System Administration

Networking

UNIX

Network Configuration

- **another BIIIG topic**
  - and one that is growing in importance as the world gets used to the internet
- **I will assume that you know how to *use* the internet**
  - lets me focus on the issue of how to configure Unix to get a machine on the internet...

The Four Layers

- Unix TCP/IP networking is structured into 4 basic layers
  - physical
    - *deals with media, topology, cards, etc.*
  - IP
    - *routing, addressing/lookup, network classes, etc.*
  - TCP
    - *services, daemons, etc.*
  - applications
    - *such as FTP, WWW, etc.*

Configuring N/W Devices

- **as always, performed via /dev entries**
  - loopback device: lo
    - *allows networking within a single machine*
      - allows the 'pretence' of being connected even if not
  - ethernet: eth0
  - others include:
    - *fddi: fi0; isdn: isdninfo, isdnctrl; token ring: tr0*
  - PPP dialin/out device: ppp0

/sbin/ifconfig

- the main tool for device interface configuration
  - both getting and setting
  - *"It is used at boot time to set up interfaces as necessary. After that, it is usually only needed when debugging or when system tuning is needed."*

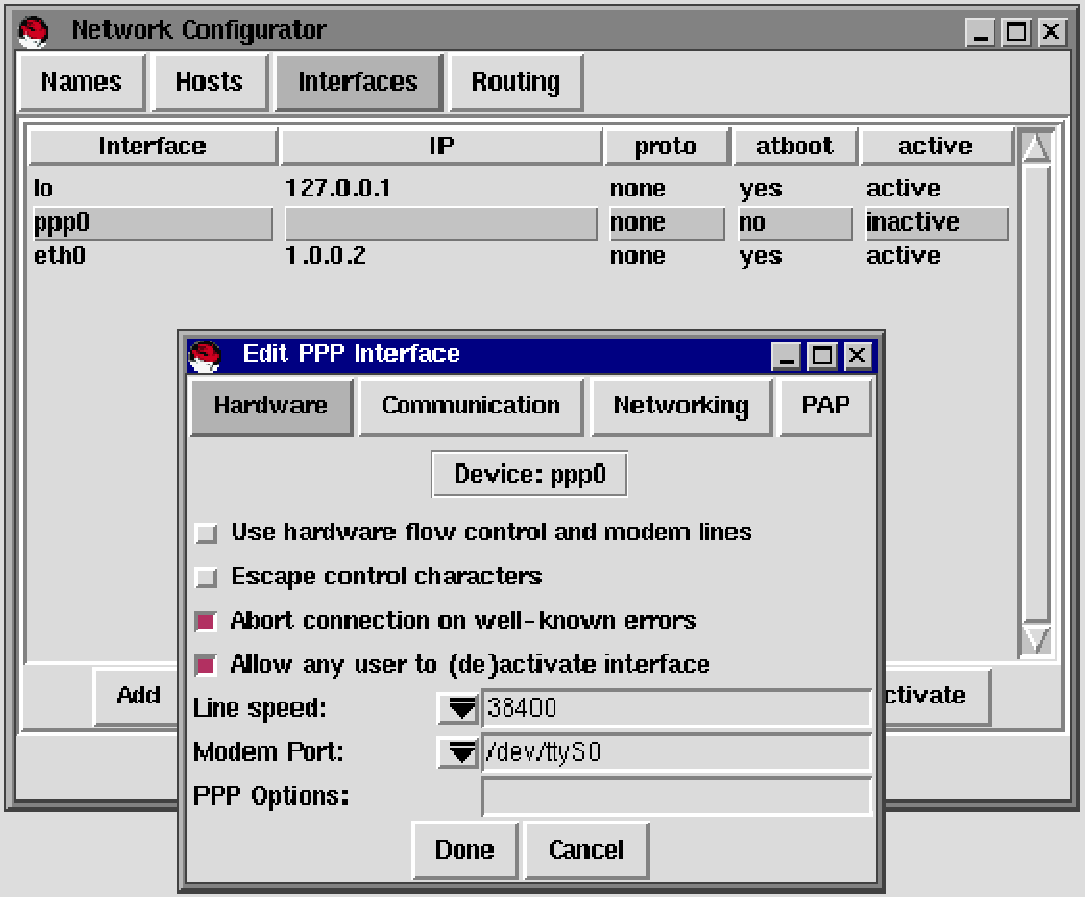
```
# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          RX packets:1 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0

eth0      Link encap:Ethernet  HWaddr 00:20:AF:09:80:C6
          inet addr:1.0.0.2  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2055 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1447 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0
          Interrupt:10 Base address:0x300
# ifconfig eth0 down
# ifconfig eth0 netmask 255.255.255.0 up
```

```
DEVICE=eth0
USERCTL=no
ONBOOT=yes
BOOTPROTO=none
BROADCAST=1.255.255.255
NETWORK=1.0.0.0
NETMASK=255.0.0.0
IPADDR=1.0.0.2
```

