Introduzione a Wireguard, una VPN moderna ispirata da SSH

Simone Degiacomi

hack.bs

WireGuard: Next Generation Kernel Network Tunnel

www.wireguard.com

Jason A. Donenfeld
jason@zx2c4.com

Draft Revision

Abstract

WireGuard is a secure network tunnel, operating at layer 3, implemented as a kernel virtual network interface for Linux, which aims to replace both IPsec for most use cases, as well as popular user space and/or TLS-based solutions like OpenVPN, while being more secure, more performant, and easier to use. The virtual tunnel interface is based on a proposed fundamental principle of secure tunnels: an association between a peer public key and a tunnel source IP address. It uses a single round trip key exchange, based on NoiseIK, and handles all session creation transparently to the user using a novel timer state machine mechanism. Short pre-shared static keys—Curve25519 points—are used for mutual authentication in the style of OpenSSH. The protocol provides strong perfect forward secrecy in addition to a high degree of identity hiding. Transport speed is accomplished using ChaCha20Poly1305 authenticated-encryption for encapsulation of packets in UDP. An improved take on IP-binding cookies is used for mitigating denial of service attacks, improving greatly on IKEv2 and DTLS's cookie mechanisms to add encryption and authentication. The overall design allows for allocating no resources in response to received packets, and from a systems perspective, there are multiple interesting Linux implementation techniques for queues and parallelism. Finally, WireGuard can be simply implemented for Linux in less than 4,000 lines of code, making it easily audited and verified.

Cosa vediamo oggi

1. Cos'è Wireguard e per cosa posso usarlo?
2. Come faccio a usarlo?
3. Cryptokey routing

Cosa è Wireguard

- Protocollo e implementazione, non un servizio
- Paragonabile a IPsec e OpenVPN

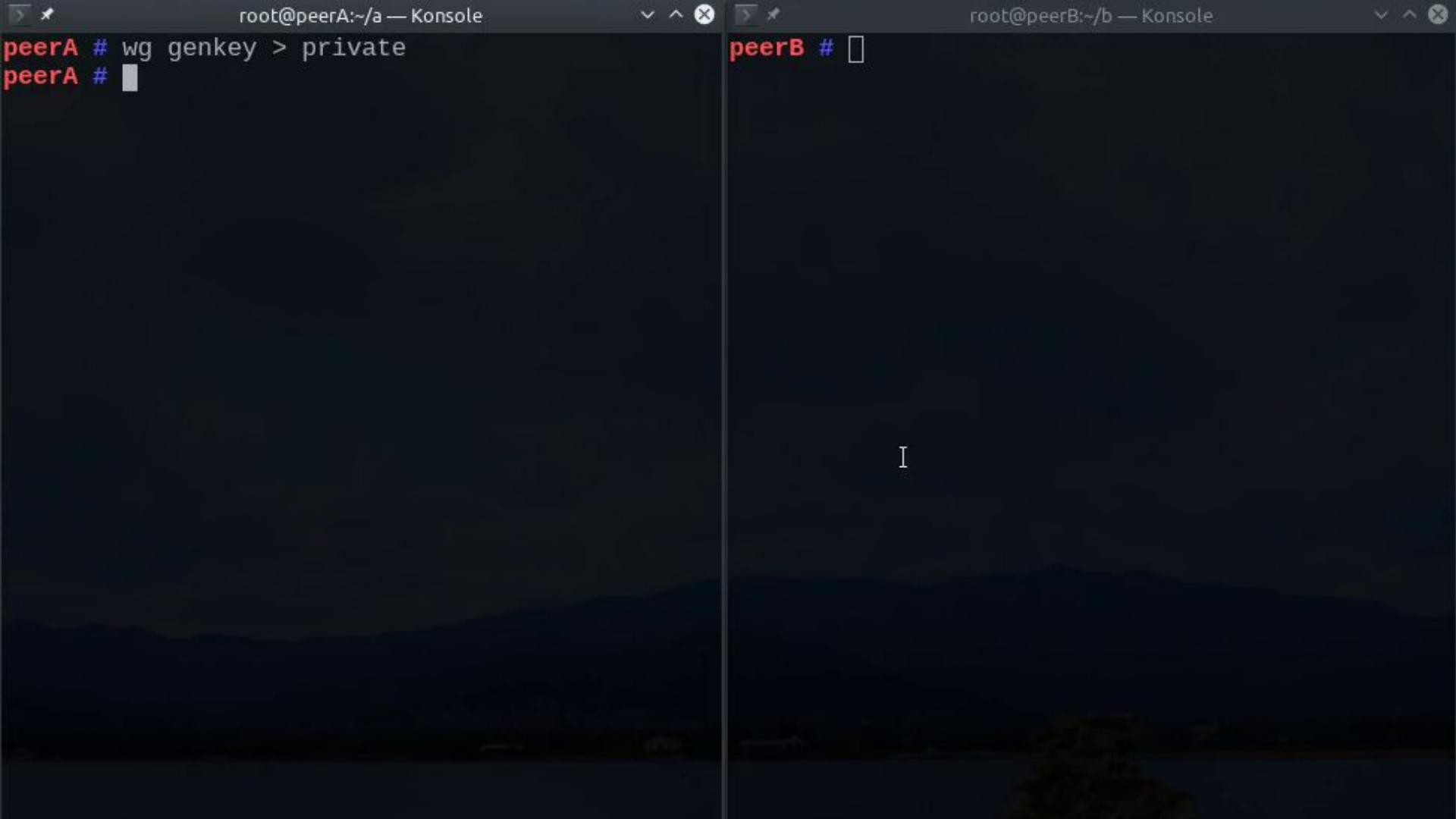
Cosa ci posso fare?

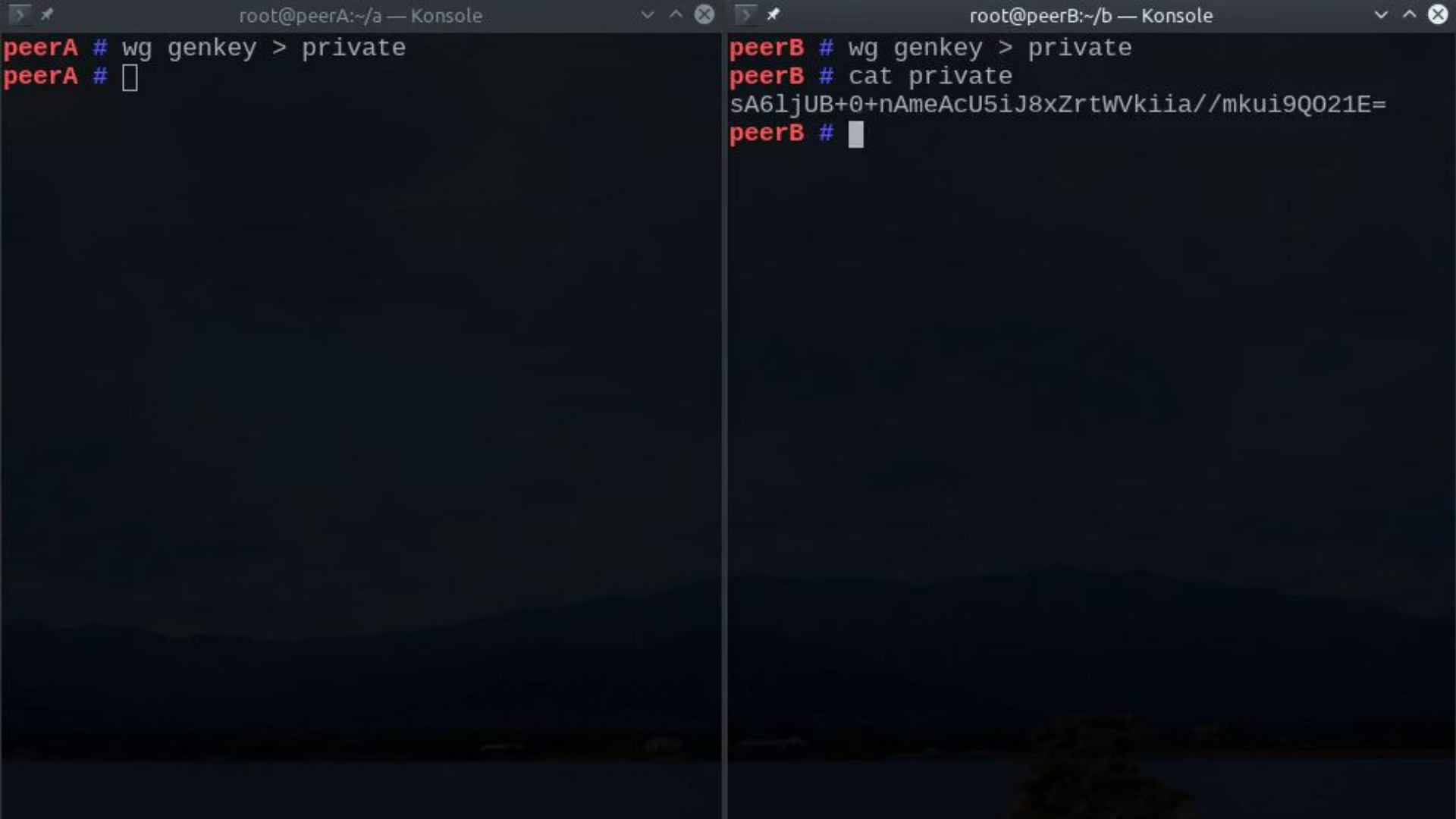
- Estendere una rete locale oltre le mura fisiche
  - accedere al tuo pc, fuori di casa, senza esporlo direttamente ad internet
  - creare dei ponti tra diversi edifici
  - garantire accesso remoto a macchine deployate chissà dove

