Pterodactyl

Zielsetzung

Das Routing von einkommendem IPv4-Traffic (z.B. von Steam) über einen Oracle Cloud Server, weiter per IPv6 durch das Internet zum lokalen Raspberry Pi (Internet Point für Server Traffic), und schließlich per IPv4 ins Heimnetzwerk zum eigentlichen Game-Server (Debian/Pterodactyl) oder zu Server wie Teamspeak auf dem Raspberry Pi.
Dieses Setup minimiert Latenz (Ping) durch den **Verzicht auf VPNs wie Tailscale/Wireguard** und nutzt natives Netzwerk-Routing via `socat`.

1. Übersicht & IPs (Beispiel Icarus Server)

- Oracle VPS (Public IPv4)
  - IP Adressen
  - Der Server, mit dem sich die Spieler verbinden
- Raspberry Pi (Public IPv6)
  - IP Adressen
  - Das Gateway im Heimnetz
- Game-Server Debian (Local IPv4)
  - IP Adressen
  - Der Rechner, auf dem Pterodactyl/Docker läuft
- Benötigte Ports
  - Icarus: `17777 UDP`
  - Steam: `27015 UDP`

2. Pterodactyl (Game-Server) Vorbereitung

Damit der Game-Server in Docker die weitergeleiteten Pakete von fremden IPs akzeptiert:

1. Im Pterodactyl-Panel unter Network sicherstellen, dass dem Server die IP `0.0.0.0` zugewiesen ist (nicht die lokale `192.168.x.x` Adresse).
2. Unter Startup bei `SERVER_IP` (falls vorhanden) ebenfalls `0.0.0.0` eintragen.
3. Server neustarten.

---

3. Oracle Cloud Setup (IPv4 -> IPv6)

Der Oracle-Server fängt den IPv4-Traffic der Spieler ab und leitet ihn per IPv6 an den Raspberry Pi zu Hause weiter.

Voraussetzungen auf Oracle

Sicherstellen, dass in der Oracle Cloud Web-Oberfläche (Dashboard) die benötigten Ports (`17777` und `27015 UDP`) in den Ingress Rules (Eingehend) der Virtual Cloud Network Security List geöffnet sind.
Zusätzlich müssen die Ports in der Ubuntu-Firewall geöffnet sein:

```
sudo iptables -I INPUT -p udp -m udp --dport 17777 -j ACCEPT
sudo iptables -I INPUT -p udp -m udp --dport 27015 -j ACCEPT
```

Portmapping mit socat (Oracle)

Alte socat-Prozesse beenden:

```
sudo pkill socat
```

Neue Routen für Game- und Query-Port setzen (Die IPv6 des Pi muss in eckigen Klammern stehen):

```
nohup sudo socat UDP4-LISTEN:17777,fork,reuseaddr UDP6:[IPv6-Raspberry-Pi]:17777 &
nohup sudo socat UDP4-LISTEN:27015,fork,reuseaddr UDP6:[IPv6-Raspberry-Pi]:27015 &
```

> Hinweis: Die Ausgabe nohup: ignoring input and appending output to 'nohup.out' ist normal und bedeutet, dass der Prozess erfolgreich im Hintergrund läuft.

>

> If you want to debug incoming and outcoming trafic:

```
sudo tcpdump -i any udp port 27015 -n
```

---

4. Raspberry Pi Setup (IPv6 -> Lokale IPv4)

Der Raspberry Pi empfängt die IPv6-Pakete von Oracle und leitet sie als lokale IPv4-Pakete an den Game-Server weiter.

Voraussetzungen auf dem Pi / Heimnetz

- In der FritzBox muss eine "IPv6 Portfreigabe" (oder "Ping6 erlauben") für den Raspberry Pi für die entsprechenden UDP-Ports (`17777`, `27015`) existieren.
- `socat` muss auf dem Pi installiert sein: `sudo apt install socat`

Portmapping mit socat (Raspberry Pi)

Alte socat-Prozesse beenden:

```
sudo pkill socat
```

Neue Routen setzen (Achtung: Hier steht `UDP6-LISTEN` vorne. Die lokale IPv4 benötigt keine eckigen Klammern):

```
nohup sudo socat UDP6-LISTEN:17777,fork,reuseaddr UDP4:192.168.x.x:17777 &
nohup sudo socat UDP6-LISTEN:27015,fork,reuseaddr UDP4:192.168.x.x:27015 &
```

---

5. Neue Game-Server hinzufügen (Checkliste)

Wenn ein neues Spiel (z.B. Minecraft) auf Port `25565 TCP` gehostet werden soll:

1. Pterodactyl-Binding auf `0.0.0.0` setzen.
2. Oracle Cloud Dashboard: Port `25565 TCP` Ingress freigeben.
3. Oracle Terminal: iptables INPUT für `25565 TCP` erlauben.
4. Oracle Terminal: Neuen socat-Befehl ausführen (Achtung: Bei TCP das `UDP4` zu `TCP4` und `UDP6` zu `TCP6` ändern).
5. FritzBox: IPv6 Port `25565 TCP` für den Raspberry Pi freigeben.
6. Raspberry Pi Terminal: Neuen socat-Befehl ausführen (TCP6 -> TCP4).

Revision #23
Created 2026-03-14 22:39:15 UTC by Penry
Updated 2026-06-06 23:42:12 UTC by Penry