

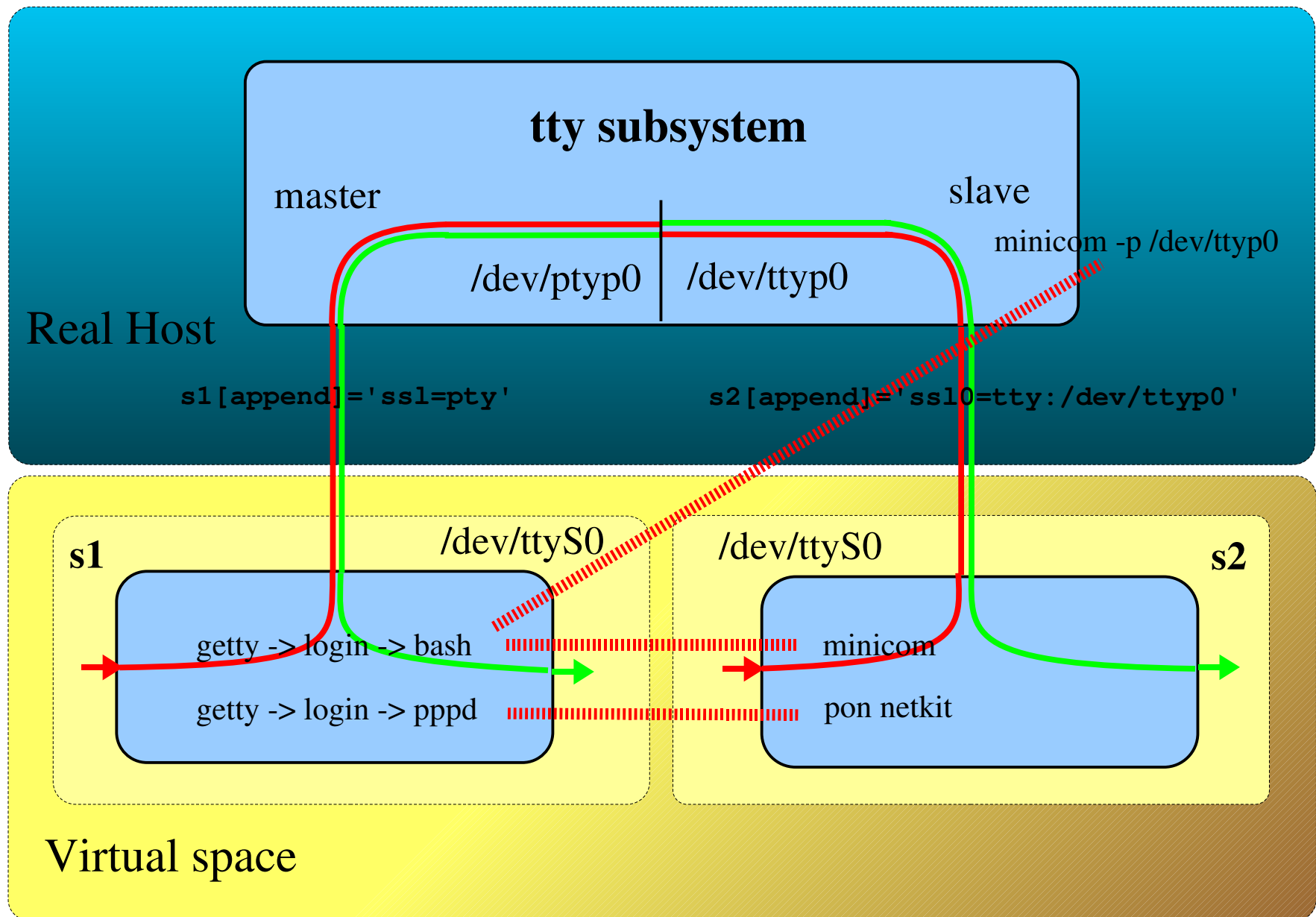
# Netkit4TIC lab

<b>Name</b>	serial
<b>Version</b>	1.0
<b>Author</b>	Sandro Doro
<b>E-mail</b>	sandro.doro@istruzione.it
<b>Web</b>	<a href="http://www.tic.fdns.net/tic/html/lab.html">http://www.tic.fdns.net/tic/html/lab.html</a>
<b>Description</b>	configuring serial lines, connect them together with minicom or with pppd

# copyright notice

- All the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright.
- This material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide.
- This material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes.
- Information contained in this material cannot be used within network design projects or other products of any kind.
- Any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement.
- The authors assume no responsibility about this material and provide this material “as is”, with no implicit or explicit warranty about the correctness and completeness of its contents, which may be subject to changes.
- This copyright notice must always be redistributed together with the material, or its portions.

# step 1 – serial topology



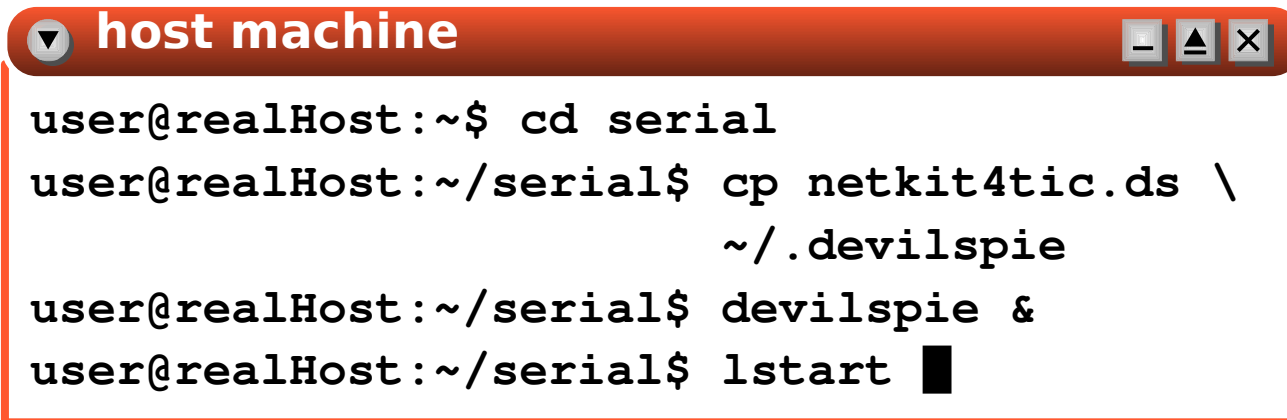
## step 2 – prerequisite

```
▼ realHost
```

```
realHost:~# cat /proc/tty/drivers
```