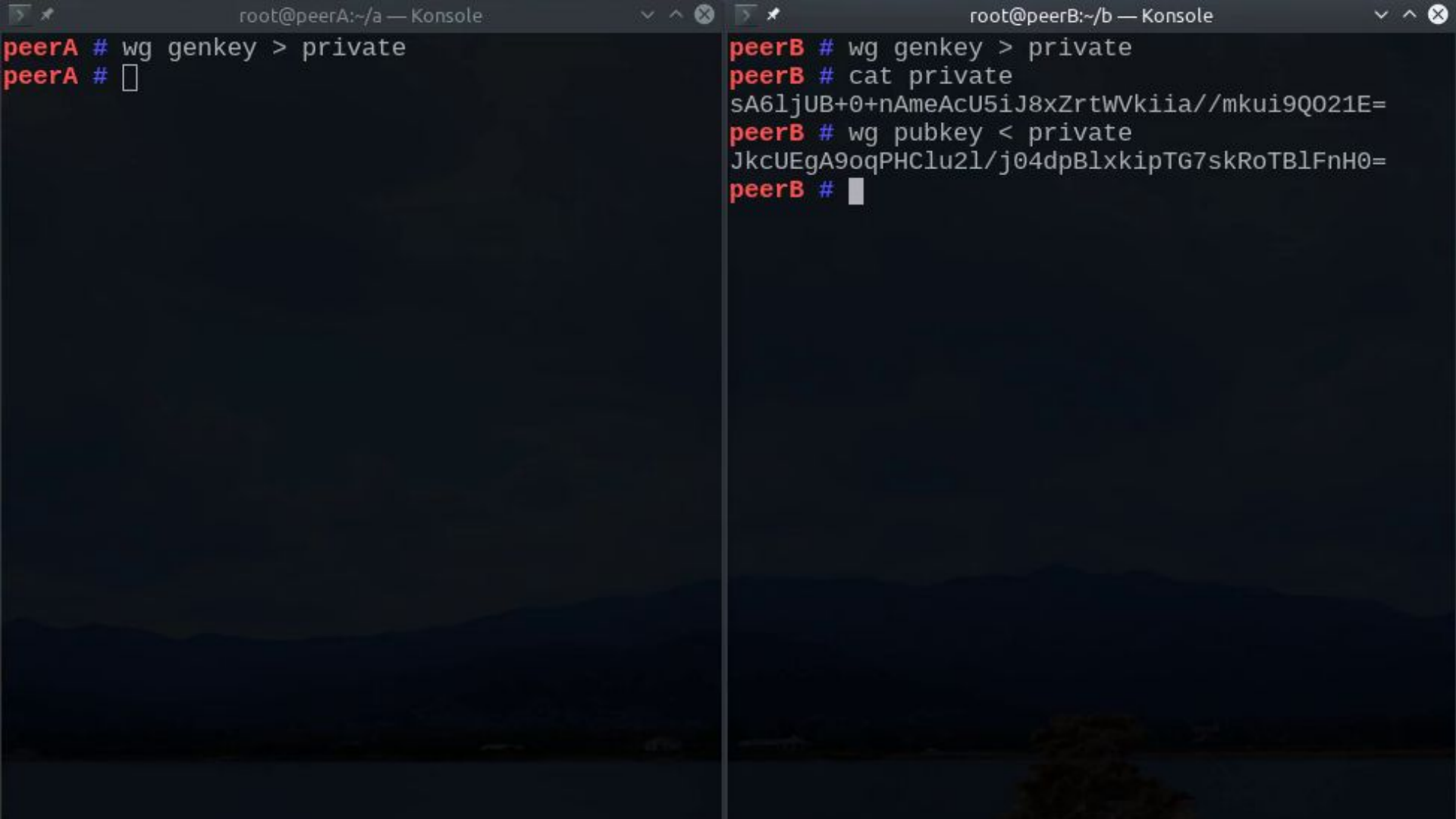
Cosa ci posso fare?

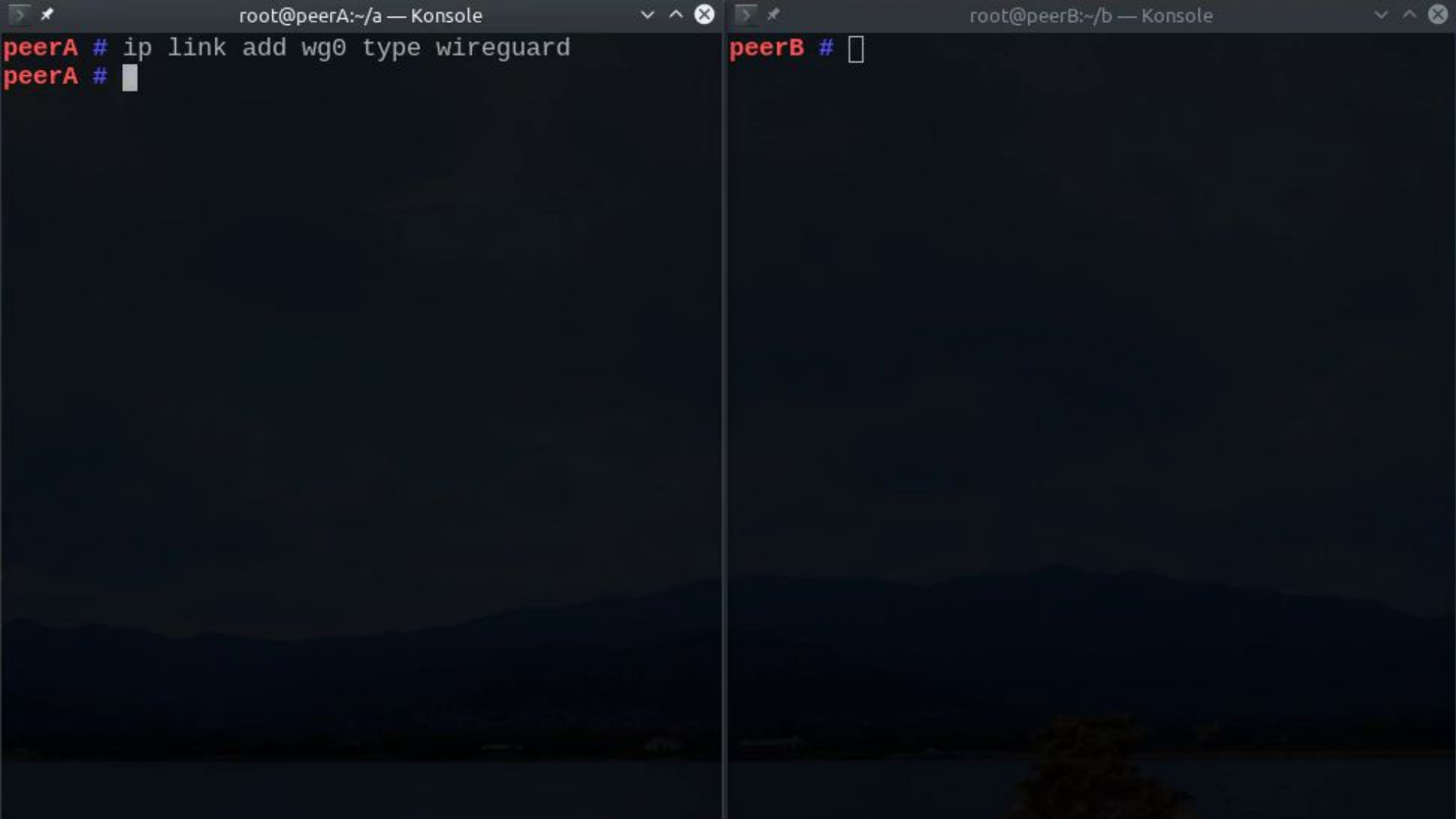
- Privacy
  - Puoi impedire al tuo ISP di tracciare a quali server ti connetti
- IP Spoofing
  - Puoi connetterti a server utilizzando un IP diverso da quello assegnato dal tuo ISP

