# Como utilizar o comando ss

# Simular o tcpview

```
watch -n 1 "sudo ss -tunap"
```

# Portas específicas

```
ss -tunap sport = :80
ss -tunap '( dport = :80 or dport = :443 )'
ss -tunap '( dport = :80 or dport = :443 or dport = :8080 )'
ss -tunap '( dport >= :8000 and dport <= :9000 )'
ss -4 -tunap '( dport = :443 )'
```

- `dport = :443`: Filtra conexões com **porta de destino 443** (HTTPS).
- `-4`: IPv4 (opcional, remova se quiser IPv6 também).
- `-t`: TCP (já que HTTPS usa TCP).
- `-u`: UDP (não necessário aqui, mas pode ser incluído se quiser).
- `-n`: Exibe IPs numéricos.
- `-a`: Mostra todas as conexões (ativas e em escuta).
- `-p`: Exibe processos associados.

# Filtrar por estado (ex.: ESTABLISHED, LISTEN)

```
ss -t state established   # Conexões ativas
ss -t state listening     # Portas em escuta (LISTEN)
```

# Fazendo um filtro por IP

```
ss -tunap | grep '192.168.1.100'  # Filtra por IP
```

# Exibindo apenas IPv4 ou IPv6

```
ss -4 -tunap  # Filtra apenas conexões IPv4
ss -6 -tunap  # Filtra apenas conexões IPv6
```

## Alternativa com filtro de família de endereços (mais explícito):

No `ss`, você também pode usar `-f inet` (IPv4) ou `-f inet6` (IPv6):

```
ss -f inet -tunap    # IPv4
ss -f inet6 -tunap   # IPv6
```

# Excluindo algumas portas

```
ss -4 -tunap '( dport != :22 )'
ss -4 -tunap '( sport != :22 and dport != :22 )'
ss -4 -tunap | grep -v ':22'
ss -4 -tunap '( dport != :22 and dport != :80 and dport != :443 )'
```

# Testes feito nos meus servidores

```
# No real time
ss -4 -tno state established '( dport = :80 or dport = :443 )' dst 131.72.154.130
# Real time
watch -n 1 "ss -4 -tno state established '( dport = :80 or dport = :443 )' dst 131.72.154.130"

# No real time
ss -4 -tno state established '( dport = :80 or dport = :443 )' | grep '131.72.154.'
```

```
# Real time
watch -n 1 "ss -4 -tno state established '( dport = :80 or dport = :443 )' | grep
'131.72.154.'"
```

Nesses dois últimos exemplos com o grep, temos um efeito parecido com um filtro feitos para redes/CIDR. No nosso exemplo, um /24.

Podemos também remover as portas e obter quaisquer conexões com a faixa de rede `131.72.154.0/24`

```
# No real time
ss -4 -tno state established | grep '131.72.154.'
# Real time
watch -n 1 "ss -4 -tno state established | grep '131.72.154.'"
```

- `-4`: Filtra IPv4.
- `-t`: TCP.
- `-n`: Exibe IPs numéricos.
- `-o`: Mostra timers (opcional).
- `state established`: Apenas conexões ativas.
- `dst 131.72.154.130`: Filtra por destino específico.

# Comando final (recomendado):

```
watch -n 1 "ss -4 -antp '( dport = :80 or dport = :443 )' | grep -E 'ESTAB.*131.72.154.'"
```

- `-a`: Mostra todas as conexões (útil para `ESTABLISHED`).
- `-p`: Exibe processos associados.
- `grep -E 'ESTAB.*131.72.154.'`: Filtra conexões estabelecidas na rede desejada.

# Observações

Algumas vezes os processos não serão exibidos pq são serviços do sistema e para que aparecam, precisam ser iniciado com o `ROOT` ou então com o `SUDO`

# Soluções e Comandos Úteis:

## Execute com **sudo** para ver processos de sistema:

```
watch -n 1 "sudo ss -tunap"
```

## Use `lsof` para detalhes:

```
sudo lsof -i :22  # Verifica a porta 22 (SSH)
sudo lsof -i :68  # Verifica a porta 68 (DHCP)
```

# Explicação dos Estados:

- **ESTAB (ESTABLISHED)**: Conexão ativa entre dois hosts.
- **UNCONN (UNCONNECTED)**: Típico de UDP (não há "conexão" estabelecida).
- **LISTEN**: Porta em escuta aguardando conexões.

https://blog.mygraphql.com/en/notes/low-tec/network/tcp-inspect/

https://www.linode.com/docs/guides/ss/

https://ioflood.com/blog/ss-linux-command/

https://www.sans.org/blog/linux-incident-response-using-ss-for-network-analysis/

https://www.networkworld.com/article/971764/using-the-ss-command-on-linux-to-view-details-on-sockets.html

---

Revision #1
Created 12 October 2025 18:54:11 by Krisofferson Marini
Updated 12 October 2025 18:57:32 by Krisofferson Marini