/dev/tty	/dev/tty	5	0	system:/dev/tty
/dev/console	/dev/console	5	1	system:console
/dev/ptmx	/dev/ptmx	5	2	system
/dev/vc/0	/dev/vc/0	4	0	system:vtmaster
serial	/dev/ttyS	4	64-111	serial
pty_slave	/dev/pts	136	0-1048575	pty:slave
pty_master	/dev/ptm	128	0-1048575	pty:master
pty_slave	/dev/ttyp	3	0-255	pty:slave
pty_master	/dev/pty	2	0-255	pty:master
unknown	/dev/tty	4	1-63	console

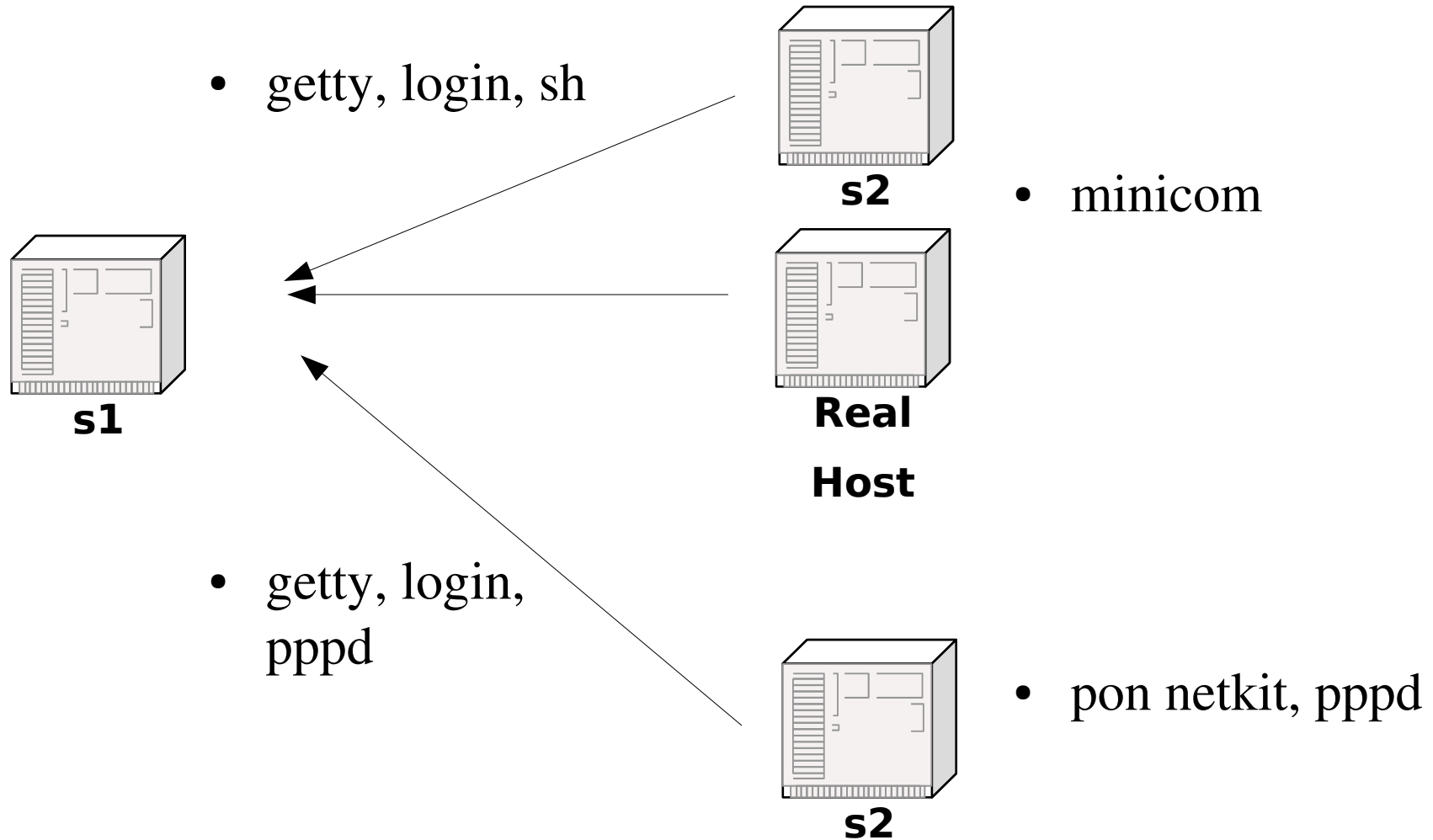
## step 3 – starting the lab



```
host machine
user@realHost:~$ cd serial
user@realHost:~/serial$ cp netkit4tic.ds \
                        ~/.devilspie
user@realHost:~/serial$ devilspie &
user@realHost:~/serial$ lstart
```

- upon launching the lab
  - 2 virtual machines are started
  - getty is started on serial /dev/ttyS0 in s1 node
  - node s2 is ready to test serial communication with s1 on its serial /dev/ttyS0 using minicom or using pppd.
  - also real host is ready to test serial communication with s1 on its /dev/tty0 using minicom

## step 4 – involved software



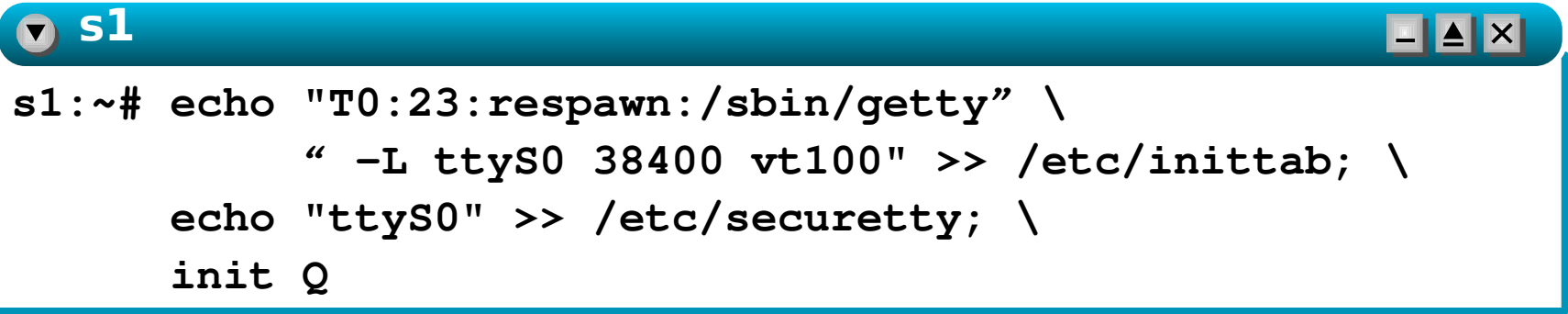
## step 5 – configure getty

- make a local serial line and attach to pseudo terminal



```
s1:~# mknod /dev/ttyS0 c 4 64; \  
      stty -F /dev/ttyS0 38400; \  
      dmesg | grep Serial; \  
      dmesg | grep "assigned console"
```

