Installing XR32 Router in Wine under Ubuntu Linux (Dick ZS6RO/ZS0MEE)

The XR32 Router is a Windows 32-bit program. I am successfully operating the Windows XR32 Router on a Linux operating system using the Wine program and it's not difficult to duplicate this effort - and its extremely stable. This configuration has been tested on Ubuntu 10.04, 11.04, 12.04 and 12.10 to date.

The **Wine** program is **not** an Emulator! It is a translation layer (or a program loader) capable of running Windows applications on Linux and similar other POSIX-compatible operating systems. Wine does **not** emulate Windows applications on Linux – instead it provides alternative implementations of DLLs that a typical Windows application calls and a process substitute for the Windows NT kernel. Wine is made of 100 percent Microsoft-free code.

This is my hardware and software list, but XR32 should work on different flavours of Linux distro's without any problems. My server carries a lot of other services which have not been compromised with the XR32 configuration.

Hardware:	1 x PC Intel +2GHz,	1 x 4GB RAM,	1 x 320GB hard drive	1 x DVD W/R drive,
	1 x 10/100/1000Mb/s Ethernet port,		4 x USB ports,	5 x RS-232 ports.
Software:	Linux Ubuntu Version 12.10, with KDE installed – (GNOME should work as well).			
	Wine 1.41,	XR32 Version 2.01b.		

It is assumed your Linux box is capable of reaching the Internet. It is also assumed you know Linux sufficiently well enough to be able to change things like file/directory permissions/ownerships etc., and know your way around the KDE or GNOME GUI.

Start by downloading and installing Wine. To do this, open a terminal window in Ubuntu 12.10 and type;

'***sudo apt-get update***<enter>'	this updates the repository database, then type;
'***sudo apt-get install wine***<enter>'	this will fetch the current program from the Ubuntu repository database from the Internet and install it.

After the install type '***wine***<enter> in a terminal window - a usage help prompt should appear – Wine is installed and the Wine defaults should be fine - it defaults to Windows XP, which is what the XR32 Router appears to like.

Follow the instructions elsewhere in this document to install the XR32 software to the hard drive.
I created a directory called **/home/xr32** and downloaded all the relevant XR32 files into this folder.
Make the XR32.EXE file executable with '***sudo chmod +x /home/xr32/XR32.EXE'***.
(Note that all the files are owned by root).

The **XR32.EXE** and the **XROUTER.CFG** files are required to run XR32 - the rest of the files can be added later. Now edit the XROUTER.CFG file with your particulars – don't worry about ports and interfaces yet – If you are using the example XROUTER.CFG file '*rem*' them out with '**;**' in the left-hand side of each relevant row for now.

In the **XROUTER.CFG** file ensure you have the following entries in there. XR32 will complain if the loopback interface and port are not entered and will not start up. Additional interfaces and ports can be added later.

```
;-----------------------------------------------------------------------
;       loopback interface, allowing self-connects.
;-----------------------------------------------------------------------
;
;       Example 'loopback' interface, allowing self-connects
;       without going via an external system
;
INTERFACE=99
        TYPE=LOOPBACK
        ID=Loopback Interface
        PROTOCOL=KISS
        MTU=256
ENDINTERFACE
;
```

```
;-----------------------------------------------------------------------
;
PORT=99
        ID=Internal   - KISS       Internal Loopback.
        INTERFACENUM=99                  ; Loopback, KISS
        MHEARD=5
        QUALITY=220
ENDPORT
;
```

Now create an **IPROUTES.SYS** text file in ***/home/xr32***' and place the contents below into this file.

```
; ZS0MEE-2:MEEXR    IPROUTE.SYS    Last modified: 20-Jan-13   ZS6RO
;
; <mode> d       Datagram (use on very good quality links only).
;              v       Virtual circuit (better for links with retries).
;              n       Netrom (for "tunnelling" through netrom-only networks).
;              e       Encapsulated (e.g. ham IP carried by Internet IP).
;              w       Route everything via Linux kernel.
;
; ROUTE    <default>     <portnum> [<gateway>  [<mode>] ]
; =============================================
IP ROUTE   DEFAULT   0                *                  w
;
```

In Windows the **NdisXpkt** driver allows XR32 to share the Ethernet NIC and have its own IP address, but is only available for Windows 2000 and XP.   The **NdisXpkt** driver is not used in Linux.

