ;;
        5) pick_iface ;;
        0) exit 0 ;;
        *) echo "Некорректный выбор" ;;
    esac
done
}

```

```

require_root
menu

```

Запускался на net-lab-debian1, интерфейс enp0s8.

Вывод выполнения

Пункт 2а (модель карты, скорость, duplex, link, MAC).

```

--- Драйвер / шина ---
driver: e1000
version: 6.12.107+deb13-amd64
firmware-version:
expansion-rom-version:
bus-info: 0000:00:08.0
supports-statistics: yes
supports-test: yes
supports-EEPROM-access: yes
supports-register-dump: yes
supports-priv-flags: no
--- Модель карты ---
00:08.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet
Controller (rev 02)
--- Скорость / duplex / link ---
    Speed: 1000Mb/s
    Duplex: Full
    Link detected: yes
--- MAC-адрес ---
enp0s8          UP              08:00:27:47:51:3b
<BROADCAST,MULTICAST,UP,LOWER_UP>

```

Пункт 2б (текущая конфигурация IPv4, до изменений, адрес получен по DHCP от NAT VirtualBox).

```

--- IPv4 адрес и маска ---
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    altname enx08002747513b
    inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic noprefixroute enp0s8
        valid_lft 86376sec preferred_lft 86376sec
--- Gateway по умолчанию ---
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
default via 10.0.3.2 dev enp0s8 proto dhcp src 10.0.3.15 metric 101
--- DNS (/etc/resolv.conf) ---
# Generated by NetworkManager
nameserver 10.0.2.3
nameserver fd17:625c:f037:2::3

```

```
nameserver 10.0.3.3
# NOTE: the libc resolver may not support more than 3 nameservers.
# The nameservers listed below may not be recognized.
nameserver fd17:625c:f037:3::3
```

Пункт 2с (сценарий №1, статика).

Сценарий #1 применён: 10.100.0.2/24, gw 10.100.0.1, dns 8.8.8.8

```
--- IPv4 адрес и маска ---
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    altname enx08002747513b
    inet 10.100.0.2/24 scope global enp0s8
        valid_lft forever preferred_lft forever
--- Gateway по умолчанию ---
default via 10.100.0.1 dev enp0s8
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
--- DNS (/etc/resolv.conf) ---
nameserver 8.8.8.8
```

Пункт 2д (сценарий №2, DHCP).

На установленном образе Debian 13 не было пакета isc-dhcp-client (dhclient отсутствовал), из-за чего первый запуск сценария завершился ошибкой dhclient: command not found после уже выполненного ip addr flush. Пакет установлен (apt-get install -y isc-dhcp-client), сценарий выполнен повторно.

Сценарий #2 применён (DHCP) на enp0s8

```
--- IPv4 адрес и маска ---
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    altname enx08002747513b
    inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic enp0s8
        valid_lft 86400sec preferred_lft 86400sec
--- Gateway по умолчанию ---
default via 10.0.3.2 dev enp0s8
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
--- DNS (/etc/resolv.conf) ---
nameserver 10.0.3.3
```

Часть 2. Виртуальные интерфейсы через nmcli

part2_nmcli_commands.sh

```
#!/bin/bash
```

```
IFACE="enp0s8"
```

```
# п.2a
```

```
nmcli con show
```

```
CON_NAME=$(nmcli -t -f NAME,DEVICE con show | awk -F: -v d="$IFACE" ' $2==d{print $1}')
```

```
nmcli con mod "$CON_NAME" \
    ipv4.method manual \
    ipv4.addresses 10.100.0.2/24 \
    ipv4.gateway 10.100.0.1 \
    ipv4.dns 8.8.8.8
nmcli con up "$CON_NAME"
```

```
# п.2b
```

```
nmcli con add type dummy ifname dm0 con-name dm0 \
    ipv4.method manual ipv4.addresses 10.100.0.3/24
```

```
# п.2c
```

```
nmcli con up dm0
```

```
# п.3
```

```
ping -c 4 -I "$IFACE" 10.100.0.3
```

```
ping -c 4 -I dm0 10.100.0.2
```

```
# п.4
```

```
ip -brief link show dm0
```

Запускался на net-lab-debian1, интерфейс enp0s8 переключён из NAT в intnet labnet.

Результат