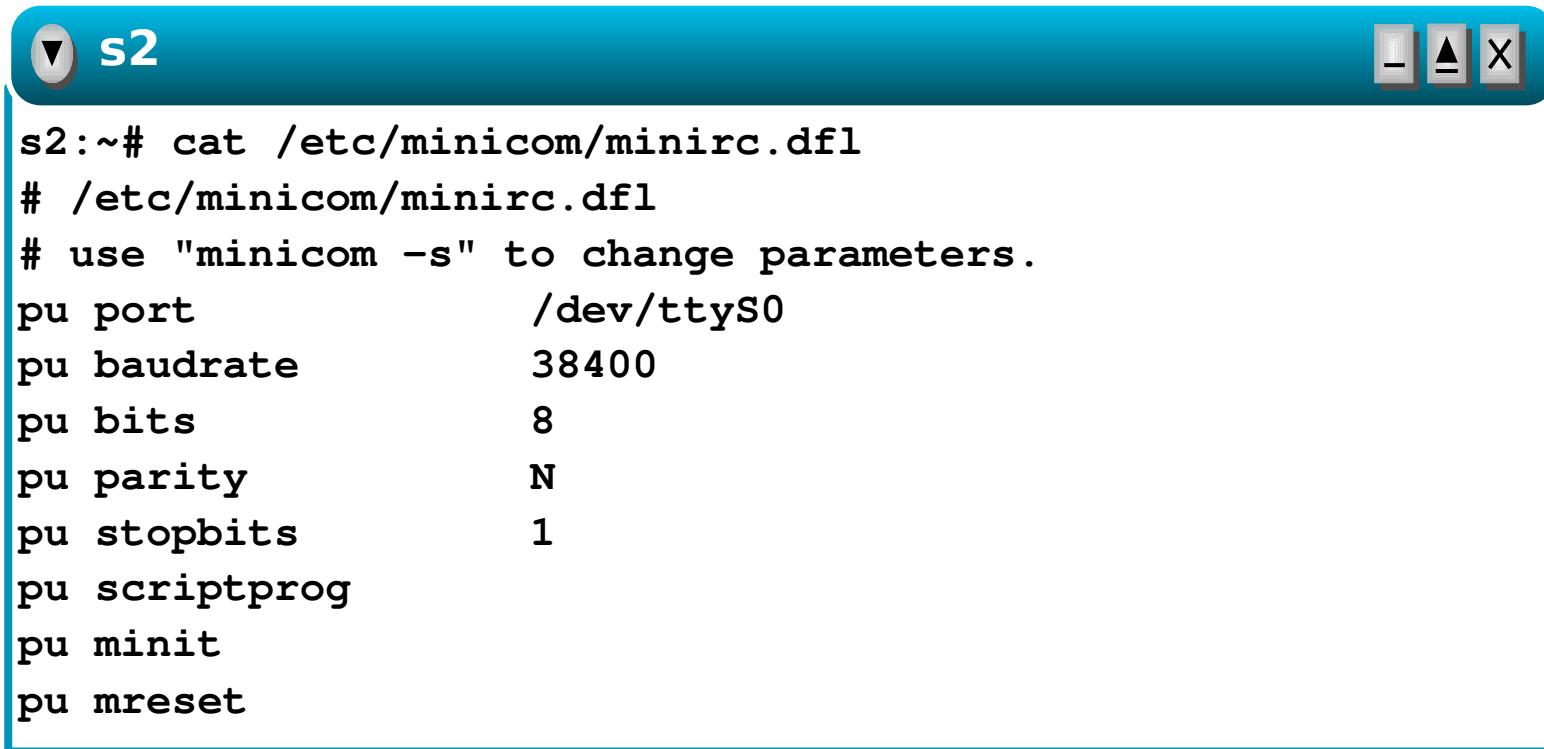
- configure getty to startup on above serial line, enable secure serial line and signals the changes to daemon:



```
s1:~# echo "T0:23:respawn:/sbin/getty" \  
      " -L ttyS0 38400 vt100" >> /etc/inittab; \  
      echo "ttyS0" >> /etc/securetty; \  
      init Q
```

## step 5 – configure minicom

- investigate the configuration of minicom (null modem)

