Inhaltsverzeichnis

Packet Radio via HAMNET

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 2. April 2018, 15:08 Uhr (Quelltext anzeigen)
Oe1rsa (Diskussion | Beiträge)
← Zum vorherigen Versionsunterschied

Version vom 2. April 2018, 16:36 Uhr (Quelltext anzeigen)
Oe1rsa (Diskussion | Beiträge)
(→Zugang von Linux aus)
Zum nächsten Versionsunterschied →

Zeile 59:

ax0 OEnxxx-1 19200 256 2 axudp interface via ...	ax0 OEnxxx-1 19200 256 2 axudp interface via ...
ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreiber** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **das** Port gedacht ist.	ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreibe** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **der** Port gedacht ist.
Wird fortgesetzt (**by** OE1RSA**)**	**Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS** (**Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird.**

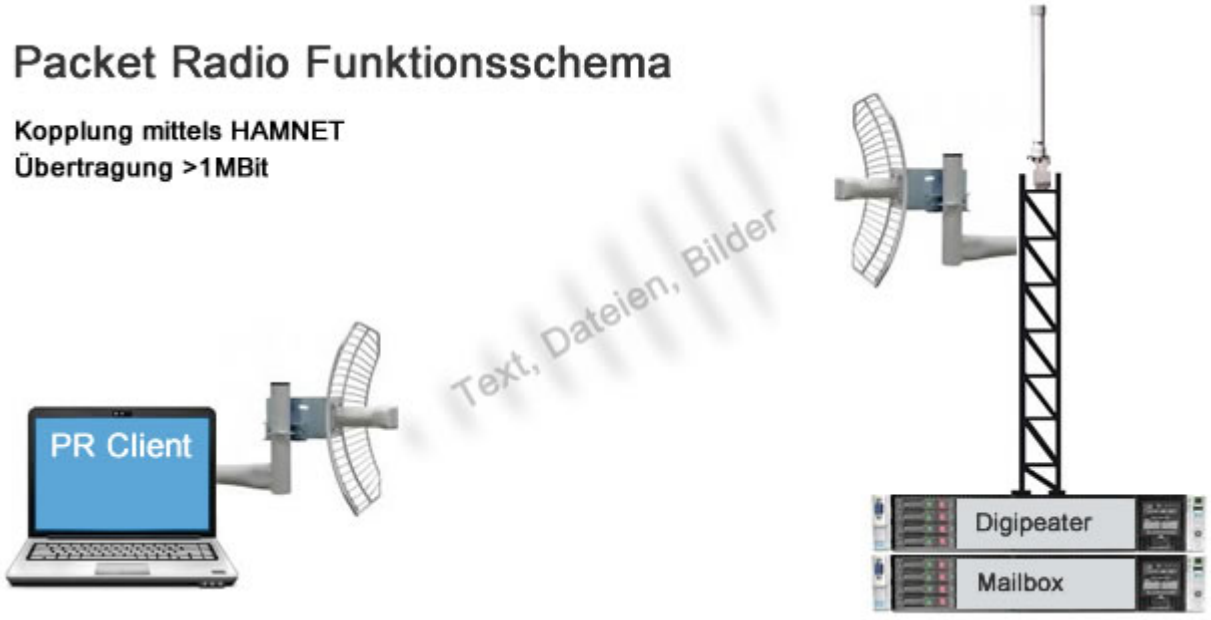
–	+ In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.
	+
	+ Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:
	+
	+ socket udp
	+ mode tnc
	+ mycall OEnxxx-1 # bitte einsetzten
	+ device /dev/ttyq0
	+ speed 9600
	+ loglevel 4
	+ broadcast NODES
	+ route OE1XAR 44.143.7.25 udp 10094 b
	+
	+ Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.
	+

+ **Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.**
+
+ **#! /bin/sh**
+
+ **case "$1" in**
+ **start)**
+ **# create pseudo tty devices:**
+ **socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &**
+ **socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &**
+ **sleep 3**
+
+ **/usr/sbin/kissattach -l /dev/ptyq0 ax0**
+ **/usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip**
+ **exit 0**
+ **;;**
+
+ **stop)**
+ **killall -TERM ax25ipd**
+ **killall -TERM kissattach**
+ **killall -TERM socat**
+ **exit 0**
+ **;;**

```
+
+   *)
+       echo "Usage: ax25 {start|stop}"
+       exit 0
+       ;;
+
+   esac
+
+   exit 0
+
+ Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit
+
+   sudo ax25 start
+
+ starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.
+
+ Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:
+
+   axcall ax0 OE1XAR
+
```

+	**Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel [http://xnet.swiss-artg.ch/pdf/xnet138.pdf hier] gefunden werden kann.**
+	
+	**Am Ende können wir bei Bedarf mit**
+	
+	**sudo ax25 stop**
+	
+	**die ax25 Umgebung wieder deaktivieren.**
+	
+	**Viel Erfolg wünscht Euch** OE1RSA.
== Beispiel Anleitungen ==	== Beispiel Anleitungen ==

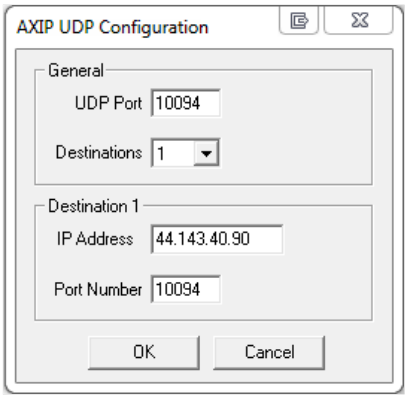
Version vom 2. April 2018, 16:36 Uhr



Inhaltsverzeichnis

PR-Zugang via HAMNET

Packet Radio kann nicht nur über die herkömmlichen 1200 bzw. 9600 Baud Zugänge oder via Internet gemacht werden. Auch im HAMNET - Highspeed Amateur Multimedia Network kann man sich Zugang zum Packet Radio Netzwerk verschaffen.