- for RedHat, initialization-time settings are stored in /etc/sysconfig/network-scripts/ifcfg-*interface*
- other network settings are in /etc/sysconfig/network
  - this is where the hostname is set, for instance
    - *it is copied from this file to /etc/HOSTNAME at bootup*

```
NETWORKING=yes
FORWARD_IPV4="yes"
HOSTNAME=redhat
GATEWAYDEV=""
GATEWAY=""
```

RedHat's netcfg



Network Addressing

- **a 4-byte field in all IP packets**
  - can be given in one of two forms
    - *Fully Qualified Domain Name (FQDN)*
      - human-readable format: redhat.skewst.home.net.au
    - *Dotted Decimal*
      - nearer the machine's understanding of an address: 1.0.0.1
- **network address classes**
  - can be used to build a hierarchy
    - *through different address classes*
      - *(see p.588, Frisch)*
  - also allows simple multicasting
- **subnetting**
  - uses a network *mask*
    - *says which bits of an address should be treated as network part and which identify the host portion*
  - makes it possible to isolate groups of addresses from other groups of addresses
    - *so that each floor in a building can be treated as a separate network, while actually sharing a class B network*

Name Translation

- translation of human-readable form of address to machine-readable version is done in one of two ways
  - /etc/hosts file
    - *suitable only for small, static, isolated networks*
    - (see p.600, Frisch)

```
127.0.0.1       localhost  loopback
1.0.0.1         aunty      aunty.skewst.home.net.au
1.0.0.2         redhat     samba   www     redhat.skewst.home.net.au
1.0.1.1         mac        mac.ppp.skewst.home.net.au
10.102.177.114  bob.dialup.dstc.edu.au
130.102.176.12  trapdoor.dstc.edu.au
```

  - Domain Name Service
    - next slide...
  - Linux allows a hybrid of the two schemes
    - *order keyword in /etc/host.conf*

```
order hosts,bind
```

  - the network services switch file /etc/nsswitch.conf determines where to get different kinds of data types; from what file or database

```
hosts: files nis dns
```

Domain Name System

- converts machine names to the IP numbers; maps from name to address and from address to name
  - redhat.skewst.home.net.au ←→ 1.0.0.2
    - *implemented by the named daemon*
    - *a subnet may have one primary and multiple secondary and caching-only nameds*
      - gives the network a degree of resilience
- configured by /etc/named.conf
  - refers to many other files and a cache directory structure
- whole books have been written about DNS configuration
  - it can be a difficult thing to do
    - *especially with regards to setting up a primary server*
  - *(see p.601, Frisch)*

Caching-Only Server Config.

```
# /etc/named.conf
options {
        directory "/var/named";
};
zone "." {
         type hint;
         file "root.hints";
};
zone "0.0.127.in-addr.arpa" {
         type master;
         file "pz/127.0.0";
};
```

```
# nslookup
Default Server:  localhost
Address:  127.0.0.1
> 127.0.0.1
Server: ns.skewst.home.net.au
Address:  1.0.0.2

Name:    localhost
Address:  127.0.0.1
```