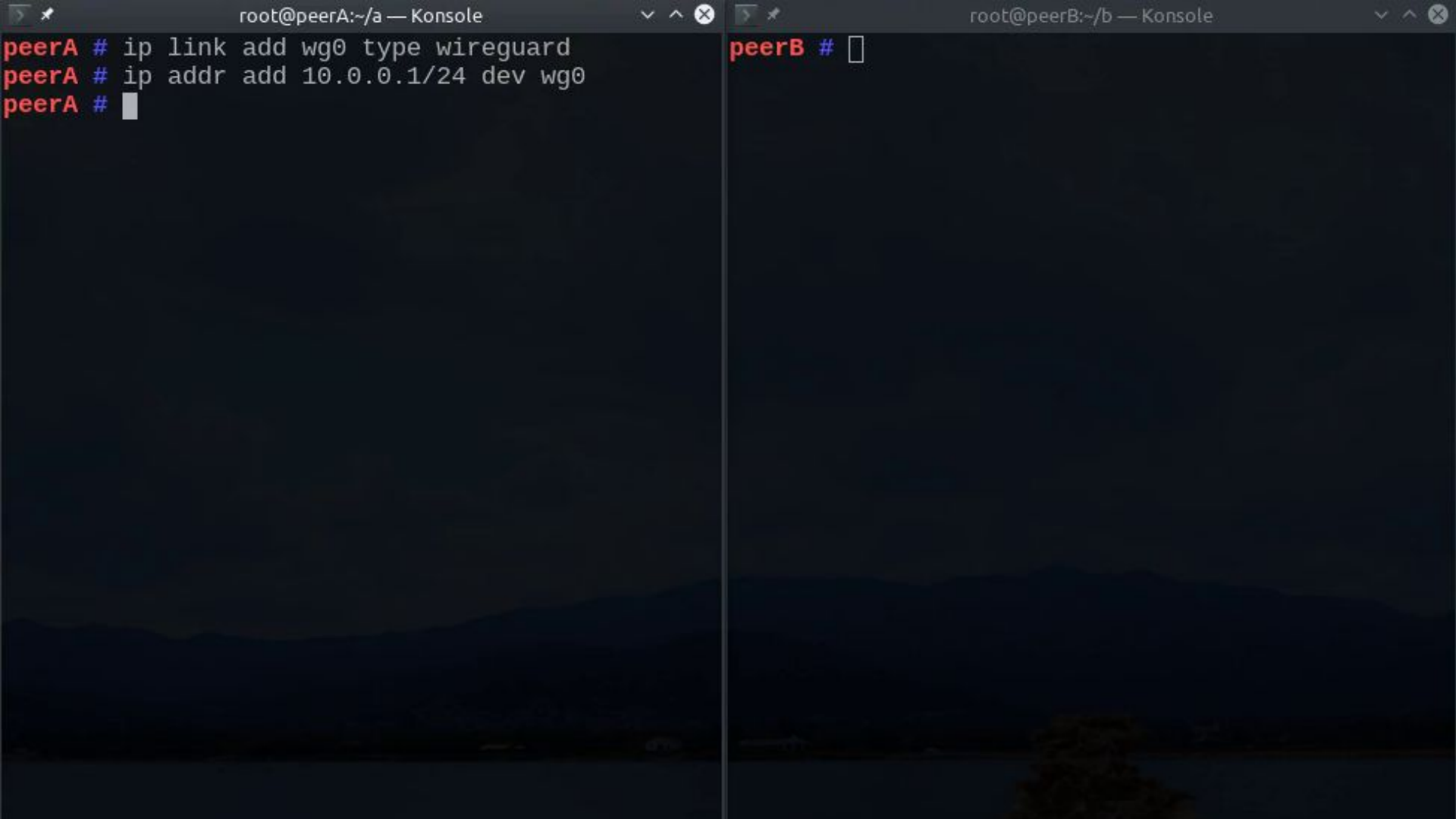
Come lo uso?

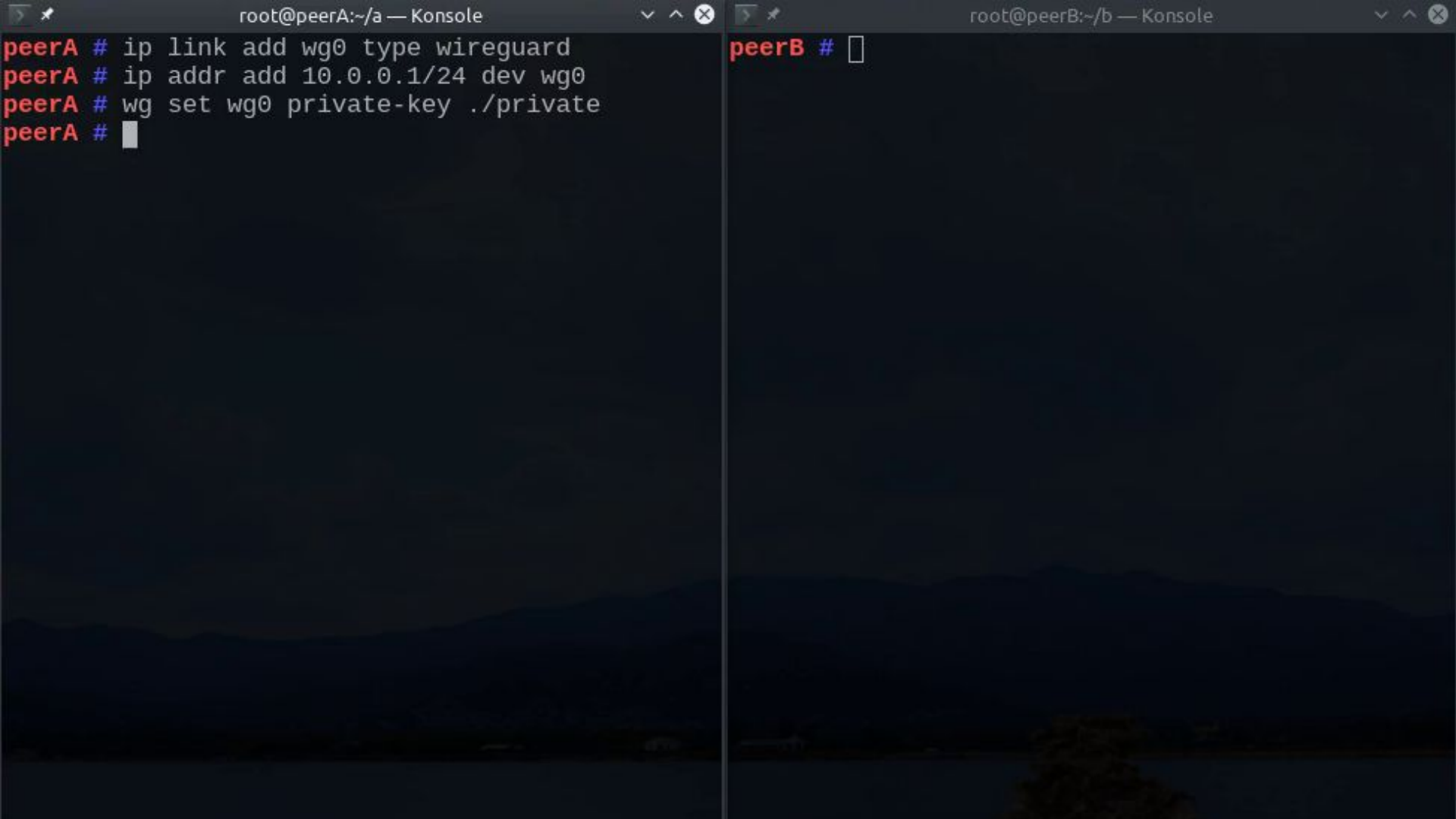

root@peerA:~/a — Konsole
peerA # wg genkey > private
peerA #
root@peerB:~/b — Konsole
peerB #

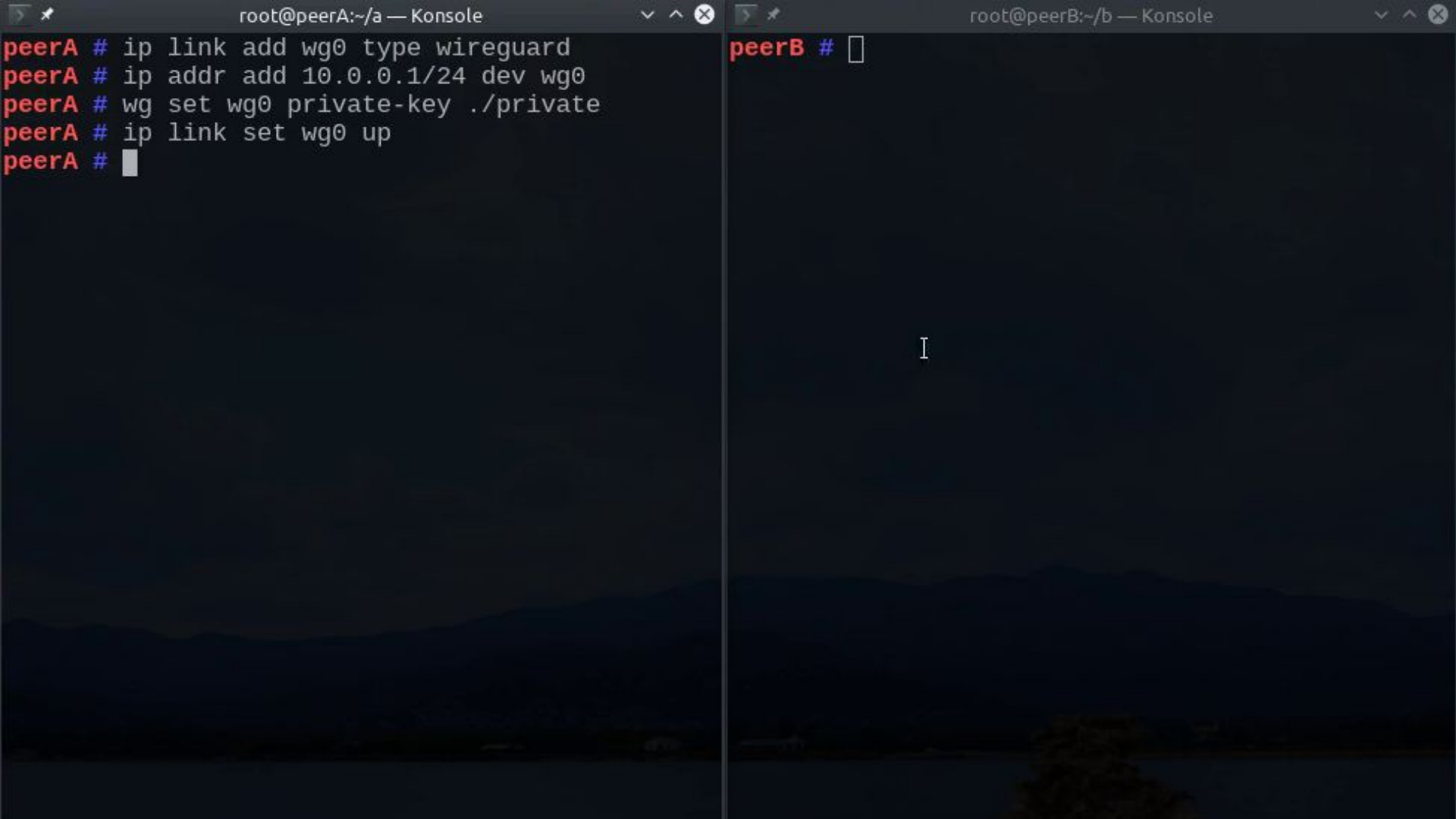

root@peerA:~/a — Konsole
peerA # wg genkey > private
peerA #
root@peerB:~/b — Konsole
peerB # wg genkey > private
peerB # cat private
sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
peerB #

