The sys admin's daily grind: ifdata

Precision Interfacing

Script-friendly ifdata from the Moreutils package delivers absolutely precise network interface status information, with no need to extract individual values. *By Charly Kühnast*

When I need to know whether a particular interface exists – and, if so, what IP address and MTU it has – my script is inevitably bulky. I then bombard the results of `ip addr show eth0`, `ifconfig`, or `iwconfig` with `grep`, `cut`, and regular expressions until the requested information is chiseled out. In contrast, ifdata from the Moreutils package [1] solves my task elegantly. I can use parameters to control the ifdata output so that only the desired result appears, which is ideal for ongoing processing in a script. If I want to know whether a specific network interface exists, I can use:

```
ifdata -e eth0
```

The command produces no output, but the return value reveals whether the interface exists. In a script, this is very convenient if I want to branch:

```
if $(ifdata -e eth0); then
    echo "Interface found";
    [...]
else
    echo "Interface not found";
    [...]
fi
```

For human-friendly output, the `-pe` parameter tells ifdata to output *yes* or *no*. With a mini-loop, I can quickly conjure up an overview of the number of active interfaces:

```
$ for i in `seq 0 3`;do ifdata -pe eth ➥
  $i;done
yes
yes
 yes
 no
```

In this example, the test machine has four interfaces but is using only three.

Groundwork

Ifdata outputs a space-separated list of the most frequently needed data – IP address, netmask, broadcast address, and MTU – if you specify the `-p` parameter:

```
$ ifdata -p eth0
  10.0.0.106 ➥
  255.255.255.0 ➥
  10.0.0.255 1500
```

I could also get this information by specifying `-pa`, `-pn`, `-pb`, and `-pm` individually. If I want to know whether the interface is in promiscuous mode, I can type

```
$ ifdata -pf eth0
```

to compose a list of flags (Figure 1), although you can't retrieve the items in the list separately – which is fine by me. After all, who can remember all these parameters? You can also find out more about the network traffic and error conditions. For example, I can enter

```
$ ifdata -soe eth0
```

to read the error counter – and then turn to other tasks with the reassurance that the counter reads `0`. ■■■

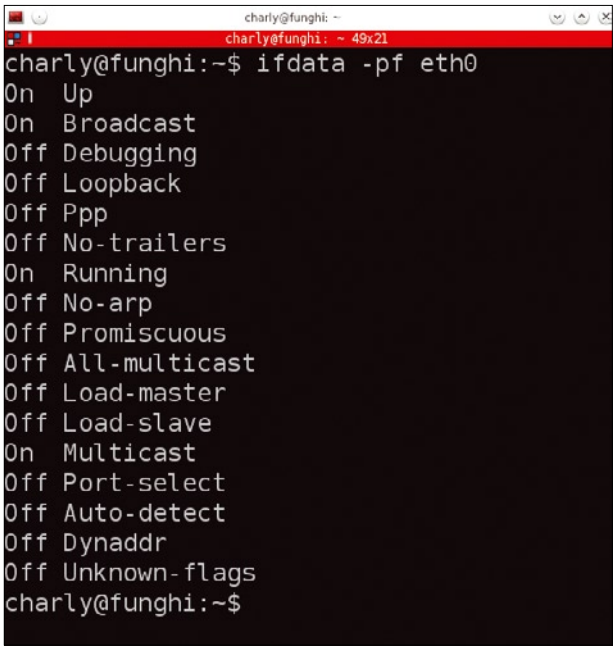

Figure 1: Ifdata looking for flags set by the network interface.

INFO

[1] Moreutils: *http://joeyh.name/code/moreutils/*

CHARLY KÜHNAST

Charly Kühnast is a Unix operating system administrator at the Data Center in Moers, Germany. His tasks include firewall and DMZ security and availability. He divides his leisure time into hot, wet, and eastern sectors, where he enjoys cooking, freshwater aquariums, and learning Japanese, respectively.