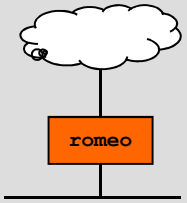
```
# /var/named/root.hints
# listing of known top-level DNS servers
  .                       6D IN NS        G.ROOT-SERVERS.NET.
 [snip]
  .                       6D IN NS        I.ROOT-SERVERS.NET.
  .                       6D IN NS        F.ROOT-SERVERS.NET.

 G.ROOT-SERVERS.NET.     5w6d16h IN A    192.112.36.4
 [snip]
 I.ROOT-SERVERS.NET.     5w6d16h IN A    192.36.148.17
 F.ROOT-SERVERS.NET.     5w6d16h IN A    192.5.5.241
```

```
# /var/named/pz/127.0.0
# (SOA == Start Of Authority)
 @               IN      SOA     ns.skewst.home.net.au. hostmaster.skewst.home.net.au. (
                                  1       ; Serial
                                  8H      ; Refresh
                                  2H      ; Retry
                                  1W      ; Expire
                                  1D)     ; Minimum TTL
                                NS      ns.skewst.home.net.au.
  1                             PTR     localhost.
```

Routing

- for a single, disconnected network routing is not necessary
- for a network that is connected to other networks (possibly via redundant paths) routing becomes important
- 2 routing techniques
  - static; uses explicit route commands

```
# route add -net 192.1.2.0 gw romeo
```



  - dynamic; handled by the routed/gated daemon
    - *invoked at boot time to manage the maintenance of the kernel's network routing tables*
    - *listens on the route service socket (defined in /etc/services)*

```
efs             520/tcp                         # for LucasFilm
route           520/udp         router routed   # 521/udp too
timed           525/udp         timeserver
```

    - *if the host is an internetworking router, it periodically supplies copies of its routing tables to any directly connected hosts and networks*
    - *routed is becoming obsolete in favor of gated*
      - gated implements a more suitable algorithm for big networks

More Routing

- **netstat**
  - gives information regarding the current routing table state

```
# netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
mac             *               255.255.255.255 UH     1500 0          0 eth0
192.56.76.0     *               255.255.255.0   U      1500 0          0 eth0
127.0.0.0       *               255.0.0.0       U      3584 0          0 lo
1.0.0.0         *               255.0.0.0       U      1500 0          0 eth0
default         aunty           0.0.0.0         UG     1500 0          0 eth0
```

  - netstat is actually a lot more useful that this...
- **traceroute**
  - netstat tells you where to go for the first hop, but what about all the other (non-local) hops?
    - "The Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route one's packets follow (or finding the miscreant gateway that's discarding your packets) can be difficult. Traceroute utilizes the IP protocol 'time to live' field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host."
    - *IP doesn't need to know this itself, but it can be useful information to have when debugging slowdowns, etc.*

```
# traceroute mac -i ppp0
traceroute to mac (1.0.1.1), 30 hops max, 40 byte packets
 1  mac (1.0.1.1)  78.254 ms  74.055 ms  78.459 ms
```

And Yet More Routing

- /bin/ping
  - always the first tool to turn to...

```
# ping aunty
PING aunty (1.0.0.1): 56 data bytes
64 bytes from 1.0.0.1: icmp_seq=0 ttl=128 time=1.2 ms
64 bytes from 1.0.0.1: icmp_seq=1 ttl=128 time=1.2 ms
64 bytes from 1.0.0.1: icmp_seq=2 ttl=128 time=1.3 ms

--- aunty ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1.2/1.2/1.3 ms
```

TCP Services

- there exists many TCP services
  - *(see p.593, Frisch)*
- standard ports are specified in the /etc/services file
  - non-standard ones are not listed here by default
    - *but should be added on a local-network basis*
- of special note is inetd
  - often called the *internet super daemon*
    - *it is a daemon that is responsible for initiating other daemons in response to incoming requests for service*
  - started at boot time, configured via /etc/inetd.conf
    - *(we saw this when we looked at security)*

```
ftp     stream   tcp   nowait  root  in.ftpd -l -a
telnet  stream   tcp   nowait  root  in.telnetd
gopher  stream   tcp   nowait  root  gn
```

    - *this mechanism reduces load on the system*
      - not appropriate for heavily used services such as rwhod, timed, etc.
        - the cost of continually starting and stopping these daemons would be too great…
    - *when /etc/inetd.conf is changed, inetd should be informed*

```
# kill -HUP `cat /var/run/inetd.pid`
```