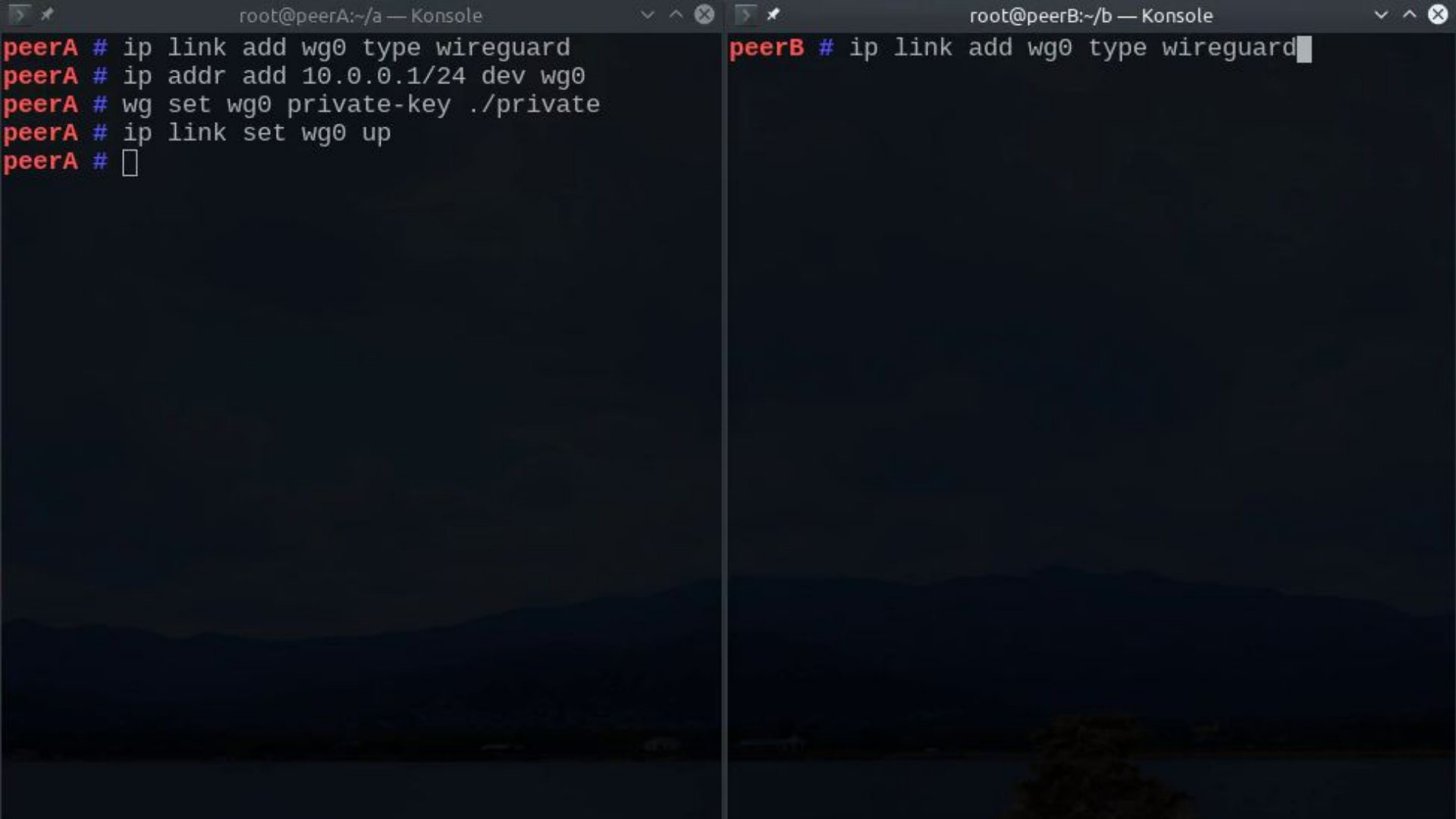

root@peerA:~/a — Konsole
peerA # wg genkey > private
peerA #
root@peerB:~/b — Konsole
peerB # wg genkey > private
peerB # cat private
sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
peerB # wg pubkey < private
JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
peerB #

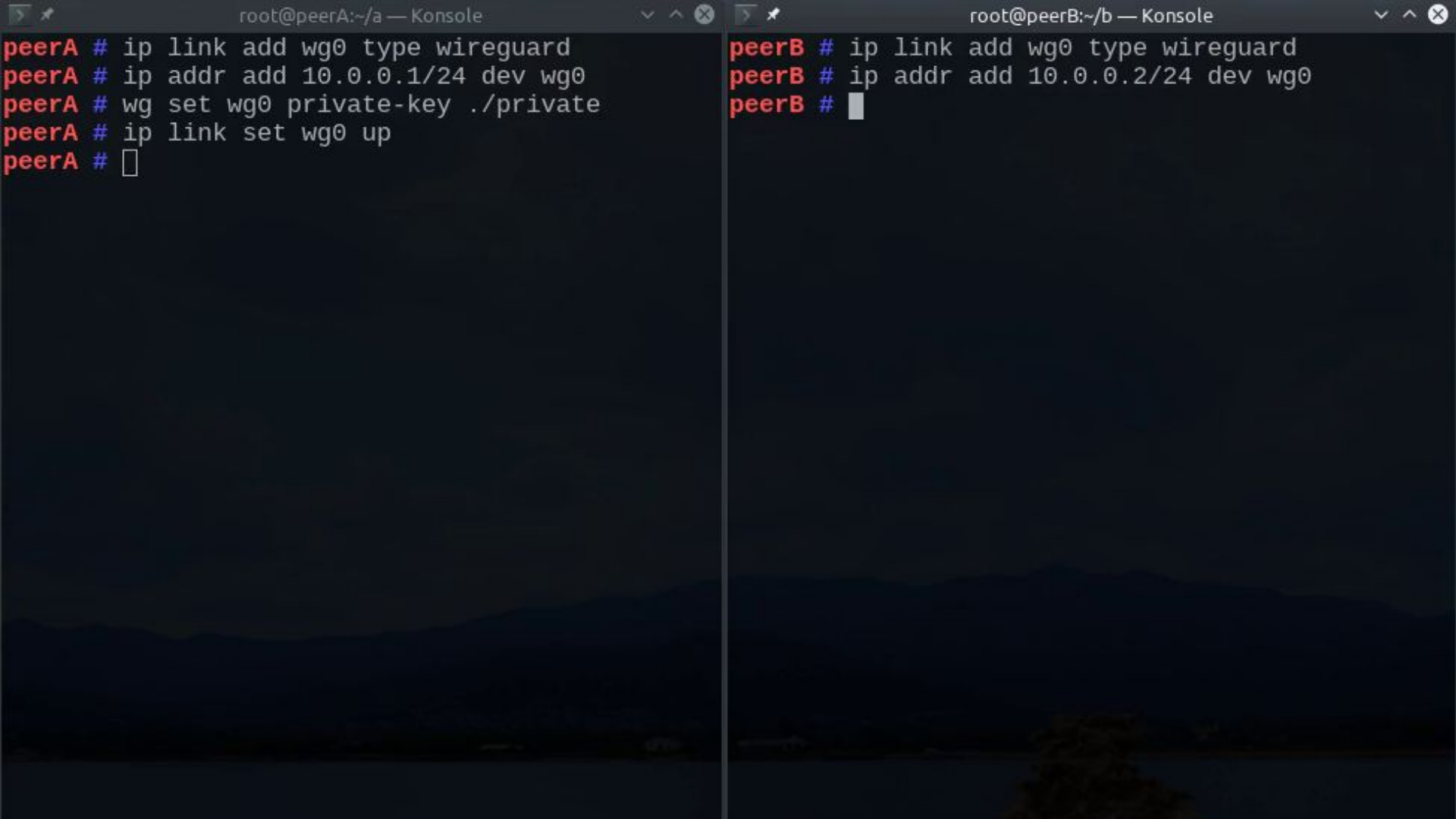

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA #
root@peerB:~/b — Konsole
peerB #