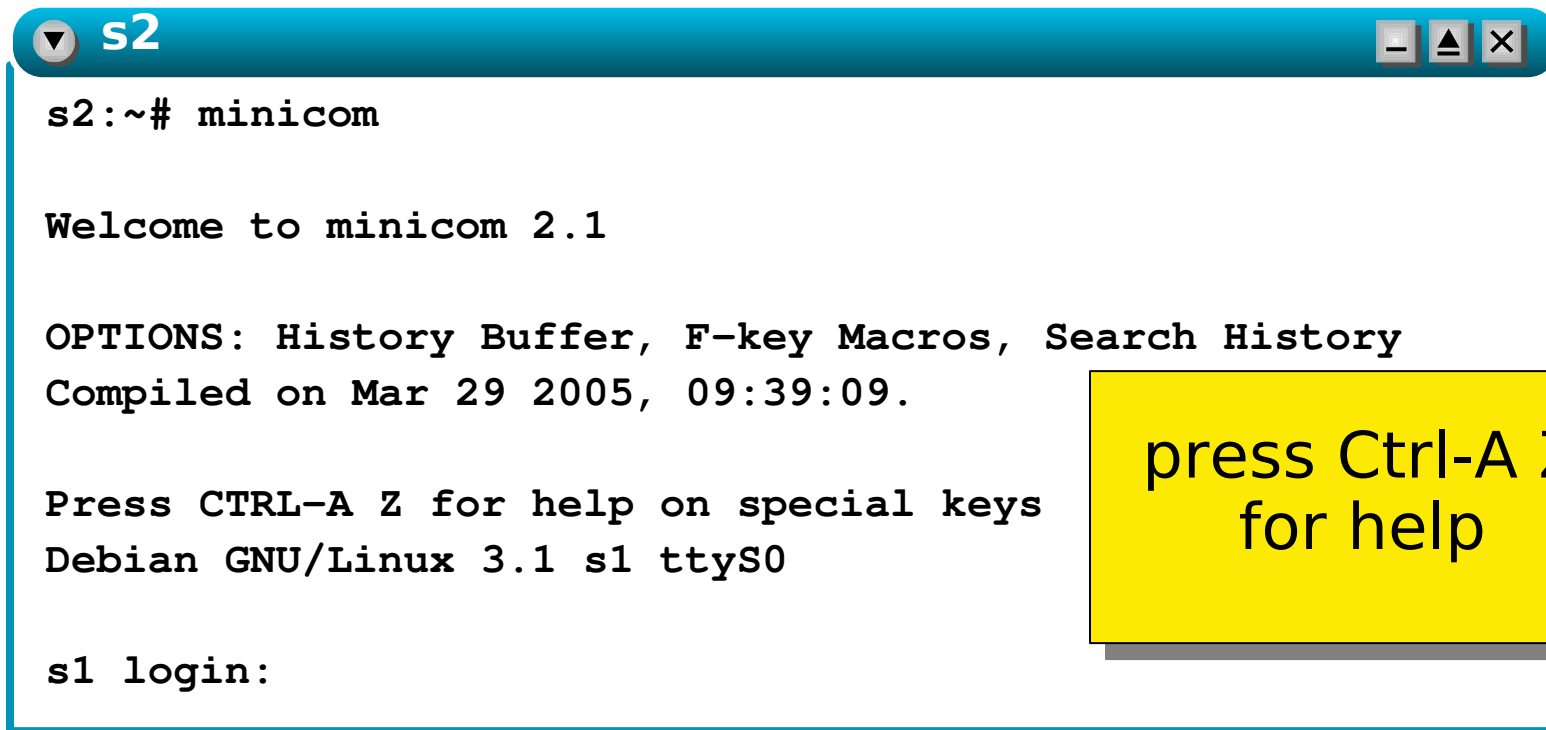
A terminal window with a blue title bar containing a dropdown arrow, the text 's2', and three window control buttons (minimize, maximize, close). The terminal content shows the output of the 'cat' command for the minicom configuration file.

```
s2:~# cat /etc/minicom/minirc.dfl
# /etc/minicom/minirc.dfl
# use "minicom -s" to change parameters.
pu port                /dev/ttyS0
pu baudrate            38400
pu bits                8
pu parity              N
pu stopbits            1
pu scriptprog
pu minit
pu mreset
```



## step 5 – test getty/minicom from s2

- configure minicom with system-wide configuration (null-modem):  
/etc/minicom/minirc.dfl
- start minicom on **s2**.



A terminal window titled "s2" with standard window controls (minimize, maximize, close) in the top right corner. The terminal displays the output of the "minicom" command. The text shown is: "s2:~# minicom", "Welcome to minicom 2.1", "OPTIONS: History Buffer, F-key Macros, Search History", "Compiled on Mar 29 2005, 09:39:09.", "Press CTRL-A Z for help on special keys", "Debian GNU/Linux 3.1 s1 ttyS0", and "s1 login:". A yellow callout box with a folded corner is positioned to the right of the terminal, containing the text "press Ctrl-A Z for help".

```
s2:~# minicom

Welcome to minicom 2.1

OPTIONS: History Buffer, F-key Macros, Search History
Compiled on Mar 29 2005, 09:39:09.

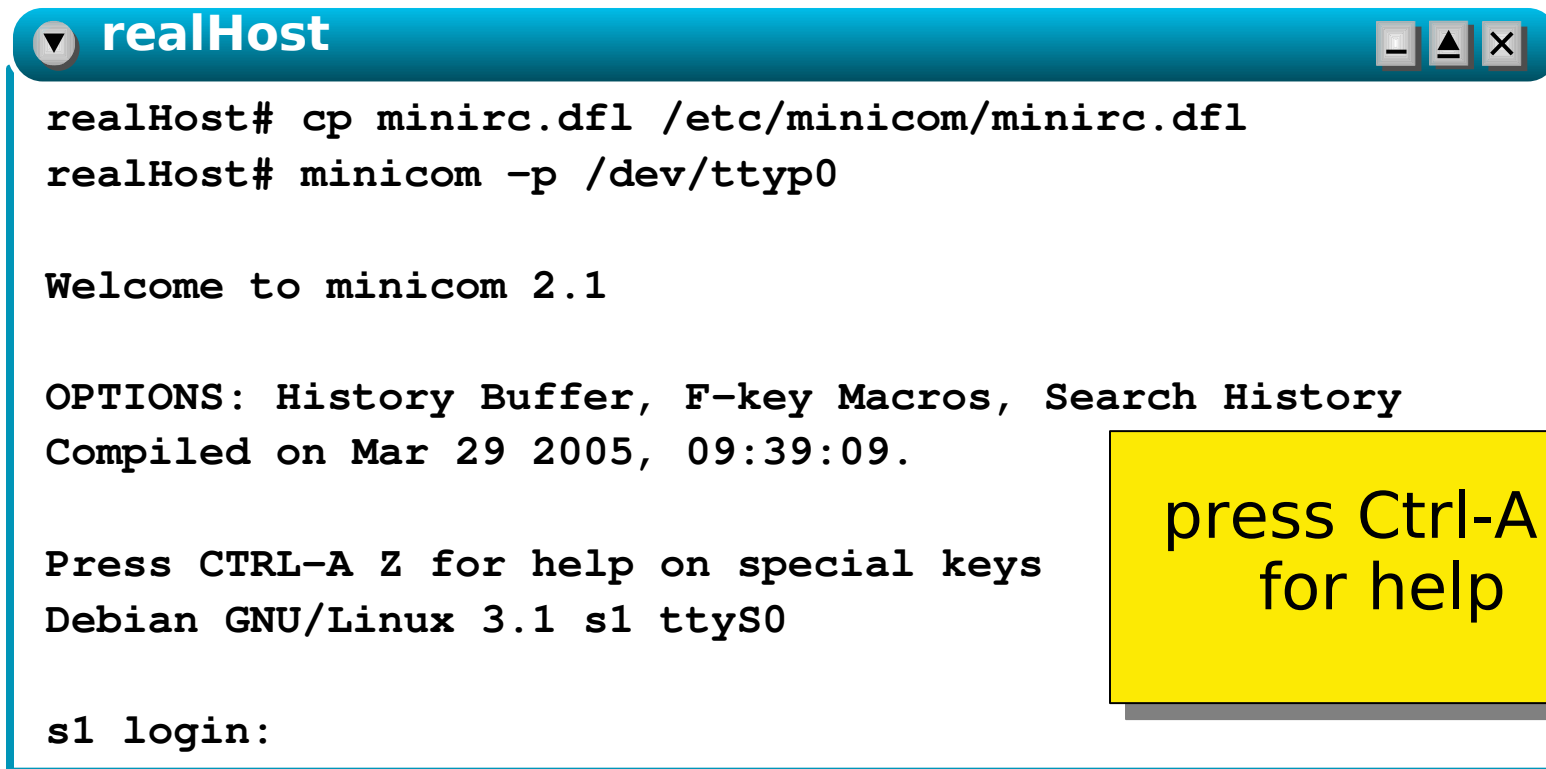
Press CTRL-A Z for help on special keys
Debian GNU/Linux 3.1 s1 ttyS0

s1 login:
```