The entry added above into the IPROUTE.SYS file allows the XR32 Router to route everything to the Linux kernel.   After editing the IPROUTE.SYS file and with XR32 still running, type '**IP ROUTE LOAD**<enter>' in one of the three consoles provided in XR32, this will update the XR32 routing tables. From one of the XR32 console windows you should now be able to ping Internet domains, like 'ping google.com'.

Elsewhere in this documentation it states that 'IP ROUTE…' doesn't require the 'IP' in front of 'ROUTE…' when adding entries into the IPROUTE.SYS file.   It **IS** required in XR32 version 2.01b as indicated above in the IPROUTE.SYS file – ensure you include the 'IP'.

There are a few methods to start the Windows XR32 Router in Linux using Wine. I use the second method but either is permissible.

1. Make sure you change to the XR32 directory - in a terminal window, type '***cd /home/xr32***' to change directories. Now type '***wine start XR32.EXE***'. A window should pop up – this is the XR32 Console window. Great – you are running a Windows XR32 program in Linux with Wine. Now press '***Alt x***'. A small red window pops up in the Console window with '**Exit – are you sure**'. Type '**y**<enter>'. The XR32 Console window should close and disappear. Type '**ps -ef|grep -v grep|grep XR32.EXE**' in a terminal window. Only the Linux prompt should display. Send the same command in a separate terminal window when XR32 is running – the process number will be displayed.

2. I mentioned **KDE** and **GNOME** earlier – I use the **KDE** GUI. In the **KDE** file manager navigate to the **/home/xr32** directory. Right-click your mouse on the **XR32.EXE** file, then on the pop-up window go to '***open with***' and left-click on '***wine windows loader***'. This should open a similar XR32 Console window like in the previous paragraph. Now press '***Alt x***'. A small red window pops up in the Console window with '**Exit – are you sure**'. Type '**y**<enter>'. The XR32 Console window should close and disappear. Type '**ps -ef|grep -v grep|grep XR32.EXE**' in a terminal window. Only the Linux prompt should display. Send the same command in a terminal window when XR32 is running – the process number will be displayed.

Now follow the instruction elsewhere in this document to configure the rest of XR32 Router. Remember that in Linux, 'COM-1' is equal to 'ttyS0', 'COM-2' is equal to 'ttyS1', etc.

ooOoo

Configuring LinFBB and XR32 both running on Ubuntu Linux

Hardware:	1 x PC Intel +2GHz,	1 x 4GB RAM,	1 x 320GB hard drive	1 x DVD W/R drive,
	1 x 10/100/1000Mb/s Ethernet port,		4 x USB ports,	5 x RS-232 ports.
Software:	Linux Ubuntu Version 12.10, with KDE installed – (GNOME should work as well).			
	Wine 1.41,	XR32 Version 2.01b.	LinFBB version 7.04L	

It is assumed that you have XR32 running with Wine successfully (follow instruction elsewhere in this document) and that LinFBB is working correctly on your Linux machine.

LinFBB was probably set up to a radio port(s) via the **port.sys** file. That assumes you are using the AX25 utilities which are usually part of the Linux kernel. If you have more than one radio port on LinFBB, duplicate the instructions below for each radio port. If LinFBB has a telnet port (port 6300), that will continue to work as before even after this modification.

```
-----------------------                                                     ----------------------
I                    I                                                      I                    I----o Telnet
I                    I                        AX25                          I                    I       6300
I  RADIO PORT  I-------------------------  rem out  ----------------------------I      LinFBB       I
I                    I                     connection                       I                    I
I                    I                                                      I                    I
------------------------                                                    ------------------------
```