```
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/4)
```

```
Connection 'dm0' (8c77b5cf-de7f-41ce-be50-7bcab15fb6d7) successfully added.
```

```
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/6)
```

```
PING 10.100.0.3 (10.100.0.3) from 10.100.0.2 enp0s8: 56(84) bytes of data.
```

```
From 10.100.0.2 icmp_seq=1 Destination Host Unreachable
```

```
From 10.100.0.2 icmp_seq=2 Destination Host Unreachable
```

```
From 10.100.0.2 icmp_seq=3 Destination Host Unreachable
```

```
From 10.100.0.2 icmp_seq=4 Destination Host Unreachable
```

```
--- 10.100.0.3 ping statistics ---
```

```
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3811ms
pipe 4
```

```
PING 10.100.0.2 (10.100.0.2) from 10.100.0.3 dm0: 56(84) bytes of data.
```

--- 10.100.0.2 ping statistics ---

4 packets transmitted, 0 received, 100% packet loss, time 3879ms

dm0 UNKNOWN 2e:79:9f:9a:63:60 <BROADCAST,NOARP,UP,LOWER_UP>

Контрольный ping 10.100.0.3 на debian1 без -I:

PING 10.100.0.3 (10.100.0.3) 56(84) bytes of data.

64 bytes from 10.100.0.3: icmp_seq=1 ttl=64 time=0.018 ms

64 bytes from 10.100.0.3: icmp_seq=2 ttl=64 time=0.052 ms

--- 10.100.0.3 ping statistics ---

2 packets transmitted, 2 received, 0% packet loss, time 1003ms

rtt min/avg/max/mdev = 0.018/0.035/0.052/0.017 ms

Без -I пакет доходит мгновенно (0.018 мс), потому что адрес 10.100.0.3 принадлежит локальному dm0, и ядро находит его в локальной таблице маршрутов напрямую. С -I enp0s8 ядро форсирует выход именно через этот интерфейс и резолвит адрес получателя ARP-запросом на проводе, а dm0 на ARP не отвечает (флаг NOARP), ip neigh show фиксирует 10.100.0.3 dev enp0s8 FAILED, ping отвечает Destination Host Unreachable. Обратное направление, ping -I dm0 10.100.0.2, теряет пакеты молча, без Destination Host Unreachable, потому что dm0 сам помечен NOARP и не пытается резолвить получателя через ARP, поэтому кадры с dm0 уходят, но ответа с enp0s8 не находят. Это другое проявление того же конфликта (два интерфейса одного хоста в одной подсети 10.100.0.0/24). Последняя строка вывода выше показывает MAC-адрес dm0, это 2e:79:9f:9a:63:60.

Часть 3. Реальные интерфейсы через netplan

part3_netplan_config.yaml

```
network:
  version: 2
  ethernets:
    enp0s8:
      addresses:
        - 10.100.0.4/24
        - 10.100.0.5/24
      routes:
        - to: default
          via: 10.100.0.3
      nameservers:
        addresses: [8.8.8.8]
```

Применён на net-lab-debian2 как /etc/netplan/01-lab.yaml (интерфейс enp0s8, тот же intnet labnet, что и на debian1).

Результат

Команды из part3_check_commands.sh, выполнены на debian2. Вывод debian2.

```
--- ping 10.100.0.2 ---
PING 10.100.0.2 (10.100.0.2) 56(84) bytes of data.
64 bytes from 10.100.0.2: icmp_seq=1 ttl=64 time=1.59 ms
64 bytes from 10.100.0.2: icmp_seq=2 ttl=64 time=0.815 ms

--- 10.100.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1021ms
rtt min/avg/max/mdev = 0.815/1.202/1.589/0.387 ms
--- ping 10.100.0.3 ---
PING 10.100.0.3 (10.100.0.3) 56(84) bytes of data.
64 bytes from 10.100.0.3: icmp_seq=1 ttl=64 time=0.994 ms
64 bytes from 10.100.0.3: icmp_seq=2 ttl=64 time=0.881 ms