Miscelleny

- /etc/protocols
  - lists those DARPA internet protocols that are available from the TCP/IP subsystem

```
ip      0       IP      # internet protocol, pseudo protocol number
icmp    1       ICMP    # internet control message protocol
igmp    2       IGMP    # internet group multicast protocol
ggp     3       GGP     # gateway-gateway protocol
tcp     6       TCP     # transmission control protocol
pup     12      PUP     # PARC universal packet protocol
udp     17      UDP     # user datagram protocol
idp     22      IDP     # WhatsThis?
raw     255     RAW     # RAW IP interface
```

- arp, rarp
  - kernel keeps tables of h/w address to IP address; these commands can maniplate those tables
    - *arp is Address Resolution Protocol*
      - translate IP address to physical h/w address
    - *rarp is Reverse ARP*
      - h/w address to IP address
- bootp
  - allows a diskless workstation to determine its network parameters from the network itself
    - *since it can't get them from config files!*

Other Services

- **multitudinous!**
  - telnet
  - ftp
  - mail (SMTP)
  - news (NNTP)
  - World-Wide Web (http)
  - gopher
  - tftp
  - smb
  - rlogin
  - nfs
  - nis
  - see /etc/services for more!

```
efs             520/tcp                         # for LucasFilm
```

- **we'll look at some services separately**
  - as the course progresses

NFS

- Network File System
  - allows systems to share filesystems across a network
    - *can actually share directories, not just whole filesystems*
  - uses a number of daemon processes
    - (see p.608, Frisch)
  - configuration is fairly simple
    - */etc/fstab should contain references to the remote directories being imported*

```
server:/usr/local/pub    /pub       nfs    rsize=8192,wsize=8192,timeo=14,bg,hard,intr
docserver:/usr/man       /usr/man   nfs    bg,soft
```

    - */etc/exports acts as an access control list for exported filesystems*

```
# sample /etc/exports file
/               master(rw) trusty(rw,no_root_squash)
/projects       proj*.local.domain(rw)
/usr            *.local.domain(ro) @trusted(rw)
/home/joe       pc001(rw,all_squash,anonuid=150,anongid=100)
/pub            (ro,insecure,all_squash)
/pub/private    (noaccess)
```

  - one useful tool: showmount
    - *queries the mount daemon on a remote host for information about the state of the NFS server on that machine*
      - can find out who the clients of a filesystem/server are

More NFS

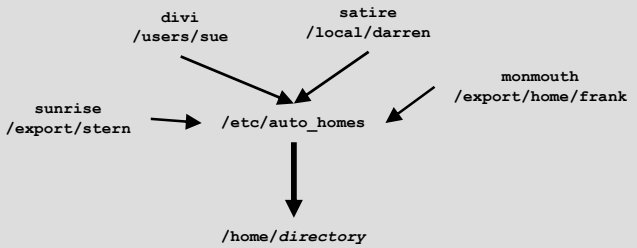
- a few things to watch out for
  - two possible failure modes
    - *hard mounts will cause a client to hang on an I/O request forever if a server goes down*
    - *a soft mount will simply return an error to the application*
      - *"it would seem that mounting filesystems soft would get around the process hanging problem. This is fine for systems mounted read-only. However, for a read-write filesystem, a pending request could be a write request, and so simply giving up could result in corrupted files on the remote filesystem. Therefore, read-write remote filesystems should always be mounted hard, and the intr option should be specified to allow users to make their own decisions about hung processes."*
  - if a server has a clock too far in advance of the client, file timestamps appear strange and can cause programs to break
    - *especially make/ld*
- nfsstat command (not Linux)

```
[snipped…]
Version 3: (21121642 calls)
null         getattr      setattr      lookup       access       readlink     read
276 0%       11644571 55% 93575 0%     6998374 33%  1266984 5%   281890 1%    477140 2%
write        create       mkdir        symlink      mknod        remove       rmdir
73580 0%     48867 0%     3908 0%      6399 0%      0 0%         20413 0%     983 0%
rename       link         readdir      readdir+     fsstat       fsinfo       pathconf
1563 0%      1008 0%      73216 0%     121994 0%    497 0%       260 0%       1066 0%
commit
5078 0%
```