In LinFBB **/etc/ax25/fbb/port.sys** file, '*rem*' out the relevant radio port(s). If you have more than one radio, work on them one at a time. Assuming you have only one VHF radio port, '*rem'* it out as below. Remember to correct the 'Ports' and 'TNC' parameters. I suggest this so as to keep track of what you are doing.

```
# /etc/ax25/fbb/port.sys
#
#Ports   TNCs
#~~~~~~ ~~~~
 1         1

#Com-device   Interface     Address (Hex)      Baud
#~~~~~~~~~~~ ~~~~~~~~~~ ~~~~~~~~~~~~~~   ~~~~
 1                 9               ****                     0        # Linux AX25 Interface.

#                 Com    axports    max
#TNC NbCh Device  MultCh.   Pacln   Maxfr  NbFwd MxBloc M/P-Fwd Mode     Freq
#~~~  ~~~~  ~~~~~~ ~~~~~~   ~~~~~ ~~~~~ ~~~~~ ~~~~~~ ~~~~~~~ ~~~~      ~~~~
  0     0       0           0          0        0        0         0        00/01      ----       File-fwd.
#1     7        1          2m         80       1         2       20        10/20      XUWYL 144.550

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Now again in the **/etc/ax25/fbb/port.sys** file add the following and change the ‘TNC’ parameters.

```
# /etc/ax25/fbb/port.sys
#
#Ports   TNCs
#~~~~~ ~~~~
 1        2

#Com-device   Interface    Address (Hex)      Baud
#~~~~~~~~~   ~~~~~~~    ~~~~~~~~~~~       ~~~~
 1                 9               ****                     0        # Linux AX25 Interface.

#                  Com    axports    max
#TNC NbCh Device  MultCh.   Pacln   Maxfr  NbFwd MxBloc M/P-Fwd Mode      Freq
#~~~  ~~~~  ~~~~~~ ~~~~~~  ~~~~~ ~~~~~ ~~~~~ ~~~~~~ ~~~~~~~ ~~~~       ~~~~
# Radio ports.
  0      0        0            0            0      0        0         0        00/01      ----        File-fwd.
#1      7        1          2m          80      1         2       20       10/20      XUWYL  144.550

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# XR32 Router.
 2      10        1        xrfbb      128      1        2        1       00/01     XUWYL  XRouter

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

In the **/etc/ax25/axport** file, add the following.

```
# /etc/ax25/axports
#
#name   callsign         speed    paclen    window   description
#~~~~   ~~~~~~        ~~~~~    ~~~~~~  ~~~~~~  ~~~~~~~~~
# AXIP Tunnel.
 xrfbb     ZS0MEE-11   38400    128        1          Link (XR32<->FBB)

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

The **/etc/ax25/ax25ipd.conf** file converts between **AXIP <> AX25**. Take note of the highlighted sections.

```
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# ax25ipd configuration file for station ZS0MEE-2
#
# Select axip transport. 'ip' is what you want for compatibility
# with most other gates ...
#
# socket <ip | udp [udp_orig_port_num] > (default UDP port # is 10093)
#socket udp 93
socket ip
#
# Set ax25ipd mode of operation. (digi or tnc)
#
mode tnc
#
# If you selected digi, you must define a callsign.  If you selected
# tnc mode, the callsign is currently optional, but this may change
# in the future! (2 calls if using dual port kiss)
#
#mycall
#mycall2
```

```
#
# In digi mode, you may use an alias. (2 for dual port)
#
#myalias
#myalias2
#
# Send an ident every 540 seconds ...
#
#beacon after 540
#btext whatever you want in a beacon, but we don't use it.
#
# Serial port, or pipe connected to a kissattach
device /dev/pts/1
#
# Set the device speed
speed 38400
#
loglevel 0
#
# Broadcast Address definition. Any of the destination addresses listed
# will be forwarded to any of the routes flagged as broadcast capable
# routes.    ssid of 0 routes all ssid's
#
#broadcast QST-0 NODES-0
broadcast FBB-0 MAIL-0
#
# ax.25 route definition, define as many as you need.
# format is route (call/wildcard) (ip host at destination)
# ssid of 0 routes all ssid's
#
# Valid flags are:
#         b  - allow broadcasts to be transmitted via this route
#         d  - this route is the default route
#
# route <destcall> <destaddr> [flags]
# route <destcall> <destaddr> [udp [udp_dest_port_no] ]
#
route zs0mee-0 192.168.0.11 bd
#
# -- End of script.
```

The entry '**device /dev/pts/1**' must be exactly as shown. When everything has started up, the '**1**' may change, but the syntax must be the same for our startup file to recognize where to place the new number.

The '**route zs0mee-0 192.168.0.11 bd**' entry must match your own details. Keep the SSID as '***-0'*** as this will match any SSID. My LinFBB BBS is ***ZS0MEE***, thus **zs0mee-0** will match. The '**192.168.0.11**' is from the XR32 XROUTER.CFG file which we will edit shortly.

The Linux AX25 modules need to do some work now and create a pseudo AXIP link between LinFBB and the XR32 router. Create a startup file called **/etc/ax25/ax25-up** which is a shell script.