--- 10.100.0.3 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.881/0.937/0.994/0.056 ms
--- ping 10.100.0.4 ---
PING 10.100.0.4 (10.100.0.4) 56(84) bytes of data.
64 bytes from 10.100.0.4: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from 10.100.0.4: icmp_seq=2 ttl=64 time=0.051 ms

--- 10.100.0.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1178ms
rtt min/avg/max/mdev = 0.022/0.036/0.051/0.014 ms
--- ping 10.100.0.5 ---
PING 10.100.0.5 (10.100.0.5) 56(84) bytes of data.
64 bytes from 10.100.0.5: icmp_seq=1 ttl=64 time=0.015 ms
64 bytes from 10.100.0.5: icmp_seq=2 ttl=64 time=0.046 ms

--- 10.100.0.5 ping statistics ---
```

```
2 packets transmitted, 2 received, 0% packet loss, time 1553ms
rtt min/avg/max/mdev = 0.015/0.030/0.046/0.015 ms
--- arp-кэш ---
10.100.0.2 dev enp0s8 lladdr 08:00:27:47:51:3b REACHABLE
10.0.2.3 dev enp0s3 lladdr 52:54:00:12:35:00 REACHABLE
10.100.0.3 dev enp0s8 lladdr 08:00:27:47:51:3b REACHABLE
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:00 REACHABLE
fe80::2 dev enp0s3 lladdr 52:54:00:12:35:00 router STALE
```

Дополнительно снято с debian1, для покрытия всех четырёх адресов и сравнения ip neigh show.

```
--- ping 10.100.0.4 from debian1 ---
PING 10.100.0.4 (10.100.0.4) 56(84) bytes of data.
64 bytes from 10.100.0.4: icmp_seq=1 ttl=64 time=0.409 ms
64 bytes from 10.100.0.4: icmp_seq=2 ttl=64 time=0.919 ms

--- 10.100.0.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1077ms
rtt min/avg/max/mdev = 0.409/0.664/0.919/0.255 ms
--- ping 10.100.0.5 from debian1 ---
PING 10.100.0.5 (10.100.0.5) 56(84) bytes of data.
64 bytes from 10.100.0.5: icmp_seq=1 ttl=64 time=1.24 ms
64 bytes from 10.100.0.5: icmp_seq=2 ttl=64 time=0.680 ms

--- 10.100.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 0.680/0.958/1.236/0.278 ms
--- ip neigh on debian1 ---
10.0.2.3 dev enp0s3 lladdr 52:54:00:12:35:00 STALE
10.100.0.3 dev enp0s8 FAILED
10.100.0.4 dev enp0s8 lladdr 08:00:27:52:31:fd REACHABLE
10.100.0.5 dev enp0s8 lladdr 08:00:27:52:31:fd REACHABLE
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:00 DELAY
fe80::2 dev enp0s3 lladdr 52:54:00:12:35:00 router STALE
```

Все четыре адреса пингуются в обе стороны, потерь нет. На debian2 .2 и .3 резолвятся в один и тот же MAC enp0s8 debian1, потому что за dm0 отвечает на ARP основной интерфейс debian1, а сам dm0 на ARP не отвечает (см. Часть 2). На debian1 запись 10.100.0.3 dev enp0s8 FAILED, тот же ожидаемый эффект для собственного dm0.

Часть 4. Объединение интерфейсов (bonding)

part4_bonding_setup.sh

```
#!/bin/bash
```

```
SLAVE1="enp0s9"
```

```
SLAVE2="enp0s10"
```

```
# п.2
```

```
lsmod | grep bonding || modprobe bonding
```

```
# п.3
```

```
nmcli con add type bond ifname bond007 con-name bond007 \
    bond.options "mode=balance-rr,miimon=100" \
    ipv4.method auto
```

```
nmcli con add type ethernet ifname "$SLAVE1" master bond007 con-name bond007-
slave1
```

```
nmcli con add type ethernet ifname "$SLAVE2" master bond007 con-name bond007-
slave2
```