The Automount Daemon

- traditional NFS mounting has a number of problems with network loading, reliability and inflexible administration
- automount is designed to overcome some of these:
  - booting time is significantly reduced because no mounting is done at boot time
  - mounts occur automatically and transparently when a user tries to access any files or directories under the designated mount point of a remote filesystem
  - network access and efficiency are improved by reducing the number of permanently active mount points
  - failed mount requests can be reduced by designating alternate servers as the source of a filesystem
    - *greatly improves reliability in large networks*
- when a directory is referenced it is mounted into a staging area and the automounter creates a symbolic link to the expected mount location

More Automount



- configured via maps
  - two types
    - *direct*
      - works according to the standard way of mounting NFS filesystems
    - *indirect*
      - *more frequently used*
      - *groups multiple source locations into an apparent regular arrangement*
        - *for example: gathering up all the varied user home locations and re-presenting them as coming from under /home*

```
# map /etc/auto_homes
stern           sunrise:/export/stern
sue             divi:/users/sue
frank           monmouth:/export/home/frank
darren          satire:/local/darren
```

      - *"when dealing with an indirect map, the automounter manages a directory of mount points instead of a set of directories that are mount points"*
- automounter looks for a special map called /etc/auto_master that contains a directory of direct and indirect maps

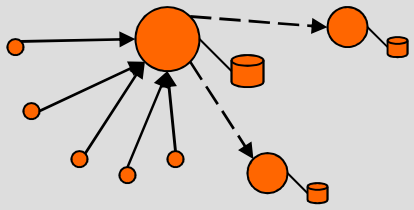
```
# /etc/auto_master
/home           /etc/auto_home
```

NIS

- Network Information Service
  - formerly knows as the "yellow pages service"
    - *but Sun were sued for trademark infringement by BT and had to change the name*
  - a distributed database service for common Unix files
    - *talk about NIS 'maps'*

```
#  ypwhich -x
Use "ethers"    for map "ethers.byname"
Use "aliases"   for map "mail.aliases"
Use "services"  for map "services.byname"
Use "protocols" for map "protocols.bynumber"
Use "hosts"     for map "hosts.byaddr"
Use "networks"  for map "networks.byaddr"
Use "group"     for map "group.byname"
Use "passwd"    for map "passwd.byname"
```

  - not very secure
    - *both NFS and NIS assume a closed, (fairly) trustworthy community*
    - *NIS+ was introduced to enhance security and also to allow better integration with DNS*
  - has a master/replicated slave structure to provide resilience



NIS Programs & Files

- **NIS involves a bunch of daemons, utilities and files**
  - /usr/sbin/portmap
    - *rpc service white pages server*
    - *rpcinfo*
  - /bin/domainname
    - *establishes a group of related machines which share the same* NIS *domain identifier*
      - *not* a DNS domain
  - ypserv
    - *server process*
  - ypbind
    - "As soon as ypbind is running, your system has become a NIS client"
  - ypwhich, ypset
    - *who is my server? point a client at a specified server*
  - yppush, ypxfr
    - *propagate a master's map to slave servers*
  - ypcat, ypmatch
    - *show, match data in an NIS map*
  - yppasswd
    - *rpc.yppasswd*
    - *note that NIS effectively nullifies any security benefits that shadow passwords might bring*
  - modified /etc/passwd and /etc/group
  - /etc/nsswitch.conf
    - *Name Services Switch configuration file*
  - plus a few other programs

NIS Configuration

- client
  - create /var/yp
  - add 'nis' to the name lookup order in /etc/host.conf
  - modify /etc/passwd to contain:

```
+::::::
```

    - *it is also possible to have a sophisticated passwd strategy:*

```
+miquels:::::::
+ed:::::::
+dth:::::::
+@sysadmins:::::::
-ftp
+:*::::::/etc/NoShell
```

    - "Shadow passwords over NIS are always a bad idea. You lost the security, which shadow gives you."
      - can be done but is not recommended
  - set the domainname; start portmap; start ypbind