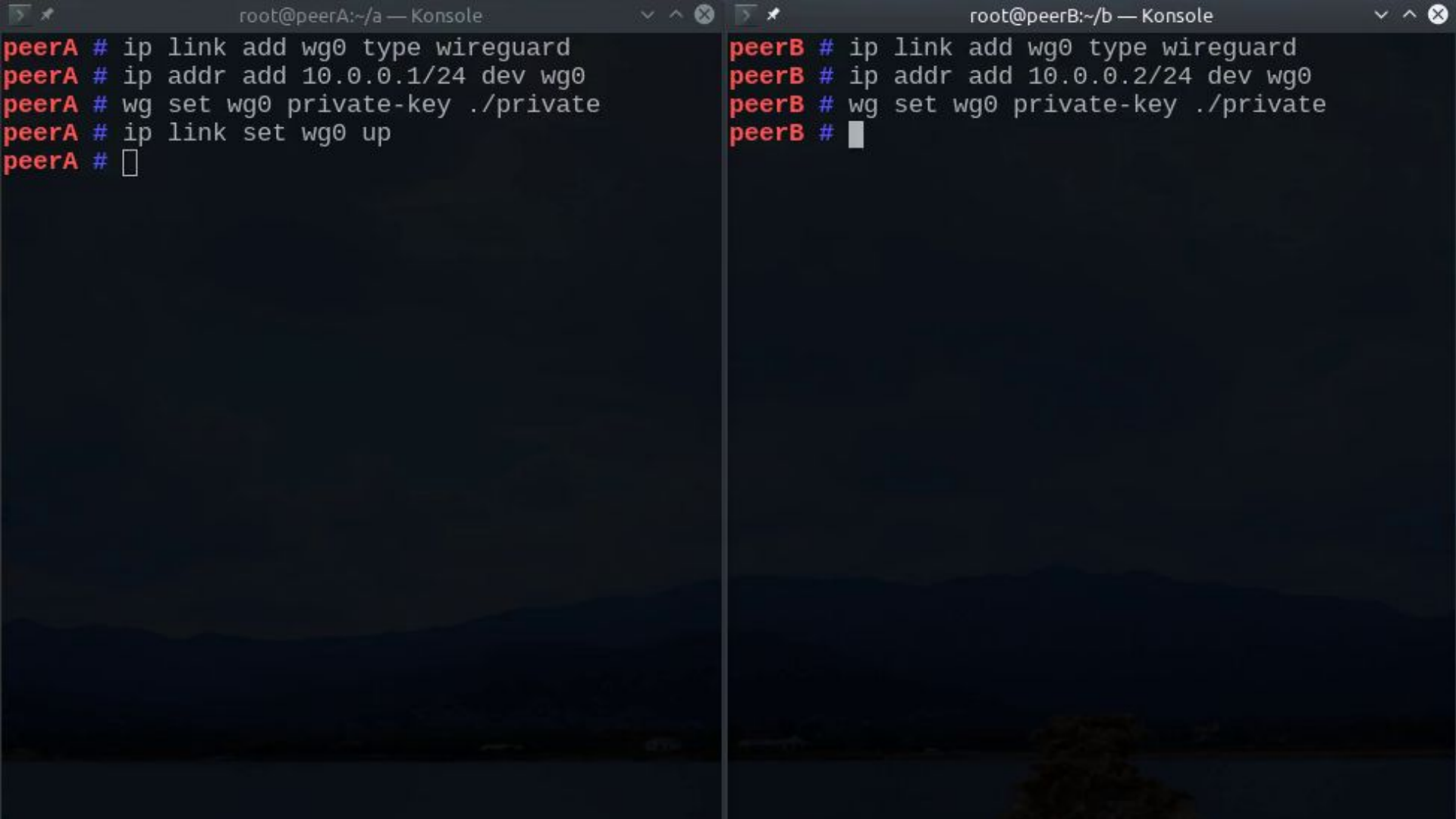

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA #
root@peerB:~/b — Konsole
peerB #

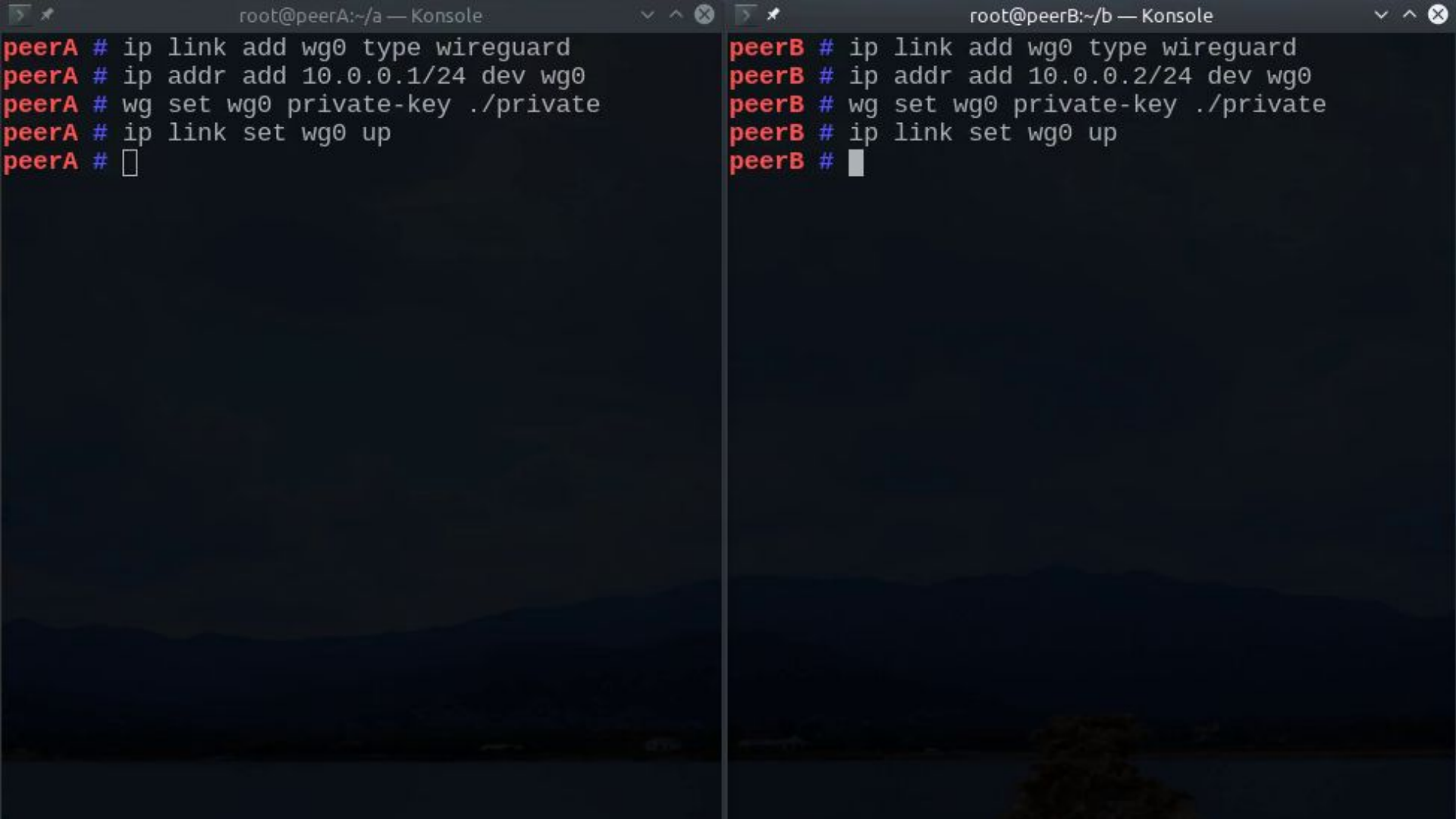

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA #
root@peerB:~/b — Konsole
peerB #