```
#!/bin/sh
#
# /etc/ax25/ax25-up
#
# Create a psuedo link between LinFBB and XR32 Router,
# Start AX25ipd daemon, ax25d daemon, Mheard daemon,
# Start up FBB BBS daemon, XR32 Router.
```

```
#
# Version 1.0   22 January 2013. Dick ZS6RO. [First Release]
#
#
# Loading modules first.
# ~~~~~~~~~~~~~~~~~
/sbin/modprobe mkiss
/sbin/modprobe ax25

tmpfile="/tmp/$$.startipd.pts"
# Create psuedo ports.
# ~~~~~~~~~~~~~~~~
echo "Starting kissnetd daemon."
killall kissnetd
/usr/sbin/kissnetd -p2 > "$tmpfile" &
echo $! > /run/kissnetd
echo "kissnetd daemon Started."

# Give kissnetd time to create psuedo ports.
sleep 2

# Now attach ports to relevant apps.
attachthem () {
        read PTS1 PTS2
        echo "Starting xrfbb on $PTS1 and $PTS2"
        /usr/sbin/kissattach -l $PTS1 xrfbb 192.168.0.11
        sleep 1
        # Add device paramaters to ax25ipd.conf file.
        sed -i "s,device /dev/.*$,device $PTS2," /etc/ax25/ax25ipd.conf
        #
        echo "Starting ax25ipd daemon."
        # Start AX25ipd daemon.
        # ~~~~~~~~~~~~~~~~~~
        sleep 1
          killall ax25ipd
        /usr/sbin/ax25ipd -l4
        echo $! > /run/ax25ipd.pid
        echo "ax25ipd daemon Started."
        #
        echo "Starting ax25d daemon."
        # Start ax25d daemon.
        # ~~~~~~~~~~~~~~~~
        sleep 1
          killall ax25d
        /usr/sbin/ax25d &
        echo $! > /run/ax25d.pid
        echo "ax25d daemon Started."
        #
        echo "Starting mheardd daemon."
        # Start Mheard daemon.
        # ~~~~~~~~~~~~~~~~~
        sleep 1
          killall mheardd
        /usr/sbin/mheardd &
        echo $! > /run/mheard.pid
        echo "mheardd daemon Started."
        #
        echo "Starting xfbbd daemon."
        # Start up FBB BBS daemon.
        # ~~~~~~~~~~~~~~~~~~~~~
        sleep 1
          killall xfbbd
        /usr/sbin/xfbbd &
```

```
        echo $! > /run/xfbbd.pid
        echo "xfbbd daemon Started."
        #
        echo "Starting XR32 Router."
        # Start up XR32 Router.
        # ~~~~~~~~~~~~~~~~~~~
        sleep 50
           killall XR32.EXE
         cd /home/ham_services/xr32/
        /usr/bin/wine start XR32.EXE
        echo $! > /run/XR32.pid
        echo "XR32 Router Started."
        #
}

sleep 1
tail -n 1 $tmpfile | attachthem
rm $tmpfile

echo "---"
echo "Finished starting AX25ipd daemon, AX25d daemon,";
echo "Mheard daemon, LinFBB-BBS daemon, XR32 Router.";
#

exit 0;