More NIS Configuration

- **master server**
  - same as client initialization (no need to start ypbind if the server isn't also going to be a client)
  - ypinit -m
    - *creates the NIS maps from the standard /etc/* data files; leaes them in /var/yp/*domainname
    - *relies on /var/yp/Makefile to build and maintain the NIS maps*
  - start ypserv
- **slave server(s)**
  - ypinit -s *master*
  - start ypserv
  - typically, when a map is changed the master will propagate it to the slaves with yppush
    - *may want to ensure that the NIS maps are kept up-to-date, even if an update is missed because the slave was down at the time the update was done on the master*

```
20 *    * * *    /usr/lib/yp/ypxfr_1perhour
40 6    * * *    /usr/lib/yp/ypxfr_1perday
55 6,18 * * *    /usr/lib/yp/ypxfr_2perday
```

PPP

- **Point to Point Protocol**
  - an official IETF standard, RFC1661
  - a mechanism for creating and running IP and other network protocols over a serial link
    - *either a direct serial connection or a link between modems*
  - capable of encapsulating other protocols within it
    - *such as IP*
  - capable of using dynamically assigned addresses

PPP Configuration

- **a little convoluted to do by hand**
  - platform specific as well
- **3 major components to consider**
  - kernel support is required for PPP
    - *requires support for /dev/pppn devices*
  - automating modem control/login sequences with chat scripts
  - bringing a PPP link up and shutting it down

/usr/sbin/chat

- implements a versatile scripting language
  - designed to do expect/reply string matching and responses

```
# cat chat-ppp0
'ABORT' 'BUSY'
'ABORT' 'ERROR'
'ABORT' 'NO CARRIER'
'ABORT' 'NO DIALTONE'
'ABORT' 'Invalid Login'
'ABORT' 'Login incorrect'
'' 'ATM0'
'OK' 'ATDT12345678'
'CONNECT' ''
'TIMEOUT' '30'
'' '\n'
'e:' 'bob'
'd:' 'mypassword'
'service' 'ppp'
```

/usr/sbin/pppd

- daemon process implementing the PPP communication protocol
- when active, operates /dev/pppn

```
# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:1.0.0.2  P-t-P:1.0.1.1  Mask:255.0.0.0
          UP POINTOPOINT RUNNING  MTU:1500  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0
          Memory:abc038-abcc04
```

- for dialout PPP, use a command like:

```
# /usr/sbin/pppd connect "/usr/sbin/chat \"\" ATZ ATDT1234 CONNECT \"\" ogin: fred word: hellothere!" \
crtscts modem defaultroute 192.187.42.6:192.187.43.57 /dev/modem 38400 &
```

  - pppd should run suid root if anyone should be able to use it
- dialin PPP is pretty straightforward:
  - run a getty on the dialin line
    - *preferably uugetty, which understands locking*
  - create a user account for the dialin user

```
pppuser:x:999:999:Mr. PPP User:/home/pppuser:/usr/local/bin/pppuserloginscript
```

  - prepare a script to act as the user's login shell

```
#!/bin/sh
# /usr/local/bin/pppuserloginscript
exec /usr/sbin/pppd -detach silent modem crtscts 111.222.111.222:222.101.202.101.202
```

