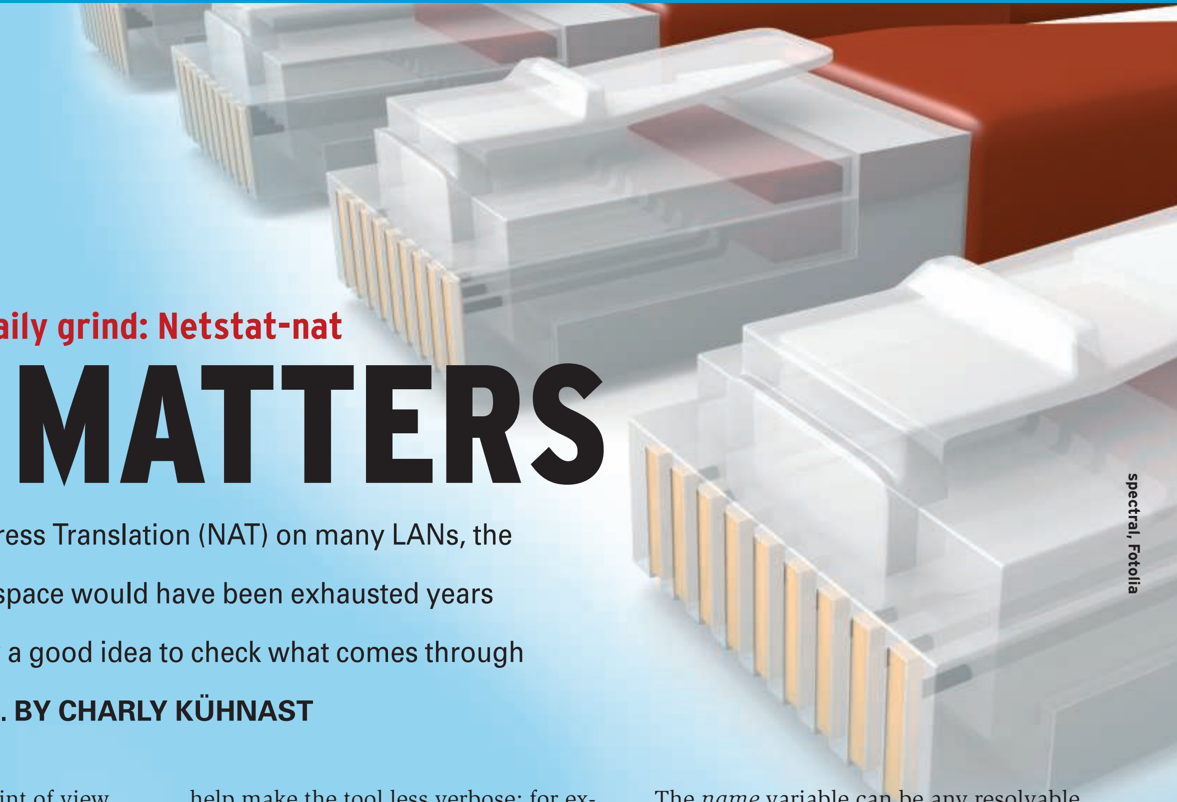


spectral, Fotolia

The sys admin's daily grind: Netstat-nat

NAT MATTERS

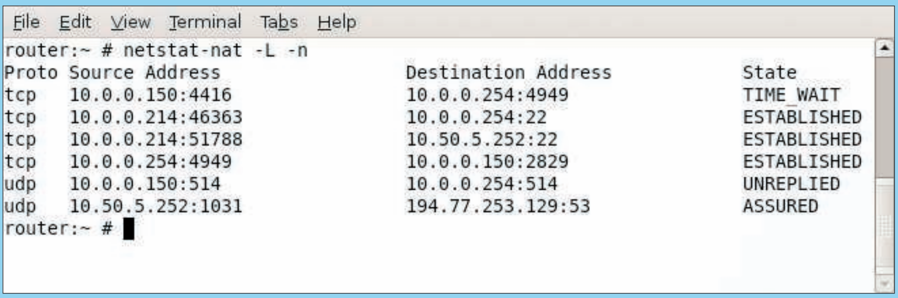
Without Network Address Translation (NAT) on many LANs, the IPv4 Internet address space would have been exhausted years ago. Still, it's probably a good idea to check what comes through your NAT connections. **BY CHARLY KÜHNAST**

From a topology point of view, LANs are the Internet's hidden flaw. Many public hosts are actually gateways behind which any number of computers with private IPs might reside. Network Address Translation makes sure that computers without a live IP can still access the Internet. However, NAT does not contribute much to visibility in networks from the admin's point of view, which is why I am happy to have Netstat-nat [1] in my toolbox.

The small C program – available in *tar.gz*, PRM, and Deb formats – shows the status of NAT connections by tapping into the connection data that iptables writes to */proc/net/ip_conntrack**. On the Internet, Netstat-nat's output can become cluttered, but a bunch of options help make the tool less verbose; for example, Netstat-nat supports a rough categorization by protocol type. Typing

```
netstat-nat -p tcp
```

for instance, hides UDP connections; this restricts the output to TCP connections. Also, setting the *-S* and *-D* options specifies whether you want to see NAT source or destination connections. A source NAT (SNAT) converts internal addresses, which are typically in the RFC 1918 area such as 192.168.0.0/16, to valid public IP addresses. DSL routers for small offices and home networks use source NAT. Destination NAT (DNAT) works the other way around.



```
router:~ # netstat-nat -L -n
Proto Source Address              Destination Address        State
tcp   10.0.0.150:4416             10.0.0.254:4949            TIME_WAIT
tcp   10.0.0.214:46363            10.0.0.254:22              ESTABLISHED
tcp   10.0.0.214:51788            10.50.5.252:22             ESTABLISHED
tcp   10.0.0.254:4949             10.0.0.150:2829            ESTABLISHED
udp   10.0.0.150:514              10.0.0.254:514             UNREPLIED
udp   10.50.5.252:1031            194.77.253.129:53          ASSURED
router:~ #
```

Figure 1: Entering -L -n tells Netstat-nat to display the connections that do not pass through the NAT gateway.

Establishing the Facts

Entering the following finds out whether a specific computer in the masked network is currently establishing a connection via the NAT gateway:

```
netstat-nat -s name
```

The *name* variable can be any resolvable host name or an IP address. This also works in the other direction – setting the *-d name* parameter displays computers that are NAT connection targets.

But what about connections that do not pass through the NAT gateway, such as my own SSH connection to the gateway?

Typing the following displays output like that in Figure 1:

```
netstat-nat -L -n
```

The *-n* parameter prevents host and port name resolution. Although this is not implemented right now, it would be useful to tell the tool to resolve host and port names. Appending *|cat -b* to the command for longer output makes sense because it adds a line number to each line, making it easier to guess how many pages just scrolled past without actually reading them. ■

SYSADMIN

INFO

[1] Netstat-nat: *http://tweegy.demon.nl/projects/netstat-nat/*

THE AUTHOR

Charly Kühnast is a Unix System Manager at the data center in Moers, near Germany's famous River Rhine. His tasks include ensuring firewall security and availability and taking care of the DMZ (demilitarized zone).