The Sysadmin's Daily Grind: Rinetd

Man in the Middle

No matter if you're talking about the protagonist in a mediocre spy movie or a server, you will probably prefer to use a man in the middle, rather than look danger in the eye. BY CHARLY KÜHNAST

The man in the middle of a network will normally be a proxy. Proxies elevate traffic to the application level where they verify, cache and manipulate it. If you do not need this extended functionality, you might consider using a simple redirector, such as rinetd [1]: Rinetd accepts connections on a specified port and relays them to a pre-defined port on another host. Since there is no need to elevate the traffic to application level, this method is quick and easy on your resources.

Rinetd is available for Linux and Windows; the Linux version is a tarball that weighs in at a mere 35 Kbytes, and can be easily extracted, using the typical »make; make install« procedure. The redirection rules are stored in the »/etc/rinetd.conf« file, which is not installed automatically – you will have to take care of that yourself.

To provide a simple example, let's construct a redirector for a web server. We want to redirect the server with the IP 10.0.0.1 to the server at IP 10.0.0.2. The web server is listening on port 80 on both systems. The line in »rinetd.conf« will read:

```
10.0.0.1 80 10.0.0.2 80
```

Of course, you can use names instead of IP addresses. If the server at 10.0.0.1 has more than one IP address, and I want the redirection to apply to port 80 for any of the other IPs, there is no need to add a redirection rule for each IP. Instead you can simply type

```
0.0.0.0 80 10.0.0.2 80
0.0.0.1
```

This redirects port 80 for every IP address the server owns to 10.0.0.2.

Allow and Deny Rules

To prevent every connection being redirected, I can use the »allow« and »deny« rules to specify the customers allowed or not allowed to use the redirector. The rules preceding the first redirection rule in »rinetd.conf« are global, that is they apply to all the redirections defined in the file. For example:

```
allow 192.168.0.*

10.0.0.1 80 10.0.0.2 80
10.0.0.1 22 10.0.0.2 22
1.1.1.1 3128 1.1.1.2 8080
```

This configuration allows redirection of connections that originate in the 192.168.0.* network. However, if you want to apply this restriction to the first rule only, you must insert the »allow« rule after the redirection rule:

```
10.0.0.1 80 10.0.0.2 80
allow 192.168.0.*
10.0.0.1 22 10.0.0.2 22
1.1.1.1 3128 1.1.1.2 8080
```

In this case the last two rules apply to connections from everywhere, but the first rule rejects any connection attempts that do not originate in the 192.168.0.* network.

If you want to know what »rinetd« is up to, you will have to convince the program to write to a logfile, by adding another entry to »rinetd.conf«. The entry will be as follows:

```
logfile /var/log/rinetd.log
```

The additional »logcommon« line makes »rinetd« write its logs in Common Logfile Format (CLF) that is also used by Apache and Squid (if so configured). This has the added advantage that many programs designed for evaluating logfiles can be used here, since practically any reporting tools can handle CLF files.

While the first report is being generated, you could always watch a mediocre spy film. Who knows – you might learn something. ■

INFO

[1] Rinetd home page: *http://www.boutell.com/rinetd*

THE AUTHOR

Charly Kühnast is a Unix System Manager at a public datacenter in Moers, near Germany's famous River Rhine. His tasks include ensuring firewall security and availability and taking care of the DMZ (demilitarized zone). Although Charly started out on IBM mainframes, he has been working predominantly with Linux since 1995.