Automating PPP

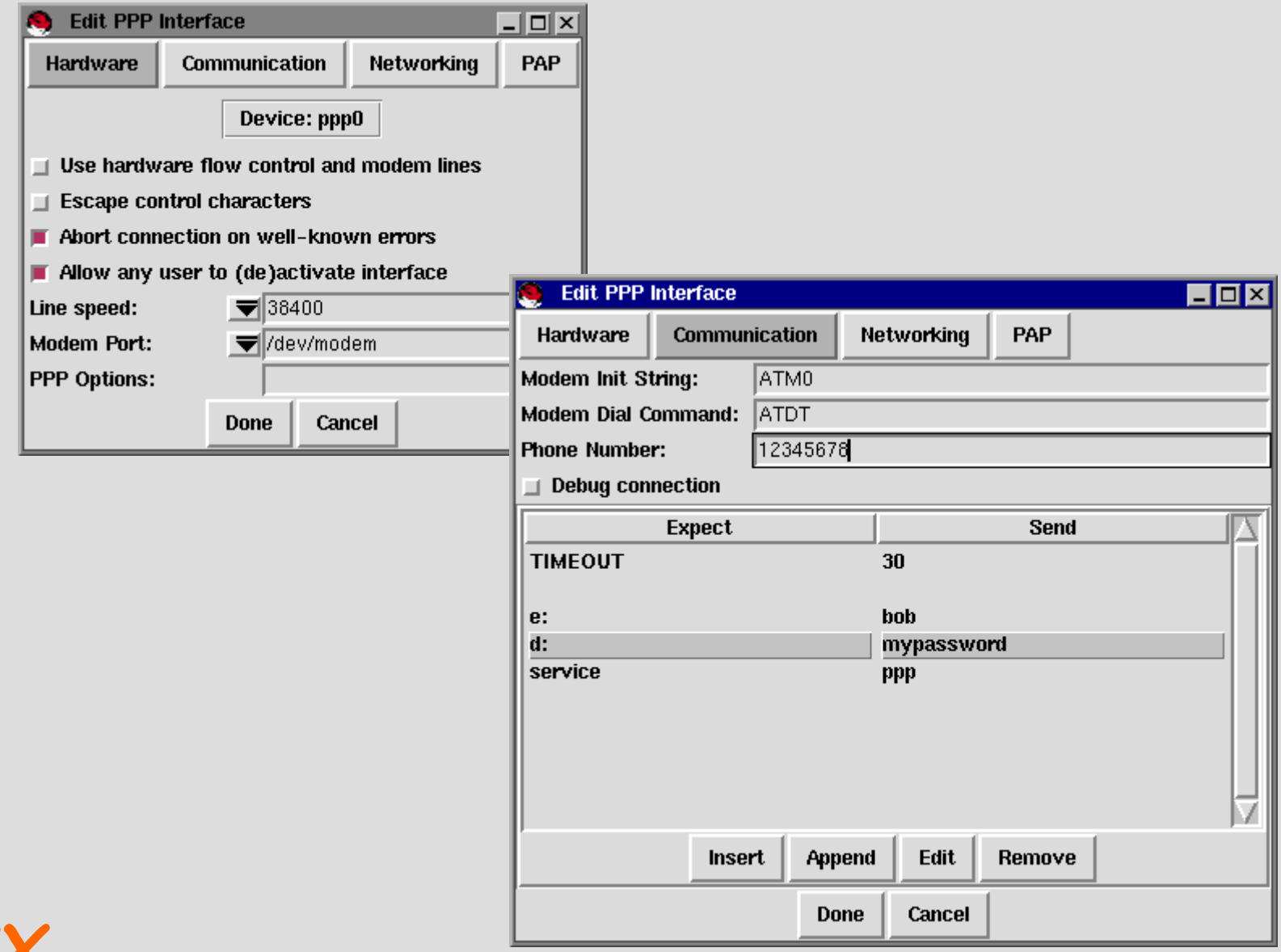
- RedHat provides a (fairly intricate) set of scripts to automate this process
  - so that all one needs to do is
    - *configure via a simple gui*
    - *either*
      - run from a simple gui X-ISP
      - via the command line

```
# ifup ppp0
```

        - fires off a series of scripts

```
# ls /etc/sysconfig/network-scripts/
chat-ppp0          ifdown@            ifup-aliases*      ifup-routes*
ifcfg-eth0         ifdown-post*       ifup-ipx*          ifup-sl*
ifcfg-lo*          ifdown-ppp*        ifup-plip*         network-functions
ifcfg-ppp0*        ifdown-sl*         ifup-post*
ifdhcpc-done*      ifup@              ifup-ppp*
```

Redhat Configuration



Cheops

- a simple X based network monitoring tool
  - useful for connectivity tests
    - *simple ICMP "ping" packets are used to initially search a network for hosts that are alive*
    - *Domain Name Transfers are used to list hosts in a domain*
    - *OS detection is done using invalid flags on TCP packets*
    - *port detection is done (somewhat) silently*
    - *mapping is done using UDP (or optionally ICMP) packets*
  - several commercial equivalents exist
    - *expensive!*
    - *e.g. IBM NetView*