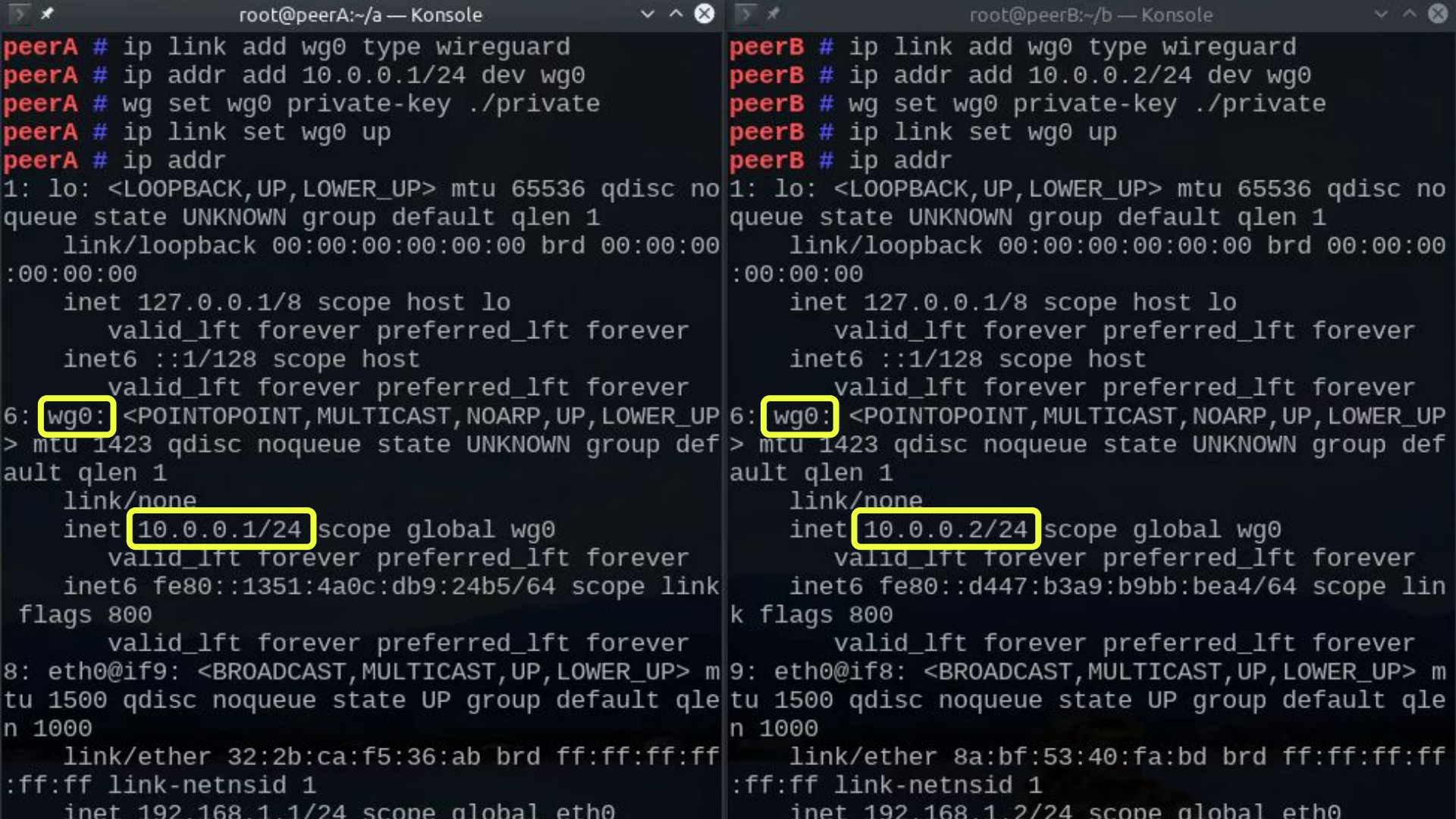

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA #
root@peerB:~/b — Konsole
peerB #

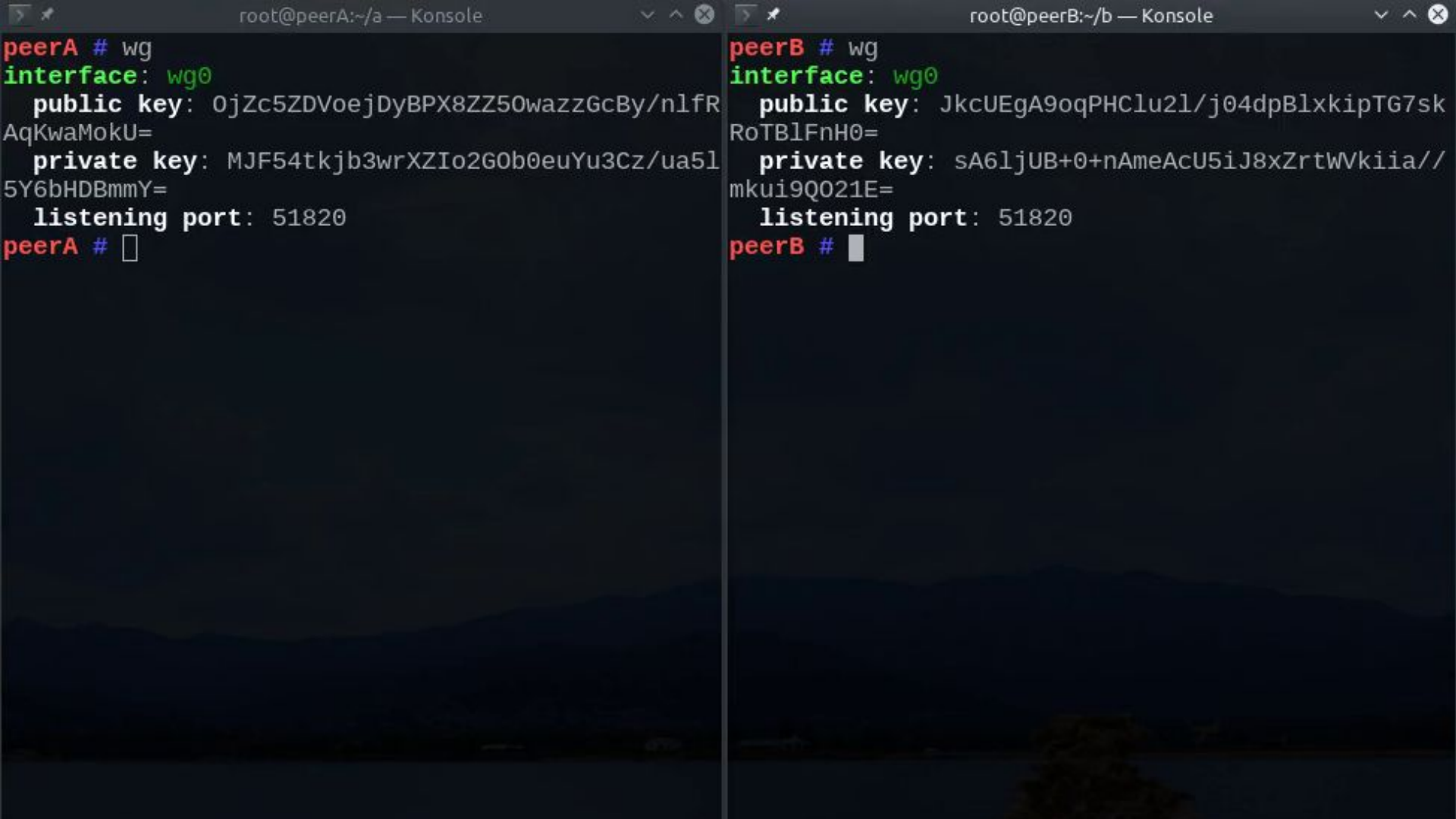

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA #
root@peerB:~/b — Konsole
peerB # ip link add wg0 type wireguard

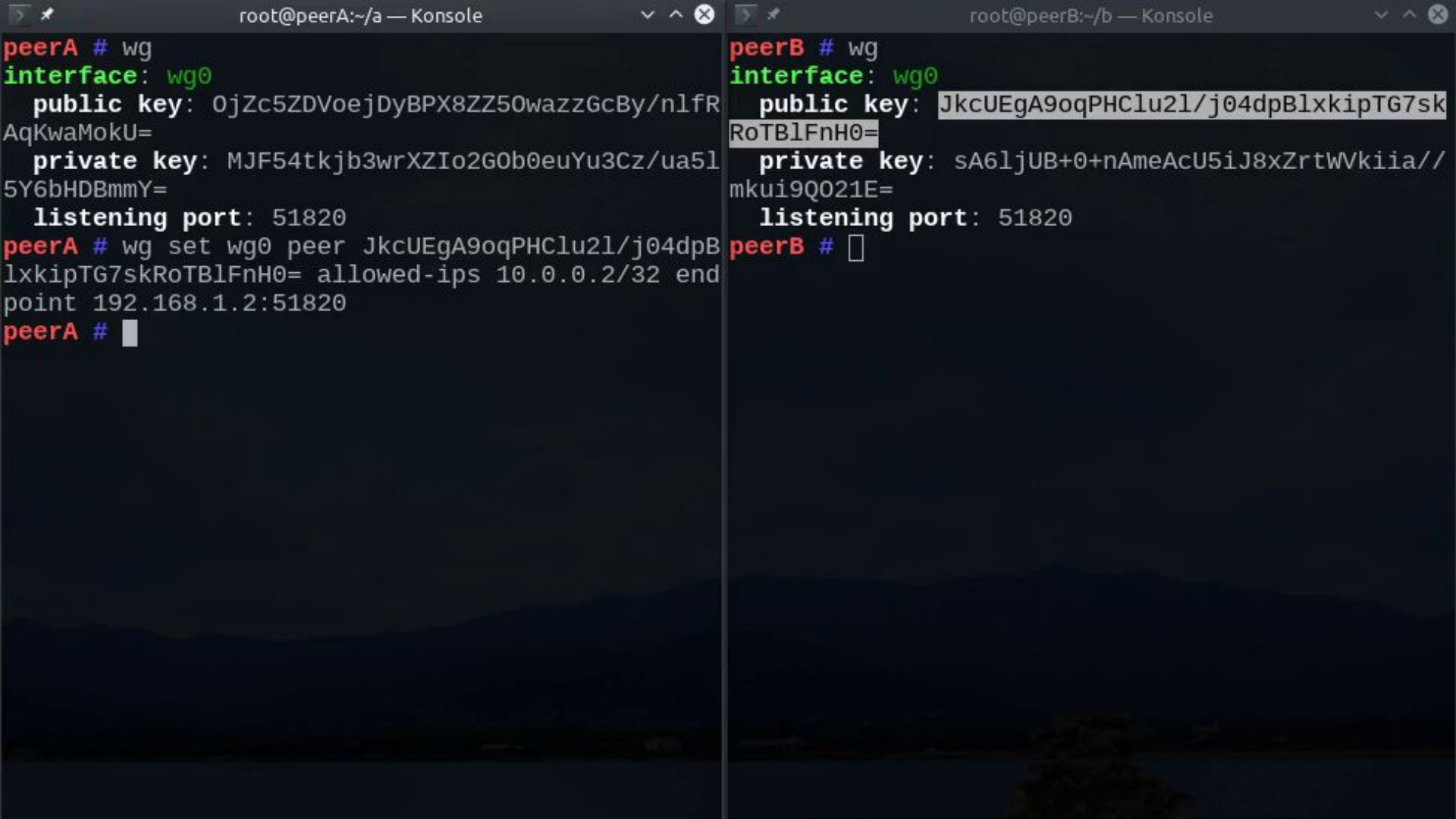

root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA #
root@peerB:~/b — Konsole
peerB # ip link add wg0 type wireguard
peerB # ip addr add 10.0.0.2/24 dev wg0
peerB #