# --End script.
#
```

Don't start up the '**/etc/ax25/ax25-up**' shell script file until the XROUTER.CFG has been configured.

Make the '**/etc/ax25/ax25-up**' shell script file executable with '***sudo chmod +x* /etc/ax25/ax25-up**'.

The '*sleep*' commands give time for the previous command to have carried out its function before continuing.   Remember to make the '**/etc/ax25/ax25-up**' execute when the Linux machine boots up.

The highlighted section in the '/etc/ax25/ax25-up' file will create a pseudo pipe and place the slave side of the pipe into /etc/ax25/ax25ipd.conf file where I said that the syntax must be exact ('*device /dev/pts/?*').

This startup file when run will load some AX25 modules, create the pseudo pipe - add the slave end to the ax25ipd.conf file - the master end of the pipe will be fed to the Linux kernel, then start up the various daemons and finally start up the XR32 router.

When the ax25-up file has executed, XR32 will have loaded but you won't see the console !!   If you type '**ps -ef|grep -v grep|grep XR32.EXE**' in a terminal window, you will see the X32.EXE process running – take note of the process number.   Currently I don't know how to cure this.   I suggest you do what I do and that is, grep XR32.EXE as shown above, then kill the process '**kill <process-number>**', then use one of the two methods to start XR32 as mentioned earlier in this document.   One good thing out of this is if there is a power failure and you aren't nearby, at least XR32 will start and run properly when the power returns.

Most times when working on XR32 files, you only need to do an **ALT-x** to kill the XR32 process and then restart it by one of the two methods mentioned.   All other daemons will still function correctly.

Now to configure the XR32 **XROUTER.CFG** file so that LinFBB will be accessible via the XR32 Router. We need the following entries to get the radio port and the LinFBB port configured.

```
;-----------------------------------------------------------------------
;       145.550Mhz 1200bps User-channel Interface.
;-----------------------------------------------------------------------
;
INTERFACE=2
        TYPE=ASYNC
        COM=7                           ; COM7=ttyS6. hex=000. IRQ=19.
        FLOW=1                          ; RTS/CTS
        SPEED=9600                      ; Baud rate between PC and TNC
        PROTOCOL=KISS
        KISSOPTIONS=NONE
        MTU=128
ENDINTERFACE
;


;-----------------------------------------------------------------------
;       AXIP Interface for LinFBB.
;-----------------------------------------------------------------------
;
INTERFACE=10
        TYPE=AXIP
        MTU=128
ENDINTERFACE
;


;-----------------------------------------------------------------------
;
PORT=2
        ID=144.550 Mhz. 1200bps.  VHF User channel.
        INTERFACENUM=2                          ; ASYNC, KISS
        PORTCALL=ZS0MEE-2                       ; ZS0MEE-2 connects to XR32
        PORTALIAS=MEEXR                         ; MEEXR connects to XR32
        RFBAUDS=1200                            ; Radio channel baud rate
        EXCLUDE=NOCALL,PK232
        NODESINTERVAL=0                         ; Nodes broadcast time interval (min)
        QUALITY=0                               ; 0 disables L3/L4 frames.
        MINQUAL=50
        TXDELAY=450                             ; Milli-seconds
        TXTAIL=75                               ; Milli-seconds
        RETRIES=5
        FRACK=7000                              ; Milli-seconds
        PACLEN=128                              ; Adaptive paclen when paclen '0'
        MAXFRAME=1                              ; Adaptive maxframe when paclen '0'
        SLOTTIME=120
        RESPTIME=1500
        USERS=20
        SESSLIMIT=20
        MHEARD=15
        MHFLAGS=7
        DIGIFLAG=0                              ; Do not digipeat
        DIGIPORT=0                              ; Digipeat on this channel
        BCAST=FBB,MAIL                          ; Destination FBB and MAIL
        BCFROM=ZS0MEE                           ; From ZS0MEE BBS
        PIPE=11                                 ; Pipe to ZS0MEE BBS (Port-11)
        PIPEFLAG=515                            ; UI/Non-UI (*not*) + Bidirectional
ENDPORT
;
```

