A skype-server under Ubuntu 6.10

The thought behind this is to have a computer just running the skype software and providing telephone connectivity to the telephones connected via the USB telbox.

I wanted a computer which I also could control remote via VNC-sessions – and for this setup x11vnc seemed the best solution. Feel free to skip the x11vnc-install and configuration part if you don't have any need for it.

NB! most of the commands below need to be run as **root**! either do this by

- writing sudo in front of each and every command or
- write sudo bash in a konsole-window right from the start

The System

- install Kubuntu 6.10 from the cd
- add the user, in this case I used **administrator**
- with the installed package manager ***adept*** remove stuff not needed on the server (like amarok and such stuff
  - but do not remove the likes of kmix!!! you will need it to test the normal soundsystem ...
  - edit /etc/apt/sources.list and enable more repositories – most of the ones in the default file, you will need them later on!
- install kpackage if you experience the same troubles viewing ALL packages as I did in adept and want to have a nice graphical way instead of the usual apt-get-commandline interface for the next step
- with the help of kpackage remove even more un-needed stuff like openoffice, gimp and so on – it is MUCH easier to see what can be removed in this graphical way IMHO ...

x11vnc

- install x11vnc (either via kpackage or via apt-get install x11vnc – this one is one of them NOT in the standard repositories so doublecheck /etc/apt/sources.list!)
- configure x11vnc
- make a directory .vnc (mkdir ~/.vnc)
- store a password for the user (x11vnc --storepasswd .vnc/passwd)

- make a logfile for x11vnc
    - touch .vnc/skype:0.log
- copy the .x11vncrc from the backup-cd or create the following file
- /home/administrator/.x11vncrc should look like this
- (if you want to restrict the access to certain computers in the net, list them on the allow-line)

```
forever

#localhost

rfbauth /home/administrator/.vnc/passwd

display :0

#allow x.y.z.a

nolookup

logappend /home/administrator/.vnc/skype:0.log
```

- copy the sharex11vnc-skript back from the backup-cd or copy the following into a new file
- (the stuff is found at http://www.ubuntuforums.org/showthread.php?t=45565)
- sharex11vnc should contain the following:

```
#!/bin/sh


x11vnc -nap -bg -many -rfbauth ~/.vnc/passwd -desktop "VNC ${USER}@${HOSTNAME}"|grep \ -Eo "[0-
9]{4}">~/.vnc/port.txt


# comment out the following if you don't want a popup telling you which port you're using.

#zenity --info --text="Your VNC port is `cat ~/.vnc/port.txt`"
```

- you can remove the \ - it's just to make sure there is no newline there
- save it to /usr/local/bin/sharex11vnc and make it executable as
- chmod 755 /usr/local/bin/sharex11vnc

the system …

- add auto-inlog of the user administrator (in the system configuration – advanced – login manager)
- make sure the script sharex11vnc is run every time the user logs in, use a symlink in .kde/Autostart
    - ln -s /usr/local/bin/sharex11vnc /home/administrator/.kde/Autostart/sharex11vnc
- make a general update of your system with adept or any other apt-get-frontend you like
- EVERYTHING installed should be updated now
- reboot
- make a thourough check of the system right now
    - does the reboot work?
    - is the user logged in correctly?
    - can you actually run some VNC-session to the computer?
    - and so on ...
    - it really will help later on if you know that so far everyting works as expected! :)

Skype

- fetch skype
- http://www.skype.com
- get the dynamic version
- http://www.skype.com/go/getskype-linux-dynamic
- install skype
    - in the directory where you saved the file run the following:
        - bunzip2 skype*
        - tar -xvf *skype**.tar
    - create a symbolic link so that you'll have a shorter name in the scripts and also should be able to upgrade skype easier
        - ln -s *skype-some-version-or-other* skype

- test skype
    - skype/skype&
- configure skype – for example special ports, etc ...
- carefully try out and test the sound with the normal soundcard and a headset
- close and exit skype

The USB-telbox

This part is about installling and configuring the two packages needed for the USB-telbox to work.

The first part is a modified version of the OpenSource usbb2k_api and the other an application using x11-messaging instead of the SkypeMate/Skype standard dbus-messaging bus.