press Ctrl-A Z  
for help

## step 5 – test getty/minicom from realHost

- configure minicom with system-wide configuration (null-modem):  
/etc/minicom/minirc.dfl
- start minicom on **real host**.



```
realHost# cp minirc.dfl /etc/minicom/minirc.dfl
realHost# minicom -p /dev/ttyp0

Welcome to minicom 2.1

OPTIONS: History Buffer, F-key Macros, Search History
Compiled on Mar 29 2005, 09:39:09.

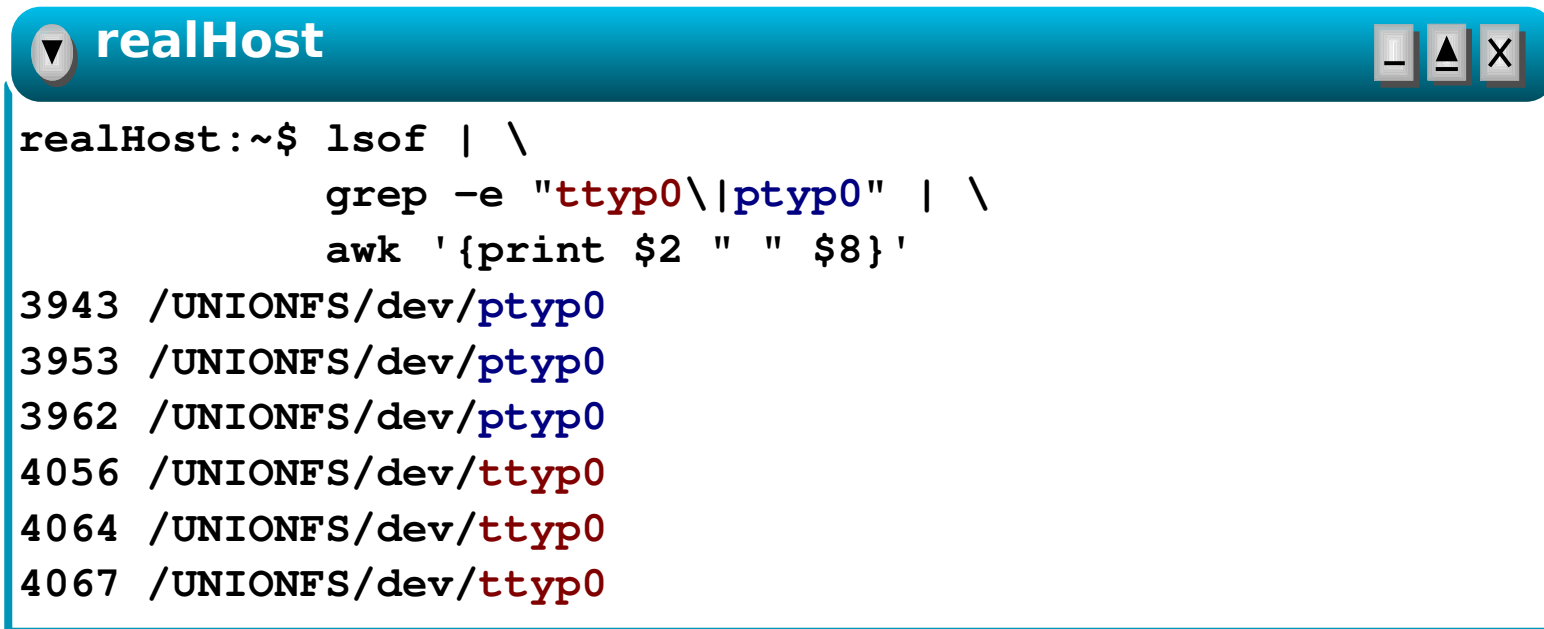
Press CTRL-A Z for help on special keys
Debian GNU/Linux 3.1 s1 ttyS0

s1 login:
```

press Ctrl-A Z  
for help

## step 6 – verify

- investigate the use of pseudo terminals inside host machine (list pid and device):



```
realHost:~$ lsof | \
    grep -e "ttyp0\|ptyp0" | \
    awk '{print $2 " " " $8}'
3943 /UNIONFS/dev/ptyp0
3953 /UNIONFS/dev/ptyp0
3962 /UNIONFS/dev/ptyp0
4056 /UNIONFS/dev/ttyp0
4064 /UNIONFS/dev/ttyp0
4067 /UNIONFS/dev/ttyp0
```

## step 7 – configure getty/pppd

- configure node s1 to act as server:
  - configure syslog to log in dedicated file /var/log/ppp.log
  - configure getty and fire up on /dev/ttyS0
  - create a new user fppp (add group dip)
  - create a logon script for the above user to startup automatically pppd
  - define pppd options
- configure node s2 to act as client:
  - configure initial chat
  - define netkit peer
  - define pppd options

## step 8 – configure s1

- configure syslogd to redirect ppp logs into a private file:



```
s1:~# grep ppp /etc/syslog.conf
local2.*                -/var/log/ppp.log
```

- restart system logging utilities:



```
s1:~# /etc/init.d/sysklogd restart
```