```
;------------------------------------------------------------------------
;
PORT=11
        ID=AXIP  Link - ZS0MEE    LinFBB BBS.
        INTERFACENUM=10                     ; AXIP
        IPLINK=192.168.0.11                 ; LinFBB BBS (ZS0MEE)
        SESSLIMIT=5
        MHEARD=10
        MHFLAGS=7
        NODESINTERVAL=0                     ; Nodes broadcast time interval (min)
        FRACK=7000                          ; Made same as radio ports (seconds)
        PACLEN=128                          ; Adaptive paclen when '0'
        MAXFRAME=1                          ; Adaptive maxframe when paclen '0'
        RETRIES=5
        QUALITY=0                           ; 0 disables L3/L4 frames.
        BCAST=FBB,MAIL
        BCFROM=ZS0MEE                       ; From ZS0MEE BBS
ENDPORT
;
```

Note the ‘**IPLINK=192.168.0.11**’ in Port 11.   This is the address that must be placed into ‘**/etc/ax25/ax25-up**’ and at the bottom of the ‘**/etc/ax25/ax25ipd.conf**‘ files.

There is a bi-directional pipe in Port 2 configured with **UI/Non-UI (*not*)** (pipeflags=515) with the other end connecting to Port 11. This allows calls from radio users on Port 2 to connect direct to the LinFBB BBS without first logging onto the XR32 Router, then connecting to the BBS. If additional radio ports are installed, a bi-directional pipe from the new radio port can also be pointed to Port 11 to allow BBS access on that radio port.

In my configuration a radio user can connect to the BBS by typing ‘**C ZS0MEE’**.   A radio user can connect to the XR32 Router by typing ‘**C ZS0MEE-2**‘ or ‘**C MEEXR**‘.

A problem I found while testing this configuration was that I had ‘digital feedback’ when the radio transmitter sent data. It was picked up by the receiver and sent to the TNC which sent it back to XR32. This caused the XR32 to repeat what was transmitted continuously until I switched the radio off.

I had initially taken the audio from my receiver, before the squelch and amplified it a bit before sending it to the TNC.   My radio was designed to block the squelch when transmitting to prevent the TX audio from being heard in the loudspeaker. I fitted an IN4148 diode and a 10k resistor in series to the base of my added transistor circuit and connected to a switched 8V TX PTT line. When TX’ing, the audio stage transistor was switched hard-on preventing any audio from passing through – no more ‘digital feedback and looping – and it was fast.

```
                                     .------------o--------RES-----------o +12V
                                     I            I
               IN4148    10k         R            I
    8 +v    o---DIODE---RES---------. E            o------------I I----------> AF OUT
   TX PTT      a       k            \ S           /
                                     \I          I/ collector
    AF IN  >-------------I I-----------------o--------I
     1mV                              I          I\  emitter
                                      R           \
                                      E            I
                                      S            I
                                       I            I
            o--------------------------------o-----------o-------------------------o GND
```

We have now ended up with a configuration where the XR32 is controlling nearly everything.   LinFBB can still have its own Telnet session via port 6300 like it did before this change and doesn't involve the XR32, but the radio port(s) would now be controlled by the XR32 Router to access LinFBB.   The LinFBB BBS is also accessible from the XR32 Router itself.

In the XROUTER.CFG file, add ‘**COMMAND=BBS C 11 ZS0MEE S**‘. This will allow any user who is connected to the XR32 Router to just type ‘**BBS**<enter>’.   This will connect the user straight to the LinFBB BBS. The ‘***C 11 ZS0MEE***’ will connect to the BBS on Port 11.   The ‘***S***’ at the end will return the user to the XR32 command prompt instead of disconnecting the user totally from the Router.   When a user types ‘**?**<enter>’, a help list is presented and at the bottom of the list will be seen the command ‘***BBS***’.

```
-----------------------              -----------------------              -----------------------
I                    I                I                    I                I                    I----o Telnet
I                    I      AX25      I                    I  AXIP<>AX25    I                    I        6300
I  RADIO PORT        I----------------I       XR32         I----------------I       LinFBB       I
I                    I                I                    I                I                    I
I                    I                I                    I                I                    I
------------------------             -----------------------              ------------------------
```

Currently I can't get the unproto boadcasts sent from LinFBB to go out of the radio port and have instructed my radio users to edit their Winpack configuration to not do sync-requests. This isn't a problem because Winpack will interrogate LinFBB on its autotimer and get a list of message/bulletin headers to work from.

My final ‘issue’ is that XR32 ‘crashes’ and closes when WinPack requests a compressed download. I've asked my radio users to download/upload in plain-text until I can figure this problem out.

ooOoo