Network Binder

Bind multiple internet connection endpoints into single one using the Multipath TCP (mptcp).

Project status

Currently **UNDER THE DEVELOPMENT**.

Development environtmen

- Ubuntu Server 24.04.2 LTS

TODOs

- Replace / rebrand the `netplan-*` to `network-binder`.

Project Structure

```
/root/
│
├── ...
├── docker-compose.yml
├── config/
│   ├── netplan/
│   │   └── 01-netcfg.yaml      # Will be generated
│   ├── notifications.yaml
│   └── benchmarks.yaml
├── scripts/
│   ├── netplan_optimizer.py    # Main logic
│   └── entrypoint.sh
├── Dockerfile
├── ...
├── LICENSE
├── README.md
└── README.pdf                  # Will be generated
```

Deployment Workflow

1. **Build and Start**:

   ```
   docker-compose build
   docker-compose up -d
   ```

2. **Manual Trigger**:

```
docker exec netplan-optimizer python netplan_optimizer.py
```

3. **View Logs**:

```
docker logs -f netplan-optimizer
```

Firewall setup

1. Install and Enable iptables

```
sudo apt update
sudo apt install -y iptables iptables-persistent
sudo systemctl enable netfilter-persistent
```

---

2. Basic Firewall Rules (Pre-Docker)

Create `/etc/iptables/rules.v4` with these **essential rules** that won't break Docker:

```
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]

# Allow established connections
-A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
-A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

# Allow loopback
-A INPUT -i lo -j ACCEPT

# Allow ICMP (ping)
-A INPUT -p icmp --icmp-type echo-request -j ACCEPT

# Allow SSH (change port if needed)
-A INPUT -p tcp --dport 22 -j ACCEPT

# Allow WireGuard VPN
-A INPUT -p udp --dport 51820 -j ACCEPT

# Allow DHCP (if running outside Docker)
-A INPUT -p udp --dport 67:68 -j ACCEPT

# Docker bridge networks (critical!)
-A FORWARD -i docker0 -o docker0 -j ACCEPT
-A FORWARD -i br-+ -o br-+ -j ACCEPT
```

```
COMMIT
```

Apply immediately:

```
sudo iptables-restore < /etc/iptables/rules.v4
```

---

3. Docker-Compatible Rules

Add these **Docker-specific rules** to allow container networking:

```
# Allow container ↔ host communication
sudo iptables -I INPUT 1 -i docker0 -j ACCEPT

# Allow container ↔ internet (via NAT)
sudo iptables -t nat -A POSTROUTING -s 172.17.0.0/16 ! -o docker0 -j
MASQUERADE

# Allow MPTCP traffic
sudo iptables -A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
sudo iptables -A FORWARD -p tcp -m multiport --dports 80,443 -j ACCEPT
```

---

4. Save Rules Permanently

```
sudo netfilter-persistent save
sudo systemctl restart netfilter-persistent
```

---

5. Verify Setup

```
# Check rules
sudo iptables -L -n -v --line-numbers
sudo iptables -t nat -L -n -v

# Test Docker connectivity
docker run --rm alpine ping -c 4 8.8.8.8

# Test external access to containers
docker run -d -p 80:80 nginx
curl localhost
```

---

6. (Optional) ufw Alternative

If you prefer `ufw`:

```
sudo apt install ufw
sudo ufw disable

# Reset and allow Docker
sudo ufw reset
sudo ufw default deny incoming
sudo ufw default allow outgoing

# Essential rules
sudo ufw allow 22/tcp
sudo ufw allow 51820/udp
sudo ufw allow in on docker0
sudo ufw allow in on br-+

# Special Docker bypass
echo -e "*filter\n:ufw-user-forward - [0:0]\nCOMMIT" | sudo tee
/etc/ufw/after.rules
echo -e "DOCKER-USER -j ufw-user-forward" | sudo tee -a
/etc/ufw/after.rules

# Enable
sudo ufw enable
```

---

7. Critical MPTCP Firewall Rules

Add to `/etc/iptables/rules.v4` before `COMMIT`:

```
# MPTCP bonding requirements
-A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
-A FORWARD -p tcp -m multiport --dports 80,443 -j ACCEPT
-A FORWARD -i eth+ -o eth+ -j ACCEPT  # Allow cross-WAN traffic
```

---

8. Final Checks

```
# Ensure rules persist after reboot
sudo apt install iptables-persistent
sudo netfilter-persistent save

# Verify Docker networking
docker run --rm curlimages/curl curl ifconfig.me
```

Firewall Rule Summary Table

Rule	Purpose	Command
Docker NAT	Container internet access	`iptables -t nat -A POSTROUTING -s 172.17.0.0/16 ! -o docker0 -j MASQUERADE`
MPTCP	Bonding traffic	`iptables -A FORWARD -i eth+ -o eth+ -j ACCEPT`
VPN Access	WireGuard port	`iptables -A INPUT -p udp --dport 51820 -j ACCEPT`
SSH	Remote management	`iptables -A INPUT -p tcp --dport 22 -j ACCEPT`

Troubleshooting

If Docker breaks:

```
# Reset to allow-all temporarily
sudo iptables -P INPUT ACCEPT
sudo iptables -P FORWARD ACCEPT
sudo iptables -P OUTPUT ACCEPT
sudo iptables -F
sudo systemctl restart docker
```

Then reapply rules from step 2.

This configuration:

✅ **Secures your host** while allowing Docker networking
✅ **Preserves MPTCP bonding** functionality
✅ **Maintains VPN access** for remote management
✅ **Survives reboots** with persistent rules