I/O Agility at the Command Line

- Some text-shell tools that come standard on many of today's operating systems extend the I/O redirection concept in interesting ways
- What follows is a brief description of some of these features — while they won't necessarily be of use to everyone in all situations, knowing that you can do these things might come in handy one day
- Needless to say, these illustrate yet again of mechanism over policy: the *ability* to communicate is decoupled from the *path* taken by that communication

ssh Tunnels

- Most of you know *ssh* as a convenient, secure way to run a remote text-based shell…but it can do way more!
- In particular, *ssh* can be used to create *tunnels* — that is, alternative network paths (like redirecting traffic)
- The secret parameters are *-L* and *-R*:
    - ◇ *-L* (for local) takes a connection that is available to the (*ssh*'d) machine and makes it available as if it were on *your* machine
    - ◇ *-R* (for remote) takes a connection that is available on your machine and makes it available to the *ssh*'d machine
    - ◇ As always, full details are in *man* pages and assorted how-to's on the net

Detachable Terminals with *screen*

- Ever wished that, for whatever reason, you could maintain a text shell session on a computer without keeping a terminal application running or even staying logged in to its GUI shell? Yes, you can — with *screen*
- *screen* "wraps" a text shell session around a detachable container: when you run *screen*, you get a standard *bash* (or other shell) session
- Pressing *ctrl-A D* (in sequence) *d*etaches the terminal from that session *without ending it*; logout and walk away as needed, and reconnect via *screen -r* anytime later

Watching Files with *tail*

- Slightly more mundane than *screen* or *ssh*, but sometimes useful in concert with them, is *tail*
- By default, *tail* displays displays the last *n* lines (the "tail end") of a file — useful for viewing the most recent entries in a log file, for example
- However, *tail -f* keeps *tail* running and "watches" the *tail*-ed file so that you can see it change on the fly
- *tail* running in *ssh* running in *screen* thus gives you a makeshift "remote file monitor"

Watching the Network

- With all of this routing and redirecting going on, the well-equipped operating system should also provide tools for monitoring these connections
- The "big brother" of these tools is *tcpdump* — with the right permissions, it allows you to see anything and everything about what's going on in with your network interfaces and devices
- For a friendlier face, you can use *Wireshark* (formerly known as *Ethereal*) — it's essentially *tcpdump* but with better presentation of filtering and other options

- The other network Swiss army knife is *netstat* (*net*work *stat*us) — depending on the command line parameters you provide, *netstat* can report on:
  - Open connections
  - Routes (i.e., where network messages go)
  - Statistics
- Other tools that may be of use when examining a machine's network setup include:
  - *ping* — The basic reachability tool, for checking if one machine can reach or "see" another on the network
  - *traceroute* (*tracert* in Windows) — Displays, when possible, the specific path taken by connections from one machine to another
  - *dig* (a successor to *nslookup*) — Queries the *domain name service* (DNS) for various items like host names, IP addresses, mail servers, etc.
  - *arp* — A lower-level tool for translating IP addresses into Ethernet ("MAC") addresses

How About GUI Agility?

Text shells don’t have all the fun — you can route your GUI around too (everything’s just bits after all), at the cost of more bandwidth

- Cross-platform tools include *vnc* (of which TightVNC is a popular implementation) and, of course, X-windows (routable either “in the open” or securely via *ssh -X*)
- Mac OS X has *Screen Sharing*, which is *vnc* compatible
- Windows has *Remote Desktop Connection (RDC)*, with free clients available on multiple platforms

Back to *The Art of Unix Programming…*

As a postscript, consider the “I/O agility” tools in the context of this additional rule from Eric Steven Raymond’s *The Art of Unix Programming*:

Rule of Diversity
Distrust all claims for one true way

Appropriate, yes? Don’t let anyone convince you otherwise — especially in matters of I/O…