

# Методика тестирования сетевой карты ОСР для сервалов, также PCIe для тiog

1. воткнуть dас-кабель
2. Далее установить пакет bwg-ng
3. Проверить наличие сетевухи командой =====***sudo lspci -d ::200***
4. Настроить сетевые интерфейсы, по примеру:

```
auto lo
iface lo inet loopback
up ip link add vrf1 type vrf table 1
up ip link set dev vrf1 up
up ip link add vrf2 type vrf table 2
up ip link set dev vrf2 up

auto enp1s0
iface enp1s0 inet dhcp

auto ens1f0np0
iface ens1f0np0 inet static
address 10.10.10.1
netmask 24
pre-up ip link set dev ens1f0np0 master vrf1

auto ens1f1np1
iface ens1f1np1 inet static
address 10.10.10.2
netmask 24
pre-up ip link set dev ens1f1np1 master vrf2
```

5. При поднятии интерфейса lo создаются два виртуальных маршрутизатора: vrf1 (таблица маршрутизации 1) и vrf2 (таблица 2).
6. Каждый VRF — это изолированное пространство маршрутизации: свои таблицы, свои сокеты, свои правила.
7. Флаг up гарантирует, что команды выполняются при активации интерфейса.

<code>ip link add vrf1</code>	Создаёт виртуальное сетевое устройство с именем <code>vrf1</code>
<code>type vrf</code>	Указывает, что это именно VRF-устройство (специальный тип в <code>iproute2</code> )
<code>table 1</code>	Привязывает VRF к таблице маршрутизации №1 из <code>/etc/iproute2/rt_tables</code>

Задаем IP-адрес и маску подсети.

◆ Привязка интерфейсов: `master vrf1`

**`pre-up ip link set dev enp66s0f0np0 master vrf1`**

Это ключевой момент.

Что происходит:

Физический интерфейс `enp66s0f0np0` становится «рабом» (slave) VRF-устройства `vrf1`.

Весь входящий/исходящий трафик этого интерфейса автоматически направляется через таблицу маршрутизации 1.

Интерфейс теряет связь с основной таблицей `main` (254).

Как идет трафик в данной схеме?

Трафик идет по проводу, так как на интерфейсах адреса, которые назначены в одной подсети.

## Server

**`*sudo ip vrf exec vrf1 iperf3 -s -B 10.10.10.1*`**

## Client

**`while true; do sudo ip vrf exec vrf2 iperf3 -c 10.10.10.1 -B 10.10.10.2 -b 0 -P 5 -t 300;done`**

### Шаг 1: Сервер запускается

- `iperf3` в контексте `vrf1` открывает сокет на `10.10.10.1:5201`.
- Сокет «живёт» в изолированном мире таблицы маршрутизации №1.
- Сервер ждёт подключений — **но только от клиентов из своего же мира (`vrf1`)**.

### Шаг 2: Клиент пытается подключиться

- `iperf3` в контексте `vrf2` решает: «`10.10.10.1` — это мой сосед по подсети `10.10.10.0/24`».
- Пытается установить TCP-соединение:
  1. **Нужен MAC-адрес** → отправляет ARP-запрос: «Кто `10.10.10.1`?»

2. ARP-запрос уходит в эфир через интерфейс `enp66s0f1np1` .

а через утилиту `bwn-ng` можно это чекать как идет трафик по проводу в сетевухе.

Interface	RX Rate (KB/s)	TX Rate (KB/s)	Total (KB/s)
lo	0.00	0.00	0.00
enp1s0	1.51	0.62	2.13
ens1f0np0	1,101.88	2,293,547.75	2,294,649.50
ens1f1np1	2,292,624.50		2,293,726.25
vrf1	868.15	1,101.88	
vrf2	2,194,489.00	0.00	868.15
docker0		0.00	2,194,489.00
	0.00	0.00	0.00
Total	4,489,085.00	2,294,650.25	6,783,735.00

На этом все, получается что сетевуха работает  
перед использованием проверить .