Brücke mit Tunnel

Auch wer sich hinter einer Firewall oder einem NAT-Gateway befindet, kann seinem Wunsch nach öffentlichen Serverdiensten Geltung verschaffen – dem Active Port Forwarder sein Dank. Charly Kühnast

Inhalt

Die meisten Partnerschaften werden offenbar am Arbeitsplatz geknüpft. Über deren Langzeitstabilität will ich nicht spekulieren, man soll Privates und Berufliches ordentlich trennen. Bei meinen Webservern halte ich das so: Der private steht bei mir zu Hause. Da ich dort keine Standleitung besitze, wechselt meine IP bei jedem Dialup.

Lösen lässt sich dieses Problem normalerweise mit einem DNS-Dienst wie Dyndns, der einen festen Namen auf die jeweils aktuelle IP-Adresse mappt. Ich stecke aber trotzdem in der Klemme, weil meine Maschine hinter einem NAT-Gateway – dem DSL-Router – steht. Der Router hat die vom Provider zugeteilte IP für sich gepachtet, während mein Heimnetz mit privaten Adressen (192 .168.*X*.*X*) werkelt.

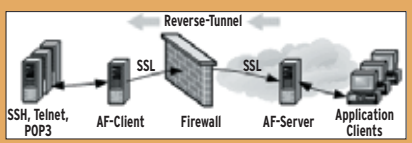

Abbildung 1: Der AF-Server bildet den Brückenkopf im Internet, er leitet die Pakete zum AF-Client.

Der Router verhindert wirkungsvoll den Zugriff aus dem Internet auf lokale Rechner – eigentlich ein wertvolles Sicherheitsfeature, das mich hier aber nur mäßig beglückt. Aktuelle Linux-Firewalls ließen sich zwar per DNAT (Destination-NAT) dazu überreden, einen externen Port auf einen internen Rechner umzuleiten, viele Appliances bieten dieses Feature aber schlicht nicht an.

Von hinten

Da NAT-Router von innen nach außen sehr durchlässig sind (Ausnahme: Der Router besitzt eigens definierte Paketfilterregeln), kann ich munter Verbindungen ins Internet aufbauen. Und mit der richtigen Software benutze ich die Verbindungen dazu, Daten rückwärts ins heimische Netz zu tunneln (siehe Abbildung 1). Genau das leistet Active Port Forwarder [1], es trägt aktuell die Versionsnummer 0.5.3.

Beim Kompilieren entstehen eine Client- und eine Server-Komponente. Außerdem brauche ich einen Brückenkopf im Internet: einen Server, auf dem ich Shell-Zugang habe. Den gibt mir ein Bekannter (kein Arbeitskollege, siehe oben), der einen Root-Server gemietet hat. Auf diesem Brückenkopf kompiliere ich den Port Forwarder. Von den entstehenden Files benötige ich nur die Server-Komponente. Sie startet durch »./afserver« mit folgenden Default-Einstellungen:

- Der Server lauscht auf Port 50127.
- Die Client-Komponente – die baue ich als Nächstes – darf sich an Port 50126 verbinden.
- Der Server akzeptiert maximal fünf gleichzeitige TCP-Verbindungen.
- Das Logging ist abgeschaltet.

Wem diese Voreinstellungen nicht zusagen, der baut eine kleine Konfigurationsdatei (ein Beispiel liegt dem Paket bei) und ruft den Server mit »./afserver -f *Konfigurationsdatei*« auf.

Beim Client geht's ähnlich einfach. Ich kompiliere auf meinem Wohnzimmer-Webserver das Paket und benutze nur die Client-Komponente: »./afclient -n *Servername* -p 80«. Als Servername eignen sich der voll qualifizierte Name oder die IP-Adresse des Brückenkopfs. Das war's! Wenn ich jetzt den Server auf Port 50127 ansurfe, antwortet mir meine Wohnzimmer-Kiste. Die Verbindung zwischen AF-Server und -Client ist übrigens SSL-verschlüsselt. *(jk)* ■

Infos

[1] Active Port Forwarder: [http://www.gray-world.net/pr_af.shtml]

Der Autor

Charly Kühnast administriert Unix-Betriebssysteme im Rechenzentrum Niederrhein in Moers. Zu seinen Aufgaben gehören die Sicherheit und Verfügbarkeit der Firewalls und der DMZ (demilitarisierte Zone). In seiner Freizeit lernt er Japanisch, um endlich die Bedienungsanleitung seiner Mikrowelle lesen zu können.