AXUDP für OE2XZR

Am OE2XZR Gaisberg bei Salzburg besteht für Benutzer bereits die Möglichkeit sich via 2,4GHz WLAN zum Accesspoint zu verbinden, und mit herkömmlicher Software wie Flexnet und Paxon Client PR Betrieb zu machen.

Lesen Sie dazu die Anleitung.

Das benötigte HF WLAN Equipment wird ebenfalls im Bereich Digitaler Backbone näher beschrieben.

AXUDP Zugangspunkte:

Station	IP	UDP Port
OE1XAR Bisamberg	44.143.7.25	10094
OE2XZR Gaisberg	44.143.40.90	10094
OE5XUL Ried/Geiersberg	44.143.105.158	10094

Transport von AX25 - Packet Radio im HAMNET

Unter Anwendung des OSI-Modells können AX.25 Datenpakete mittels AXUDP oder AX-over IP Paketen „per Rucksack“ im HAMNET transportiert bzw. eingebettet werden. Die Geschwindigkeit übertrifft dabei ein vielfaches der bestehenden 23cm 9k6 oder 19k2-FSK-Technik.

Die AX.25 Pakete können über Schnittstellen zu RMNC-Digipeatern (zb.: KISS-Karte) oder direkt an neueren Knotenrechnern (z.B: DLC7 mit XNET) in das HAMNET eingespeist und auf den Protokollschichten „huckepack“ genommen werden.

So können Linkstrecken zwischen Digipeatern auch über HAMNET-HF-Strecken zusammengeschaltet werden. Es ist auch möglich, als Funkamateur über einen HAMNET-HF-Userzugang in das Packet-Radio-Netz einzuloggen.

Eine bisher gebräuchliche Art des Huckepackverkehrs war der umgekehrte Fall, das sogenannte „IP over AX25“ oder oft auch „TCP/IP over AX“ genannt. Hierbei können über PR- Usereinstiege auch Webseiten oder andere IP-Dienste in z.T. langsamer Geschwindigkeit genutzt werden. Da AMPR einen TCPIP Stack über das AX25 Packetradio Netz benötigt, muss eine entsprechende Software wie Flexnet, AGW, WAMPES oder ax25-Linux vorhanden sein. Dabei ist der TCPIP-Stack für die jeweilige Anwendung transparent und es können diverse gewohnte Anwendungen verwendet werden.

In beiden Fällen ("IP over AX" für AMPR – sowie für das "AX over IP" im HAMNET) werden IP-Adressen benötigt.

Zugang von Linux aus

Linux erlaubt es praktisch mit "Bordmitteln" den Zugang einzurichten. Die Schwierigkeiten liegen dabei eher in der nicht immer leicht auffindbaren Dokumentation. Eine hilfreiche Quelle ist AXUDP-Gateways im Hamnet nutzen von DB0OVN. Ich habe diese Doku aber erst gefunden, nachdem ich es geschafft hatte die Verbindung herzustellen. Da ich eine alternative Methode verwendet habe beschreibe ich sie hier zusätzlich:

Das verwendete System ist Ubuntu 17.10. Ich setzte Kenntnisse im Umgang mit der Kommandozeile und im Erstellen von Shell-Skripten voraus.

Zunächst installiert man die Pakete ax25-apps und ax25-tools:

```
sudo apt-get install ax25-apps ax25-tools
```

In die Datei /etc/ax25/axports trägt man ein:

```
ax0 OEnxxx-1 19200 256 2 axudp interface via ...
```

ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder beschreibe ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür der Port gedacht ist.

Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS (Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.

Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:

```
socket udp
mode tnc
mycall OEnxxx-1 # bitte einsetzten
device /dev/ttyq0
speed 9600
loglevel 4
broadcast NODES
route OE1XAR 44.143.7.25 udp 10094 b
```

Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.

Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.

```
#! /bin/sh

case "$1" in
  start)
      # create pseudo tty devices:
      socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
      socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
      sleep 3

      /usr/sbin/kissattach -l /dev/ptyq0 ax0
      /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
      exit 0
      ;;

  stop)
      killall -TERM ax25ipd
      killall -TERM kissattach
      killall -TERM socat
      exit 0
      ;;

  *)
      echo "Usage: ax25 {start|stop}"
      exit 0
      ;;

esac

exit 0
```

Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit

```
sudo ax25 start
```

starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.

Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:

```
axcall ax0 OE1XAR
```

Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel hier gefunden werden kann.

Am Ende können wir bei Bedarf mit

```
sudo ax25 stop
```

die ax25 Umgebung wieder deaktivieren.

Viel Erfolg wünscht Euch OE1RSA.

Beispiel Anleitungen

- Packet Radio Zugang im HAMNET am OE2XZR Gaisberg
- Packet Radio via Mailclient Lesen und Antworten von Packet Radio Nachrichten via Mailclient (bspw. MS Outlook) im HAMNET am OE2XZR Gaisberg

Packet Radio via HAMNET: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 2. April 2018, 15:08 Uhr (Quelltext anzeigen) Oe1rsa (Diskussion | Beiträge) ← