Tools, old and new

Tom Ryder
tom@sanctum.geek.nz
https://sanctum.geek.nz/

Durable tools

- Text as an interface doesn’t really age
- Command-line tools last a long time
- Online command examples become holy writ
  - Wikis
  - Q&A sites (Stack Overflow)
  - Other people’s shell scripts

Deprecating things is hard

- Devs post about it on the website
- Devs commit about it to the repository
- Devs write about it on the wiki
- Devs mention it in the manual page
- Devs mention it in the `--help` output
- …and yet…

Did *you* know?

All of these tools are deprecated:

- `ifconfig(8)`
- `route(8)`
- `netstat(8)`
- `iptables(8)`
- `egrep(1)`
- `fgrep(1)`

Networking

- `ifconfig(8)`, `route(8)`, and `netstat(8)` are all deprecated
  - On Debian GNU/Linux, they can be installed with the `net-tools` package
  - Don’t do that out of mere stubbornness, though...
- Use `ip(8)` instead, from the iproute2 suite

Networking—Links (interfaces)

```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enX0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode
DEFAULT group default qlen 1000
    link/ether be:ca:ef:eb:56:06 brd ff:ff:ff:ff:ff:ff
…
```

Networking—Brief output

```
$ ip -brief link
lo      UNKNOWN   00:00:00:00:00:00  <LOOPBACK,UP,LOWER_UP>
enX0    UP        be:ca:ef:eb:56:06  <BROADCAST,MULTICAST,UP,LOWER_UP>
enX1    UP        ae:34:77:ae:bd:25  <BROADCAST,MULTICAST,UP,LOWER_UP>
enX2    UP        f2:ef:45:9c:ea:4a  <BROADCAST,MULTICAST,UP,LOWER_UP>
```

Networking—Color output

```
$ ip -brief -color link
lo       UNKNOWN    00:00:00:00:00:00  <LOOPBACK,UP,LOWER_UP>
enX0     UP         be:ca:ef:eb:56:06  <BROADCAST,MULTICAST,UP,LOWER_UP>
enX1     UP         ae:34:77:ae:bd:25  <BROADCAST,MULTICAST,UP,LOWER_UP>
enX2     UP         f2:ef:45:9c:ea:4a  <BROADCAST,MULTICAST,UP,LOWER_UP>
```

Networking—JSON output (!)

```
$ ip -json link
[{"ifindex":1,"ifname":"lo","flags":
["LOOPBACK","UP","LOWER_UP"],"mtu":65536,"qdisc":"noqueue","operstate":"UN
KNOWN","linkmode":"DEFAULT","group":"default","txqlen":1000,"link_type":"l
oopback","address":"00:00:00:00:00:00","broadcast":"00:00:00:00:00:00"},
{"ifindex":2,"ifname":"enX0","flags":
["BROADCAST","MULTICAST","UP","LOWER_UP"],"mtu":1500,"qdisc":"mq","opersta
te":"UP","linkmode":"DEFAULT","group":"default","txqlen":1000,"link_type":
"ether","address":"be:ca:ef:eb:56:06","broadcast":"ff:ff:ff:ff:ff:ff","ifa
lias":"public"},{"ifindex":3,"ifname":"enX1","flags":
["BROADCAST","MULTICAST","UP","LOWER_UP"],"mtu":1500,"qdisc":"mq","opersta
te":"UP","linkmode":"DEFAULT","group":"default","txqlen":1000,"link_type":
"ether","address":"ae:34:77:ae:bd:25","broadcast":"ff:ff:ff:ff:ff:ff","ifa
lias":"backnet"},
…
```

Networking—Addresses

```
$ ip -brief address
lo               UNKNOWN        127.0.0.1/8 ::1/128
enp43s0          DOWN
wlp0s20f3        UP             192.168.1.18/24 fe80::66a8:…
wg0              UNKNOWN        192.168.21.130/32 2404:1800:700:3501::…

$ ip address add 192.168.1.19/24 dev wlp0s20f3

$ ip address delete 192.168.1.19/24 dev wlp0s20f3
```

Networking—Routes

```
$ ip route
default via 192.168.1.1 dev wlp0s20f3 proto dhcp src 192.168.1.18 metric
600
192.168.1.0/24 dev wlp0s20f3 proto kernel scope link src 192.168.1.18
metric 600
203.114.151.8/29 dev wg0 proto static scope link metric 50

$ ip route add 192.168.2.0 via 192.168.1.254 dev wlp0s20f3

$ ip route delete default via 192.168.1.1 dev wlp0s20f3
```

Networking—Shorter

```
$ ip a
$ ip l
$ ip r
$ ip -br -c r
$ ip -j -p a
```

Networking—Connections

- Instead of `netstat(8)` to list connections, use `ss(8)`.

```
$ ss -lt
State      Recv-Q     Send-Q         Local Address:Port         Peer Address:Port
LISTEN     0          4096           203.114.151.10:9051             0.0.0.0:*
LISTEN     0          4096           203.114.151.10:9050             0.0.0.0:*
LISTEN     0          511                127.0.0.1:redis             0.0.0.0:*
LISTEN     0          256            203.114.151.10:domain           0.0.0.0:*
LISTEN     0          511            203.114.151.10:https            0.0.0.0:*
LISTEN     0          4096               127.0.0.3:http              0.0.0.0:*
…
```

Networking—Statistics

- Instead of `netstat(8)` to view interface statistics, use `ip(8)`:

```
$ ip -stats link
…
2: enp9s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP mode DEFAULT group default qlen 1000
    link/ether 60:eb:69:02:34:57 brd ff:ff:ff:ff:ff:ff
    RX:  bytes packets errors dropped  missed   mcast
    2963400536 3732738      0  167217       0  168071
    TX:  bytes packets errors dropped carrier collsns
    2232530143 2850561      0       0       0       0
```

Networking—Firewalls

- `iptables(8)` is deprecated!
  - It still works, but through a translation layer.
- Use `nft(8)` instead!
  - (Actually, don’t; use firewalld—zone-based firewall.)

Bag of tricks

- util-linux bills itself as “a random collection of Linux utilities”
- Some of it you likely already use, like `fsck(8)` or `dmesg(1)`
- Lots of other useful tools; check out the ones starting with `ls-` particularly, like `lscpu(1)`, `lsmem(1)`, `lsns(8)`…
- A nice complement to the GNU coreutils for those on GNU/Linux specifically

util-linux—Mounts—1/3

Don’t use `mount(8)` merely to list mounted filesystems for human reading; use `findmnt(8)`!

- Shows mountpoints in a hierarchy
- Filter mounts by type, e.g. `-t ext4`
- Use `--real` to get only “real” filesystems, excluding e.g. `/proc`, `/run`, `/sys`…
- JSON output mode!

util-linux—Mounts—2/3

```
$ findmnt
TARGET                                SOURCE     FSTYPE   OPTIONS
/                                     /dev/mapper/darmok--vg-root
│                                                ext4     rw,relatime,errors=remount-ro
├─/sys                                sysfs      sysfs    rw,nosuid,nodev,noexec,relatime
│ ├─/sys/kernel/security              securityfs security rw,nosuid,nodev,noexec,relatime
│ ├─/sys/fs/cgroup                    cgroup2    cgroup2  rw,nosuid,nodev,noexec,…
│ ├─/sys/fs/pstore                    pstore     pstore   rw,nosuid,nodev,noexec,relatime
│ ├─/sys/fs/bpf                       bpf        bpf      rw,nosuid,nodev,noexec,…
│ ├─/sys/kernel/debug                 none       debugfs  rw,nosuid,nodev,noexec,relatime
│ ├─/sys/kernel/tracing               tracefs    tracefs  rw,nosuid,nodev,noexec,relatime
│ ├─/sys/kernel/config                configfs   configfs rw,nosuid,nodev,noexec,relatime
│ └─/sys/fs/fuse/connections          fusectl    fusectl  rw,nosuid,nodev,noexec,relatime
├─/proc                               proc       proc     rw,nosuid,nodev,noexec,relatime
│ └─/proc/sys/fs/binfmt_misc          systemd-1  autofs   rw,relatime,fd=38,pgrp=1,timeout=0,…
│   └─/proc/sys/fs/binfmt_misc        binfmt_misc
│                                                binfmt_m rw,nosuid,nodev,noexec,relatime
…
```

util-linux—Mounts—3/3

```
$ findmnt --real
TARGET                   SOURCE                      FSTYPE      OPTIONS
/                        /dev/mapper/darmok--vg-root ext4        rw,relatime,errors=remount-ro
├─/var                   /dev/mapper/darmok--vg-var  ext4        rw,relatime
│ ├─/var/local/syncthing shaka:/syncthing            nfs4        rw,relatime,vers=4.2,…
│ ├─/var/local/download  shaka:/download             nfs4        rw,relatime,vers=4.2,…
│ ├─/var/local/systems   shaka:/systems              nfs4        rw,relatime,vers=4.2,…
│ └─/var/local/media     shaka:/media                nfs4        rw,relatime,vers=4.2,…
├─/home                  /dev/mapper/darmok--vg-home ext4        rw,relatime
└─/boot                  /dev/sda1                   ext2        rw,relatime
```

util-linux—Block devices—1/4

- If the block device hasn’t been mounted yet, you might try and find it in `/dev`
  - But even `/dev` block devices are complicated nowadays…
  - LVM, RAID, loopback devices…

util-linux—Block devices—2/4

`blkid(8)` is a nicer way; I use this to get UUIDs:

```
$ sudo blkid
/dev/mapper/darmok--vg-root: UUID="48a71dca-b091-…" BLOCK_SIZE="4096" TYPE="ext4"
/dev/mapper/darmok--vg-home: UUID="1ed41726-3e22-…" BLOCK_SIZE="4096" TYPE="ext4"
/dev/mapper/darmok--vg-var: UUID="3c256da7-1572-…" BLOCK_SIZE="4096" TYPE="ext4"
/dev/mapper/sda5_crypt: UUID="6eGMXC-e00Q-…" TYPE="LVM2_member"
/dev/sda5: UUID="4ad720a9-044f-…" TYPE="crypto_LUKS" PARTUUID="b4cbeaef-05"
/dev/sda1: UUID="1a5dd885-786b-…" BLOCK_SIZE="1024" TYPE="ext2" PARTUUID="b4cbeaef-01"
/dev/mapper/darmok--vg-swap: UUID="06e1743b-1689-…" TYPE="swap"
…
```

util-linux—Block devices—3/4

`lsblk(8)` is even nicer:

- Shows partitions/volumes in a hierarchy
- Great for cryptdisks and/or LVM
- Include filesystem data with `--fs` (e.g. usage)
- JSON output again, too (boy, they do love that…)

util-linux—Block devices—4/4

```
$ lsblk
NAME                      MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINTS
sda                         8:0    0 167.7G  0 disk
├─sda1                      8:1    0   487M  0 part  /boot
├─sda2                      8:2    0     1K  0 part
└─sda5                      8:5    0 167.2G  0 part
  └─sda5_crypt            254:0    0 167.2G  0 crypt
    ├─darmok--vg-root     254:1    0    25G  0 lvm   /
    ├─darmok--vg-var      254:2    0    10G  0 lvm   /var
    ├─darmok--vg-swap     254:3    0     8G  0 lvm   [SWAP]
    └─darmok--vg-home     254:4    0   120G  0 lvm   /home
sr0                        11:0    1  1024M  0 rom
```

util-linux—Open files—1/2

- Instead of `lsof(1)`, try `lsfd(1)`
  - Supports kernel features like namespaces
  - Different output format and calling conventions
- `lsof(1)` isn't deprecated, this is just a newer alternative
- Definitely check out the “FILTER EXAMPLES” in the man page

util-linux—Open files—2/2

```
$ lsfd -p 397956
COMMAND    PID USER  ASSOC  XMODE TYPE SOURCE MNTID      INODE NAME
bash    397956 tejr    exe ------  REG   dm-1     0     266119 /usr/bin/bash
bash    397956 tejr    cwd ------  DIR   dm-4     0     131073 /home/tejr
bash    397956 tejr    rtd ------  DIR   dm-1     0          2 /
bash    397956 tejr cgroup ------  REG   nsfs     0 4026531835 cgroup:[4026531835]
bash    397956 tejr    ipc ------  REG   nsfs     0 4026531839 ipc:[4026531839]
bash    397956 tejr    mnt ------  REG   nsfs     0 4026531841 mnt:[4026531841]
bash    397956 tejr    net ------  REG   nsfs     0 4026531840 net:[4026531840]
bash    397956 tejr    pid ------  REG   nsfs     0 4026531836 pid:[4026531836]
bash    397956 tejr  pid4c ------  REG   nsfs     0 4026531836 pid:[4026531836]
bash    397956 tejr   time ------  REG   nsfs     0 4026531834 time:[4026531834]
bash    397956 tejr time4c ------  REG   nsfs     0 4026531834 time:[4026531834]
bash    397956 tejr   user ------  REG   nsfs     0 4026531837 user:[4026531837]
bash    397956 tejr    uts ------  REG   nsfs     0 4026531838 uts:[4026531838]
…
```

`grep(1)` alternatives—1/2

- These are good for interactive use. For scripts, just stick to `grep(1)`.
- Niceties like ignoring `.git` directories by default
- I like ack, because I'm a Perl guy
- I see a lot of rg (ripgrep) and ag (The Silver Searcher) too
- GNU grep is a lot faster than you might think…
  - …a lot of effort went into that!

grep(1) alternatives—2/2

```
~/documents/Writing/Poetry$ ack -i time
biopsy/biopsy.md
16:A few times…an increasing few,
27:He says.  (Not twenty years, this time.)

gwen/gwen.md
27:Time would soon take care of that:

nothing/nothing.md
9:"We're not worried," he says, "she'll speak in time."

cold-comfort/early.md
26:And it takes some time, but like her I grow

cold-comfort/cold-comfort.md
32:And it takes some time, but like her I grow
```

Coda for `grep(1)`

- Don’t use `egrep(1)` or `fgrep(1)` anymore
  - They are deprecated
  - Eventually they will go away
- Your distribution may be covering for you
  - Debian GNU/Linux does this
- Use `grep(1)`’s `-E` and `-F` flags instead
  - **`-E`**—Search with Extended Regular Expressions
  - **`-F`**—Search for fixed strings, not expressions

Deprecation or preference?

- In some cases, the choice veers a lot closer to preference than to a deprecation:
  - GNU Screen vs tmux?
  - GNU Bash vs Zsh?
  - Vim vs Em—…on second thought, let’s not go there.
- If your older tool still works and is maintained, you needn’t leap to the new and shiny.
  - But it really is time to let `ifconfig(8)` go...

Questions? Your own favourite new tools?

- iproute2
- util-linux
- ack, ag, ripgrep

Email: tom@sanctum.geek.nz
Website: https://sanctum.geek.nz/
Fediverse: @tejr@mastodon.sdf.org