Both programs are found in the same thread at the skype forums

to compile and install the first part you will need the following stuff installed:

(install via kpackage or apt-get install *programname)*

- automake 1.9
- libusb both
- libusb-dev and
- lib-usb++-dev

the other part needs even more stuff installed: all of

- g++
- xorg-dev (which gives us the X includes)
- libjpeg62-dev
- libqt3-headers
- libqt3-mt-dev
- plus everything they install as dependencies!

- kdelibs
- kdelibs4-dev

if you have all this installed, you can begin to build the stuff :)

- go to the forum and fetch the stuff, or use the links below directly
- http://forum.skype.com/index.php?showtopic=67560
- fetch
    - http://sonar-fs.lboro.ac.uk/usbb2k-api-mod.tar.bz2
    - and
    - http://sonar-fs.lboro.ac.uk/kb2kskype-0.1.1.tar.bz2

- create a directory build (mkdir build)
- copy the files there and unpack them
- cp usbb2k-api-mod.tar.bz2 build
- cp kb2kskype.tar.bz2 build
- cd build
- bunzip2 *
- tar -xvf *.tar

  you can also use tar -xjf kb2kskype.tar.bz2

begin with the usbb2k-api which will provide the main connectivity to the telbox

usbb2k-api

- cd usbb2k-api
- if you have installed all packages above it should pose no problem to compile and install the package.

```
./configure

make

make install
```

- check if the programs really GOT installed (I have had troubles with this!)

```
find / -type f -name usbb2k-api and

find / -type f -name api_connect
```

- didn't they get installed, copy them manually to /usr/local/bin

```
cp bin/* /usr/local/bin
```

- do the testing suggested in the README-file and finish only after veryfying basic functionality!
- check for troubleshooting below ...
- the relevant part of the README file from this package is as follows:

  - # Test API
  -
  - 1. in a Consol:
  - src/usbb2k_api
  -
  - 2. in a other Consol:
  - tools/api_connect /tmp/usbb2k.sock
  -
  - #Commande for api_connect:
  - SWITCH USB/PSTN
  - RING 0 (stop ringing)
  - RING 1 (ring mode 1)
  - RING 2 (ring mode 2)
  -
  - #Msg from api_connect:

  - HANDSET ON/OFF (pickup/off handset)
  - KEY 01..09 (keyphone pressed)

kb2kskype

This is the part doing the communication between skype and the other program and thus the USB telbox. It uses X11-messaging instead of the normal standard skype way of dbus-messaging.

- change to the kb2kskype-directory

```
cd ../kb2kskype
```

- if you did install all of the above it should not be any problem to compile and install the stuff now

```
./configure

make

make install
```

- check that the program is installed

```
find / -type f -name kb2kskype
```

- make a symbolic link from /usr/local/kde/bin/kb2kskype to /usr/local/bin/kb2kskype so you can find all of the programs in the same directory :) (easier for the scripting and backup)
  - ln -s /usr/local/kde/bin/kb2kskype /usr/local/bin/kb2kskype
- it's really a good thing to reboot the computer here ... or do whatever that will reload newly installed stuff and programs and the graphical environment (which probably got much updated above!)

ALSA

ALSA is needed for the USB telbox and skype. This part has given me headaches!

See the section about troubleshooting below!

Remember, most of this you must run as root!

- check that alsa-utils is installed, if not, install it

```
apt-get install alsa-utils
```

- it's probably good to install the following as well:

```
alsa-tools-gui
alsamixergui
```

- show all installed soundcards

```
asoundconf list
```

- make a list with all the names of the soundcards the system knows about

```
alsactl names
```

- usually your normal soundcard should get index 0 and the USB-thingy index 1 – thus

```
alsamixer -c0
```

  stop with the ESC-key

```
alsactl store 0

alsamixer -c1
```

  (here all four (?) should be set to maximum. use arrow up and down to change the volume, arrow left and right to select the right thing and the tab-key to switch between input, output or all devices)

  stop with the ESC-key

```
alsactl store 1

alsactl store
asoundconf set-default-card default
```

- cross fingers that the sound works now! ...

Skype ...

- start skype again and configure it to use ALSA and the USB-VOIP-soundcard.
- Tools-Options-Sound
- – ALSA
- – USB VOIP

- make any other changes needed to skype (like disabling popups, staying logged in (set timeout for away and such to 0)

The system ...