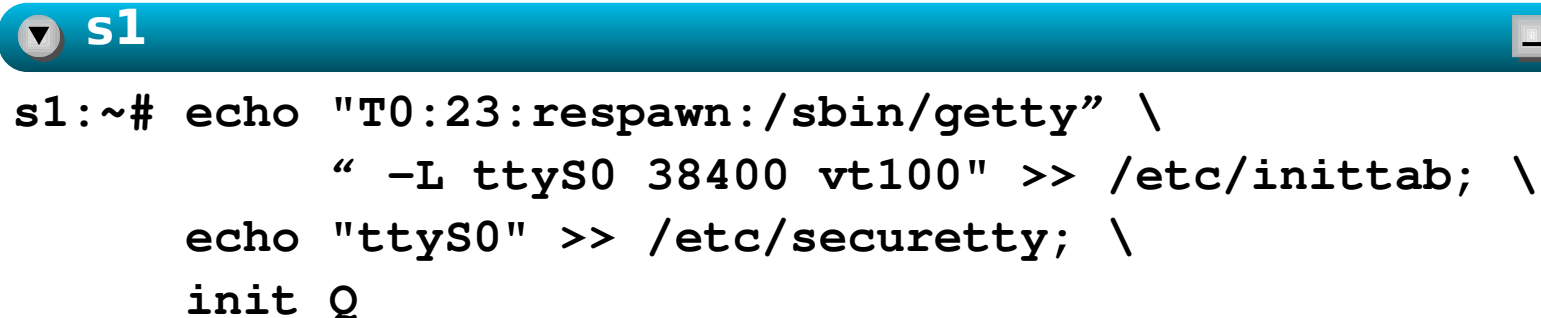
## step 8 – configure s1

- make a local serial line and attach to pseudo terminal



```
s1:~# mknod /dev/ttyS0 c 4 64; \  
      stty -F /dev/ttyS0 38400; \  
      dmesg | grep Serial; \  
      dmesg | grep "assigned console"
```

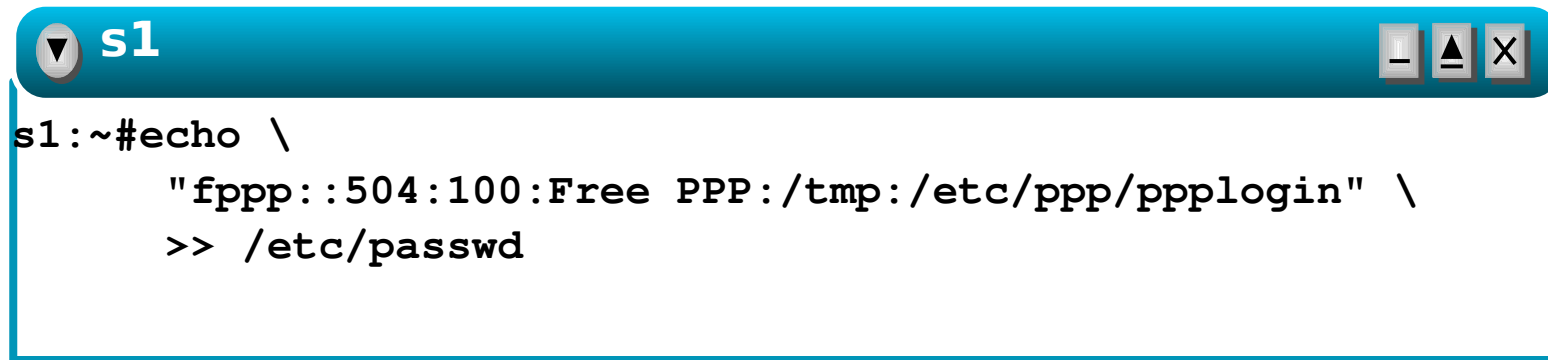
- configure getty to startup on above serial line, enable secure serial line and signals the changes to daemon:



```
s1:~# echo "T0:23:respawn:/sbin/getty" \  
      " -L ttyS0 38400 vt100" >> /etc/inittab; \  
      echo "ttyS0" >> /etc/securetty; \  
      init Q
```

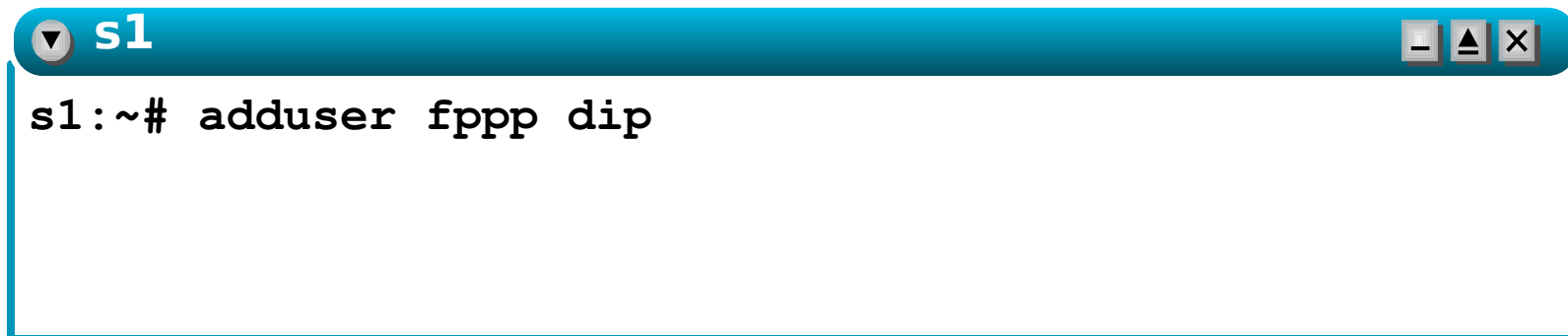
## step 8 – configure s1

- create a new user fppp



```
s1:~#echo \  
    "fppp::504:100:Free PPP:/tmp:/etc/ppp/ppplogin" \  
    >> /etc/passwd
```