```
# п.4
```

```
nmcli con up bond007
```

```
nmcli con up bond007-slave1
```

```
nmcli con up bond007-slave2
```

```
ip addr show bond007
```

```
ip -brief link show
```

```
# п.5
```

```
cat /proc/net/bonding/bond007
```

```
# п.6
```

```
cat /proc/net/dev
```

Запускался на net-lab-debian1, в роли слейвов выступили два новых NAT-адаптера enp0s9/enp0s10 (nic3/nic4), а enp0s3 (nic1, SSH) и enp0s8 (nic2, intnet) не затрагивались.

Конфигурация

Группа собрана только через nmcli, конфигурационные файлы руками не редактировались. Сгенерированный /etc/NetworkManager/system-connections/bond007.nmconnection:

```
[connection]
```

```
id=bond007
```

```
uuid=4e1999b2-9a49-4623-af12-1f4d35ba349d
```

```
type=bond
```

```
interface-name=bond007
```

```
[bond]
```

```
miimon=100
```

```
mode=balance-rr
```

```
[ipv4]  
method=auto
```

```
[ipv6]  
addr-gen-mode=default  
method=auto
```

```
[proxy]
```

/proc/net/bonding/bond007

Ethernet Channel Bonding Driver: v6.12.107+deb13-amd64

```
Bonding Mode: load balancing (round-robin)  
MII Status: up  
MII Polling Interval (ms): 100  
Up Delay (ms): 0  
Down Delay (ms): 0  
Peer Notification Delay (ms): 0
```

```
Slave Interface: enp0s9  
MII Status: up  
Speed: 1000 Mbps  
Duplex: full  
Link Failure Count: 0  
Permanent HW addr: 08:00:27:e8:24:63  
Slave queue ID: 0
```

```
Slave Interface: enp0s10  
MII Status: up  
Speed: 1000 Mbps  
Duplex: full  
Link Failure Count: 0  
Permanent HW addr: 08:00:27:de:1a:49  
Slave queue ID: 0
```

Снято через несколько секунд после `nmcli con up`. Сразу после активации `cat /proc/net/bonding/bond007` показывал `MII Status: down` для обоих слейвов, линк ещё не согласовался. Приведённый вывод снят после согласования линка. `bond007` получил адрес по DHCP от VirtualBox NAT, 10.0.4.15/24.

part4_stats_script.sh

```
#!/bin/bash
```

```
IFACE_FILTER="${1:-}"  
REPEATS="${2:-1}"  
DELAY="${3:-1}"
```

```
snapshot() {  
    local ts  
    ts=$(date '+%Y-%m-%d %H:%M:%S')
```

```

tail -n +3 /proc/net/dev | while read -r line; do
    name=$(echo "$line" | cut -d: -f1 | xargs)
    [[ -n "$IFACE_FILTER" && "$name" != "$IFACE_FILTER" ]] && continue

    rest=$(echo "$line" | cut -d: -f2)
    rx_packets=$(echo "$rest" | awk '{print $2}')
    tx_packets=$(echo "$rest" | awk '{print $10}')

    printf "%s  iface=%-10s RX=%-10s TX=%-10s\n" "$ts" "$name"
"$rx_packets" "$tx_packets"
done
}

for ((i = 1; i <= REPEATS; i++)); do
    snapshot
    (( i < REPEATS )) && sleep "$DELAY"
done

```

Анализ (п.9)

Запущен ping -c 20 -I bond007 8.8.8.8 в фоне, параллельно троекратно снята статистика по каждому слейву (part4_stats_script.sh enp0s9 3 4 и part4_stats_script.sh enp0s10 3 4):

2026-09-19 11:37:14	iface=enp0s9	RX=7	TX=32
2026-09-19 11:37:18	iface=enp0s9	RX=8	TX=33
2026-09-19 11:37:22	iface=enp0s9	RX=10	TX=35
2026-09-19 11:37:22	iface=enp0s10	RX=9	TX=28
2026-09-19 11:37:26	iface=enp0s10	RX=11	TX=30
2026-09-19 11:37:30	iface=enp0s10	RX=13	TX=32

TX-пакеты растут примерно поровну на обоих слейвах, на enp0s9 32, затем 33, затем 35, на enp0s10 28, затем 30, затем 32, что подтверждает round-robin распределение при mode=balance-rr. Сам ping -I bond007 8.8.8.8 прошёл с 0% потерь (20 из 20 пакетов), оба NAT-адаптера VirtualBox дали независимый выход в интернет, и bonding через них отработал штатно.

Вопросы и задания

1. Как с помощью команды `ip`.

a. назначить новый IPv4-адрес?