root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA #
root@peerB:~/b — Konsole
peerB # ip link add wg0 type wireguard
peerB # ip addr add 10.0.0.2/24 dev wg0
peerB # wg set wg0 private-key ./private
peerB #


root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA #
root@peerB:~/b — Konsole
peerB # ip link add wg0 type wireguard
peerB # ip addr add 10.0.0.2/24 dev wg0
peerB # wg set wg0 private-key ./private
peerB # ip link set wg0 up
peerB #


root@peerA:~/a — Konsole
peerA # ip link add wg0 type wireguard
peerA # ip addr add 10.0.0.1/24 dev wg0
peerA # wg set wg0 private-key ./private
peerA # ip link set wg0 up
peerA # ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc no
queue state UNKNOWN group default qlen 1
link/loopback 00:00:00:00:00:00 brd 00:00:00
:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
6: wg0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP
> mtu 1423 qdisc noqueue state UNKNOWN group def
ault qlen 1
link/none
inet 10.0.0.1/24 scope global wg0
valid_lft forever preferred_lft forever
inet6 fe80::1351:4a0c:db9:24b5/64 scope link
flags 800
valid_lft forever preferred_lft forever
8: eth0@if9: <BROADCAST,MULTICAST,UP,LOWER_UP> m
tu 1500 qdisc noqueue state UP group default qle
n 1000
link/ether 32:2b:ca:f5:36:ab brd ff:ff:ff:ff
:ff:ff link-netnsid 1
inet 192.168.1.1/24 scope global eth0
root@peerB:~/b — Konsole
peerB # ip link add wg0 type wireguard
peerB # ip addr add 10.0.0.2/24 dev wg0
peerB # wg set wg0 private-key ./private
peerB # ip link set wg0 up
peerB # ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc no
queue state UNKNOWN group default qlen 1
link/loopback 00:00:00:00:00:00 brd 00:00:00
:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
6: wg0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP
> mtu 1423 qdisc noqueue state UNKNOWN group def
ault qlen 1
link/none
inet 10.0.0.2/24 scope global wg0
valid_lft forever preferred_lft forever
inet6 fe80::d447:b3a9:b9bb:bea4/64 scope lin
k flags 800
valid_lft forever preferred_lft forever
9: eth0@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> m
tu 1500 qdisc noqueue state UP group default qle
n 1000
link/ether 8a:bf:53:40:fa:bd brd ff:ff:ff:ff
:ff:ff link-netnsid 1
inet 192.168.1.2/24 scope global eth0


root@peerA:~/a — Konsole
peerA # wg
interface: wg0
public key: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
private key: MJF54tkjb3wrXZIo2GOb0euYu3Cz/ua5l5Y6bHDBmmY=
listening port: 51820
peerA #
root@peerB:~/b — Konsole
peerB # wg
interface: wg0
public key: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
private key: sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
listening port: 51820
peerB #

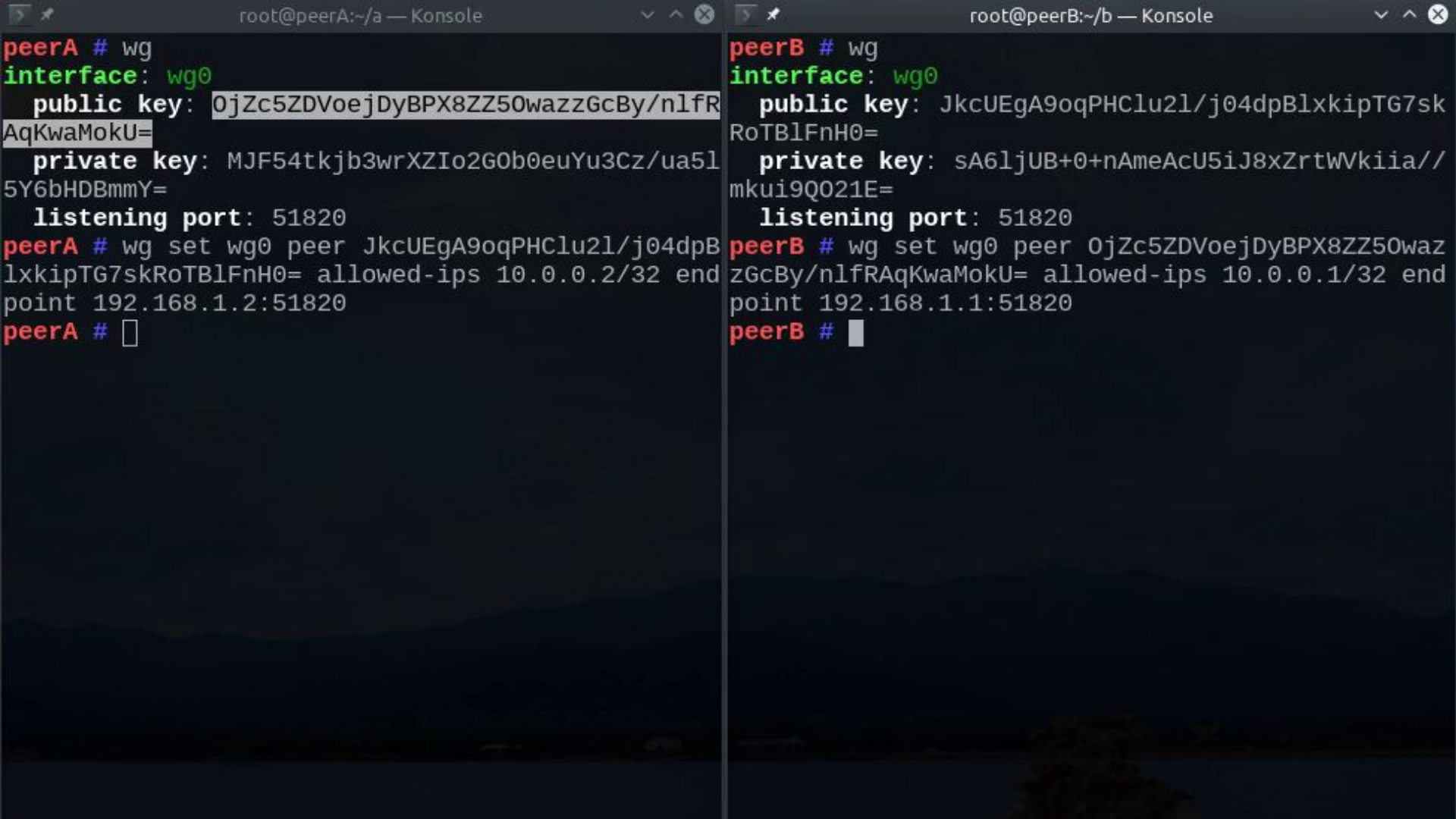

root@peerA:~/a — Konsole
peerA # wg
interface: wg0
public key: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
private key: MJF54tkjb3wrXZIo2GOb0euYu3Cz/ua5l5Y6bHDBmmY=
listening port: 51820
peerA # wg set wg0 peer JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0= allowed-ips 10.0.0.2/32 endpoint 192.168.1.2:51820
peerA #
root@peerB:~/b — Konsole
peerB # wg
interface: wg0
public key: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
private key: sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
listening port: 51820
peerB #