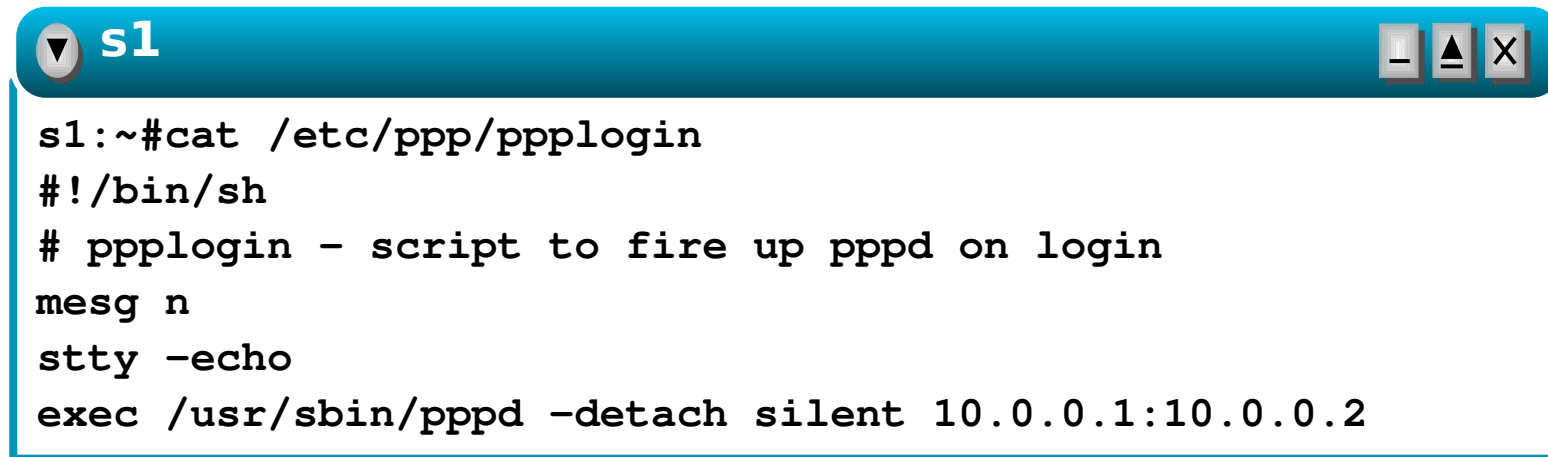
- add fppp user to dip group (Dialup IP):



```
s1:~# adduser fppp dip
```

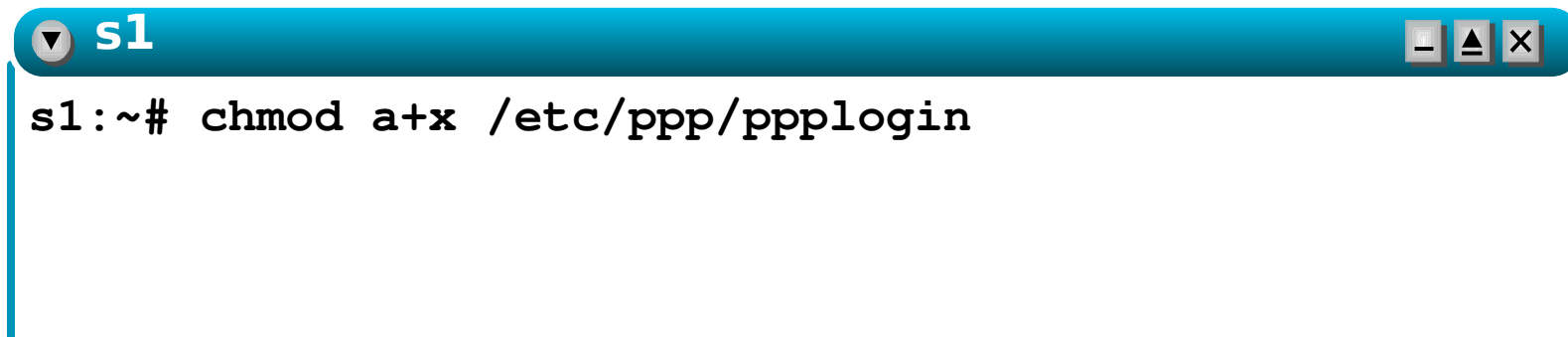
## step 8 – configure s1

- create a logon script for fppp user:



```
s1:~#cat /etc/ppp/ppplogin
#!/bin/sh
# ppplogin - script to fire up pppd on login
mesg n
stty -echo
exec /usr/sbin/pppd -detach silent 10.0.0.1:10.0.0.2
```

- make it executable:

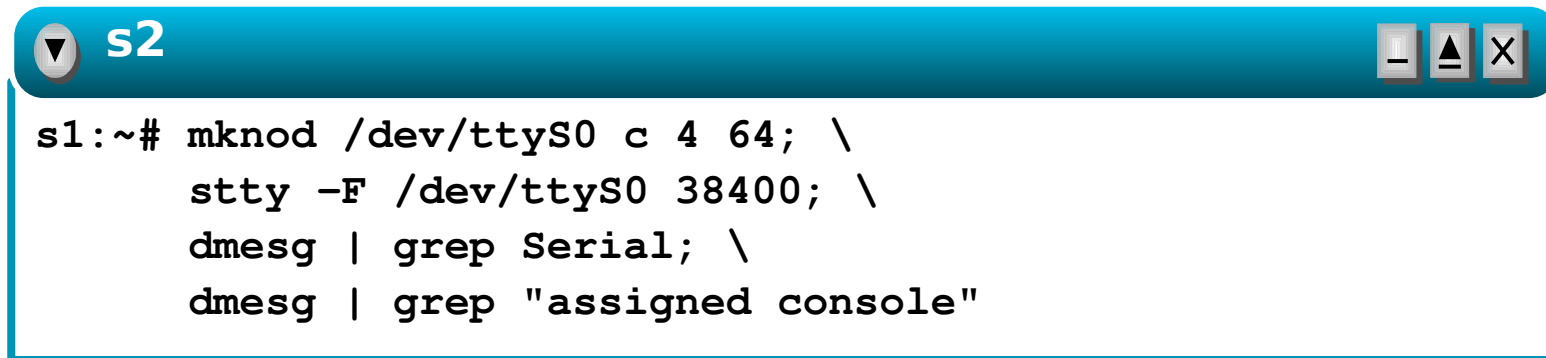


```
s1:~# chmod a+x /etc/ppp/ppplogin
```