```
ip addr add 10.100.0.2/24 dev eth0
```

b. назначить новый MAC-адрес?

```
ip link set dev eth0 down
ip link set dev eth0 address 00:11:22:33:44:55
ip link set dev eth0 up
```

c. назначить новый gateway?

```
ip route replace default via 10.100.0.1 dev eth0
```

d. вывести информацию arp-кэше?

```
ip neigh show
```

e. очистить arp-кэш?

```
ip neigh flush all
```

f. включить интерфейс?

```
ip link set eth0 up
```

g. выключить интерфейс?

```
ip link set eth0 down
```

2. Как с помощью `nmcli` назначить на интерфейс статический IP адрес, маску и настроить default gateway?

```
nmcli con mod "<имя соединения>" \
    ipv4.method manual \
    ipv4.addresses 10.100.0.2/24 \
    ipv4.gateway 10.100.0.1
nmcli con up "<имя соединения>"
```

3. Как с помощью `netplan` назначить на интерфейс статический IP адрес, маску и настроить default gateway?

Отредактировать YAML в `/etc/netplan/*.yaml`.

```
network:
  version: 2
  ethernets:
    enp0s3:
      addresses: [10.100.0.2/24]
      routes:
        - to: default
          via: 10.100.0.1
```

Затем применить.

```
sudo netplan apply
```

4. Какие режимы bonding стандартно существуют в Linux? Опишите их назначение, возможности по отказоустойчивости и необходимость поддержки со стороны оборудования.

Режим	Имя	Балансировка	Отказоустойчивость	Нужна поддержка свитча
0	balance-rr	Да, round-robin по всем интерфейсам	Да	Не обязательно, но возможна доставка пакетов не по порядку
1	active-backup	Нет, работает один активный интерфейс	Да	Не требуется
2	balance-xor	Да, по хешу (MAC/IP)	Да	Нужна статическая агрегация каналов на свитче
3	broadcast	Нет, дублирует трафик на все интерфейсы	Да (максимальная избыточность)	Не требуется, но большой оверхед трафика
4	802.3ad (LACP)	Да, динамическая агрегация	Да	Требуется свитч с поддержкой 802.3ad/LACP
5	balance-tlb	Да, только исходящий трафик по загрузке каналов	Да	Не требуется
6	balance-alb	Да, исходящий и входящий (через ARP)	Да	Не требуется

В работе используется **mode=balance-rr**: пакеты передаются поочерёдно через каждый интерфейс группы по кругу, что даёт как балансировку нагрузки, так и отказоустойчивость, без необходимости специальной настройки свитча (в отличие от LACP).

5. Какие существуют и чем отличаются режимы работы адаптера (duplex)?

- **Full Duplex.** Одновременная передача и приём данных в обе стороны, коллизии невозможны, максимальная пропускная способность канала используется в обоих направлениях сразу.
- **Half Duplex.** Передача в конкретный момент возможна только в одну сторону, при одновременной попытке передачи с двух сторон возникает коллизия (характерно для старых shared-сред на хабах и некоторых беспроводных технологий).

6. Какой, по-вашему, практический смысл в возможности назначения нескольких IP адресов на один интерфейс?

Позволяет размещать несколько сервисов/сайтов на одном физическом сетевом адаптере (IP-based virtual hosting), плавно мигрировать сервер из одной подсети в другую без простоя (временно держать адреса старой и новой сети одновременно), а также обращаться к одному узлу из разных подсетей без добавления дополнительного физического оборудования.

7. Какой, по-вашему, практический смысл в возможности создания виртуальных интерфейсов?

Виртуальные интерфейсы (dummy, bridge, VLAN, bond, tun/tap) позволяют тестировать и отлаживать сетевые конфигурации без дополнительного физического оборудования, изолировать трафик контейнеров/сетевых неймспейсов, организовывать VLAN поверх одного физического порта и объединять несколько физических адаптеров в один логический интерфейс (bonding) для отказоустойчивости или роста пропускной способности.