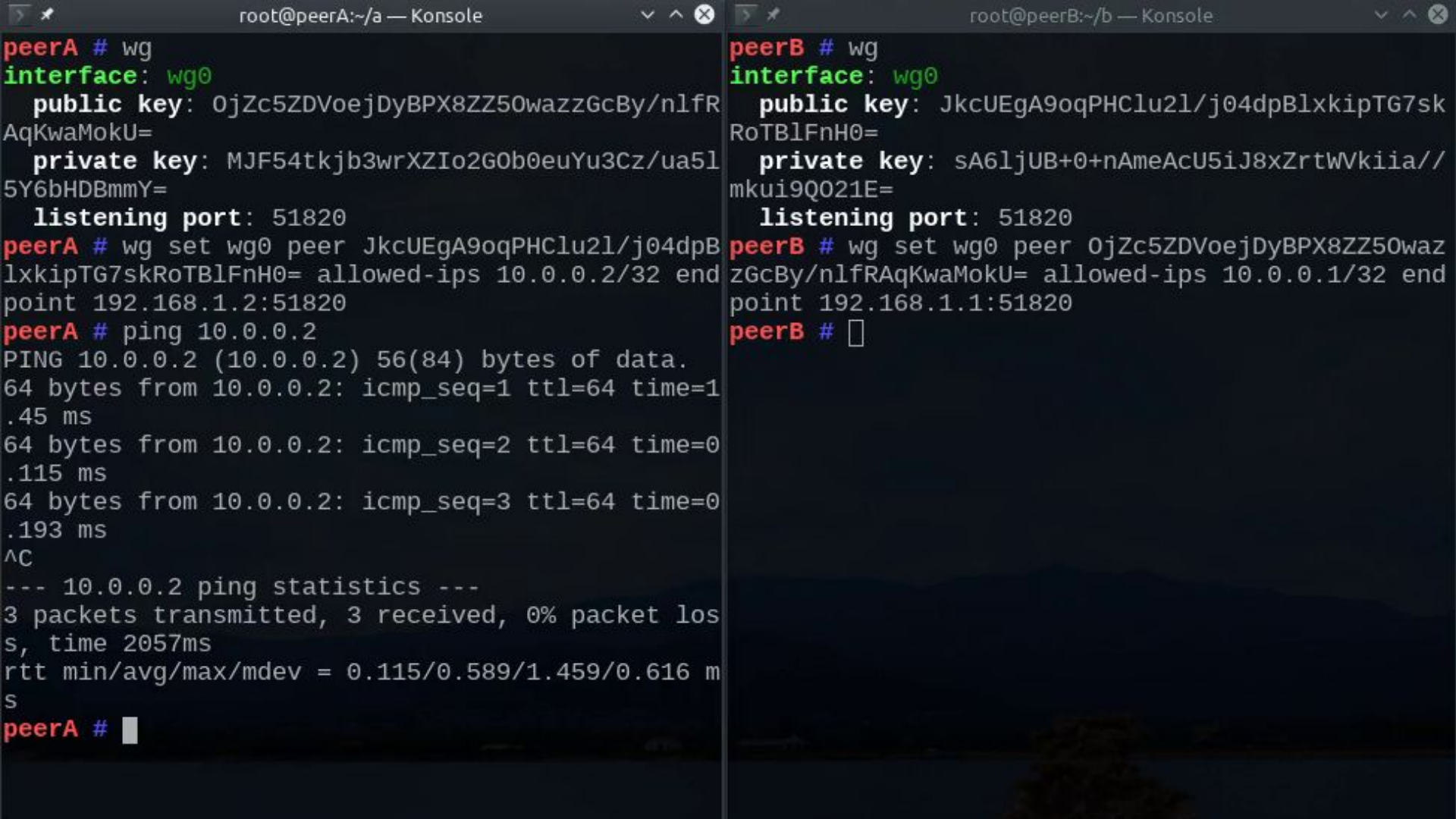
Cryptokey routing

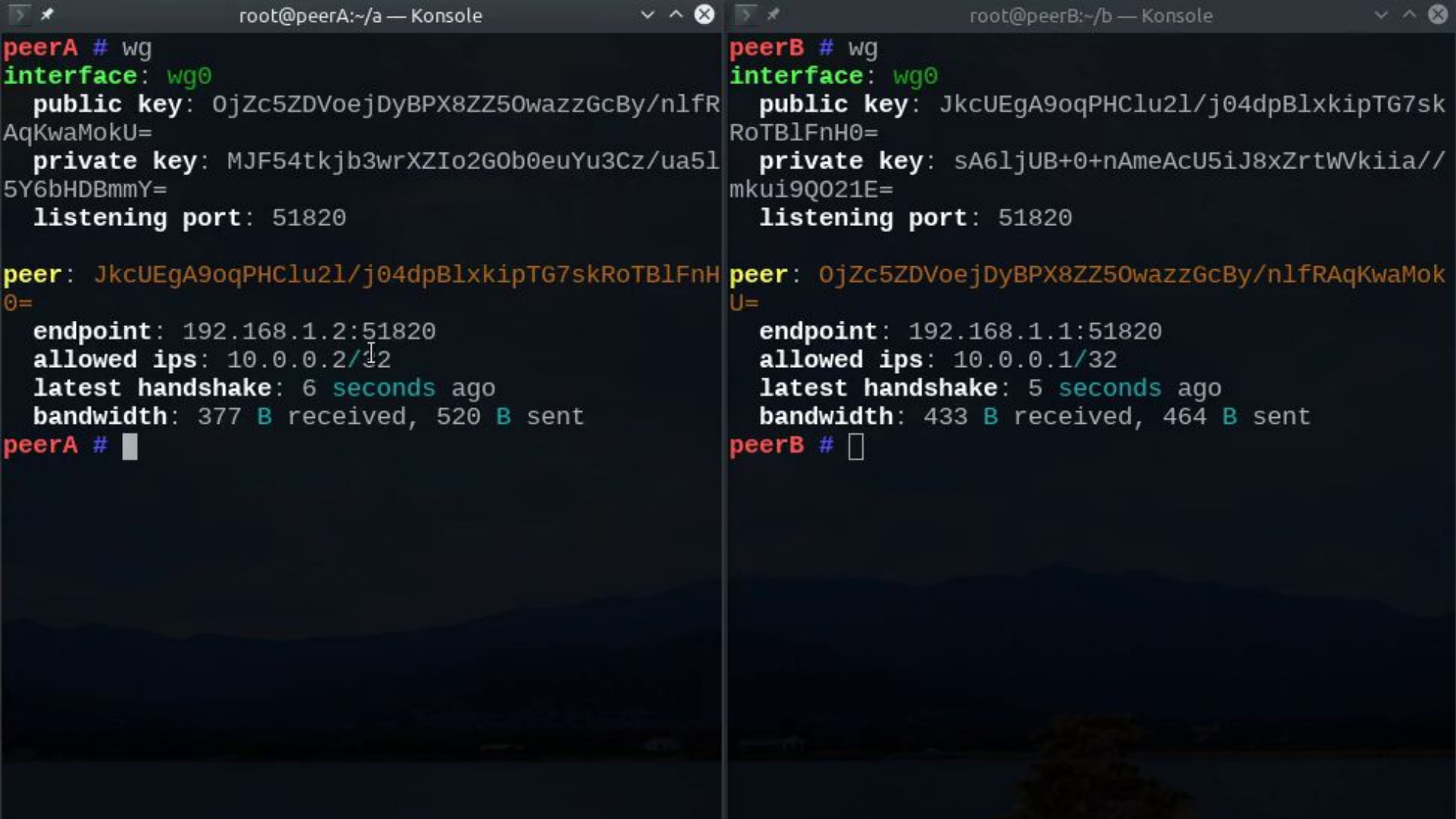
Chiave pubblica peer	Allowed IPs
JkcUEgA9oqPHCli2l/j04…	10.0.0.2/32

Cryptokey routing

Chiave pubblica peer	Allowed IPs
JkcUEgA9oqPHCli2l/j04…	10.0.0.2/32, 10.0.2.0/24
NTj4hQkNHv4lEBMO0eNk…	10.10.1.3/32, 192.168.0.0/16


root@peerA:~/a — Konsole
peerA # wg
interface: wg0
public key: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
private key: MJF54tkjb3wrXZIo2GOb0euYu3Cz/ua5l5Y6bHDBmmY=
listening port: 51820
peerA # wg set wg0 peer JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0= allowed-ips 10.0.0.2/32 endpoint 192.168.1.2:51820
peerA #
root@peerB:~/b — Konsole
peerB # wg
interface: wg0
public key: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
private key: sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
listening port: 51820
peerB # wg set wg0 peer OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU= allowed-ips 10.0.0.1/32 endpoint 192.168.1.1:51820
peerB #


root@peerA:~/a — Konsole
peerA # wg
interface: wg0
public key: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
private key: MJF54tkjb3wrXZIo2GOb0euYu3Cz/ua5l5Y6bHDBmmY=
listening port: 51820
peerA # wg set wg0 peer JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0= allowed-ips 10.0.0.2/32 endpoint 192.168.1.2:51820
peerA # ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=1.45 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.115 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.193 ms
^C
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2057ms
rtt min/avg/max/mdev = 0.115/0.589/1.459/0.616 ms
peerA #
root@peerB:~/b — Konsole
peerB # wg
interface: wg0
public key: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
private key: sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
listening port: 51820
peerB # wg set wg0 peer OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU= allowed-ips 10.0.0.1/32 endpoint 192.168.1.1:51820
peerB #


root@peerA:~/a — Konsole
peerA # wg
interface: wg0
public key: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
private key: MJF54tkjb3wrXZIo2GOb0euYu3Cz/ua5l5Y6bHDBmmY=
listening port: 51820
peer: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
endpoint: 192.168.1.2:51820
allowed ips: 10.0.0.2/32
latest handshake: 6 seconds ago
bandwidth: 377 B received, 520 B sent
peerA #
root@peerB:~/b — Konsole
peerB # wg
interface: wg0
public key: JkcUEgA9oqPHClu2l/j04dpBlxkipTG7skRoTBlFnH0=
private key: sA6ljUB+0+nAmeAcU5iJ8xZrtWVkiia//mkui9QO21E=
listening port: 51820
peer: OjZc5ZDVoejDyBPX8ZZ5OwazzGcBy/nlfRAqKwaMokU=
endpoint: 192.168.1.1:51820
allowed ips: 10.0.0.1/32
latest handshake: 5 seconds ago
bandwidth: 433 B received, 464 B sent
peerB #

wg-quick

Esempio

```
/etc/wireguard/wg0.conf (peerA)

[Interface]
PrivateKey = UBuL5i5u1Naq00im1e0fWa65uP1JZKpHVfYaqnOJk2E=
Address = 10.0.0.1/24
ListenPort = 51820

[Peer]
PublicKey = Jo0ATPXYLbzI9oF3DeKkltT5KGp4KyGSwfrrcYkdqVo=
AllowedIps = 10.0.0.2/32
Endpoint = 192.168.1.2:51820
```

```
/etc/wireguard/wg0.conf (peerB)

[Interface]
PrivateKey = 0LN0GttJCw6rmbDCYoJS7g7GKAEsRdfjItHYownJTmQ=
Address = 10.0.0.2/24
ListenPort = 51820

[Peer]
PublicKey = 8yLG8c99oS1PGzzLDGXoGbywtnb2FrYSFEVSlLj3Axw=
AllowedIPs = 10.0.0.1/32
Endpoint = 192.168.1.1:51820
```

Tunnelling di tutto il traffico

Va aggiunta qualche altra riga per poter effettuare un forward completo di tutto il traffico

- abilitare NAT
- modificare tabella di routing

Domande?