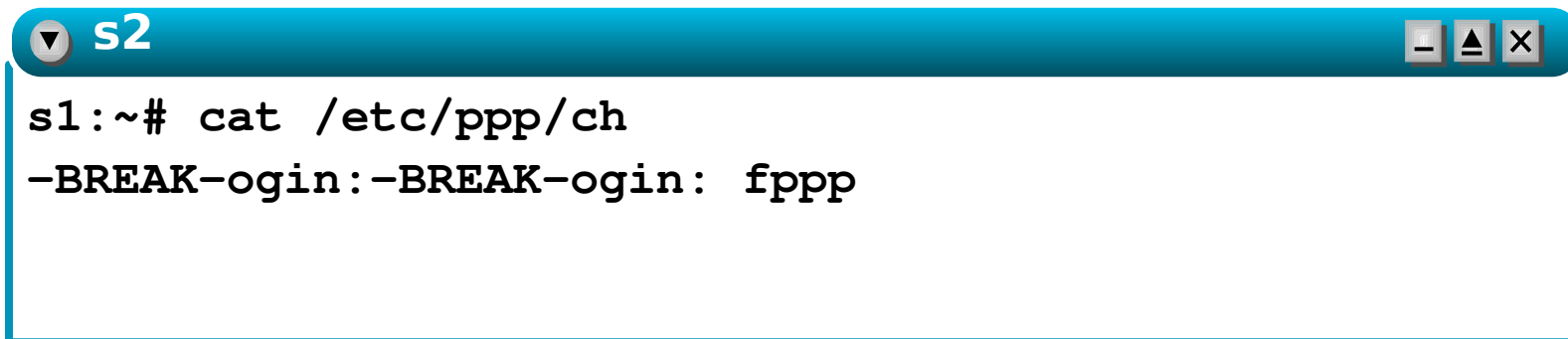
## step 9 – configure s2

- make a local serial line and attach to pseudo terminal



```
s1:~# mknod /dev/ttyS0 c 4 64; \  
      stty -F /dev/ttyS0 38400; \  
      dmesg | grep Serial; \  
      dmesg | grep "assigned console"
```

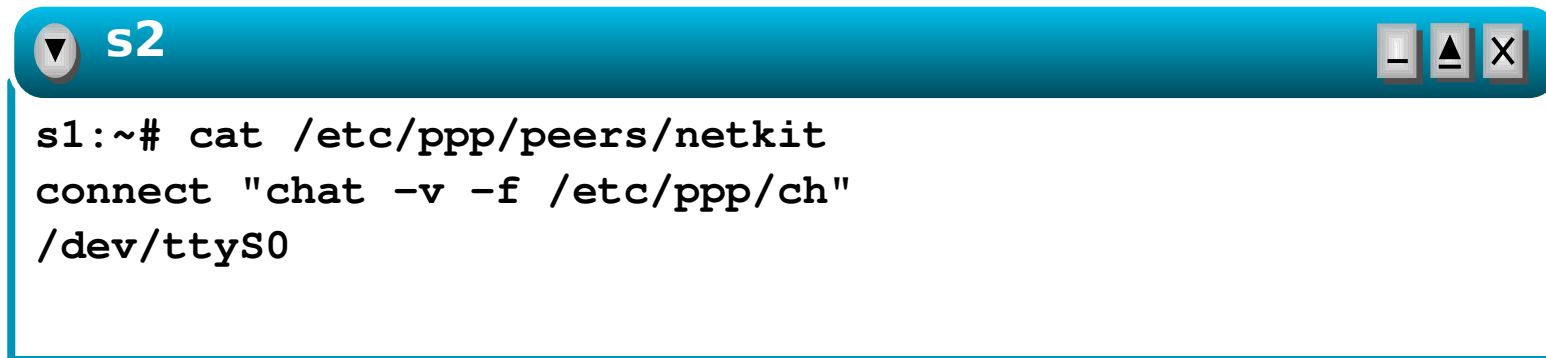
- configure initial chat:



```
s1:~# cat /etc/ppp/ch  
-BREAK-ogin:-BREAK-ogin: fppp
```

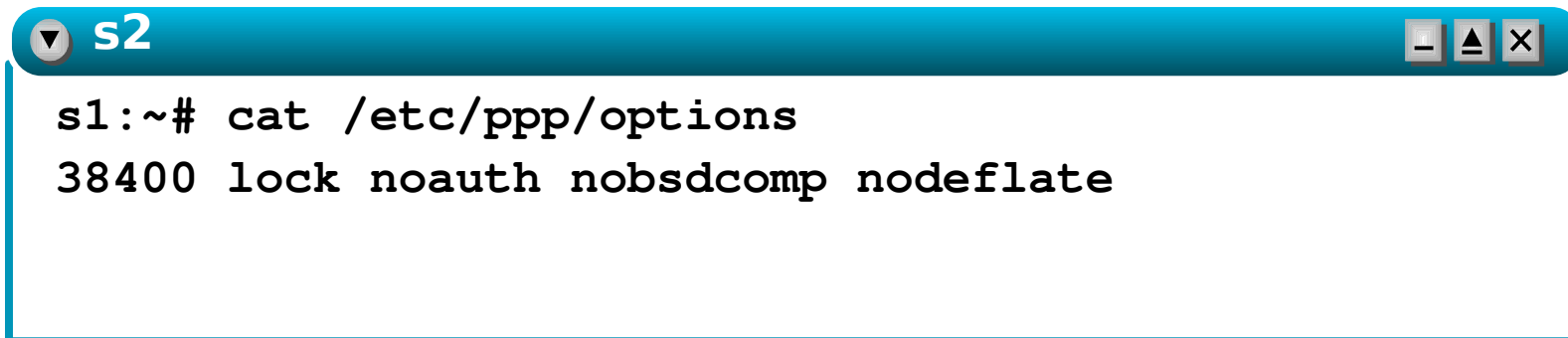
## step 9 – configure s2

- define netkit peer:



```
s1:~# cat /etc/ppp/peers/netkit  
connect "chat -v -f /etc/ppp/ch  
/dev/ttyS0"
```

- define ppp options:



```
s1:~# cat /etc/ppp/options  
38400 lock noauth nobsdcomp nodeflate
```

# step 10 – test

- connect s2 to s1 via ppp:

▼ s2

```
s2:~# pon netkit
s2:~# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.0.0.2  P-t-P:10.0.0.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:1 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:42 (42.0 b)  TX bytes:48 (48.0 b)

s2:~# ping -c 1 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=12.2 ms

--- 10.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 12.200/12.200/12.200/0.000 ms
```

# step 10 – test

- verify from s1 node:

```
s1:~# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.0.0.1  P-t-P:10.0.0.2  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:132 (132.0 b)  TX bytes:126 (126.0 b)

s1:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=12.1 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 12.131/12.131/12.131/0.000 ms
```