- to make things a bit easier, use the following shellscripts

- create a system service (run by root at boot time) to start the first part of the interaction automatically – and also giving a nice and proper way to shut it down and start it again at will if needed ...

  cut & paste the followin into a file /etc/init.d/usbb2k_api

```
#! /bin/sh

### BEGIN INIT INFO

# Provides: usbb2k_api

# Required-Start: 

# Required-Stop:

# Should-Start: 

# Should-Stop:

# Default-Start: 

# Default-Stop:

# Short-Description: init the usbb2k_api

# Description:

### END INIT INFO



PATH=/sbin:/bin:/usr/bin:/usr/local/bin:/home/administrator/bin



do_start() {

#

# create the socket in /tmp

#

/usr/local/bin/usbb2k_api & > /dev/null 1>&2
sleep 5

}


case "$1" in

start)

do_start

;;

restart|reload|force-reload)

echo "Error: argument '$1' not supported" >&2

exit 3

;;

stop)

killall kb2kskype > /dev/null 1>&2

killall skype > /dev/null 1>&2

if [ -e /var/run/usbb2k_api.pid ]; then

kill `cat /var/run/usbb2k_api.pid` > /dev/null 1>&2

sleep 5

rm /var/run/usbb2k_api.pid > /dev/null 1>&2

fi

sleep 3

if [ -e /tmp/usbb2k.sock ]; then

rm /tmp/usbb2k.sock > /dev/null 1>&2

fi

exit 0

;;

*)

echo "Usage: usbb2k_api [start|stop]" >&2

exit 3
```

```
;;
esac
```

- activate this in the proper runlevels, in the case of Kubuntu I choose 2,3,4 and 5 (the graphical way is in system configuration – advanced – system services and root mode -> search for usbb2k_api and mark it to be started at boot time)
- the script is also very handy for testing and troubleshooting purposes ...

  ```
  /etc/init.d/usbb2k_api start or /etc/init.d/usbb2k_api stop
  ```

- create a script */home/administrator/bin/skype.sh* to start both skype and the skype-helper program compiled above in an ordered fashion
- cut and paste the following into the file (you might have to change all “sleep x”- stuff, I have not a very fast PC) :

```
#!/bin/bash

/home/administrator/skype/skype& > /dev/null 1>&2
sleep 20
/usr/local/bin/kb2kskype& > /dev/null 1>&2
sleep 5
exit 0
```

- if you create a symbolic link in .kde/Autostart this will be started at boot

  dvs

  ```
  ln -s /home/administrator/bin/skype.sh /home/administrator/.kde/Autostart/skype.sh
  ```

- ***but don't do it before you are sure EVERYTING works! nightmares otherwise :(...***

Troubleshooting

The test of the usbb2k_api didn't work?

- stop usbb2k_api
- /etc/init.d/usbb2k_api stop
- start it again
- /etc/init.d/usbb2k_api start
- redo the test above, from api_connect /tmp/usbb2k.sock and onwards
- if nothing works, try the "brute-force-attack" below

The telbox is "stuck" in "line" mode even when testing?

- stop usbb2k_api
- /etc/init.d/usbb2k_api stop
- run all of the shellscripts above step by step and check the relevant logs for info
- you can have them monitored by

  tail -f /var/log/messages, tail -f /var/log/daemon.log, tail -f /var/log/kern.log

  in different shell sessions
- especially when testing check the logs for any "USB disconnected" "crashed" information
- if nothing works, try the "brute-force-attack" below

Problems with ALSA, especially when trying to save the values with alsactl store?

- try the "brute-force-attack" below

Troubleshoot!

do this step by step and cross your fingers ...

- shut down the usbb2k_api
- /etc/init.d/usbb2k_api stop
- shut down the computer
- remove the telbox
- restart the computer
- configure only the soundcard left in the computer (see the steps above! make sure you do all of the steps (list names, configure one card, store the values etc)
- shut down the computer
- plug in the telbox
- start the computer
- re-configure the telbox
- start with the api-testing (see above)
- and when that works then do the
- alsa-configuration for first the normal and then the USB-soundcard (see above)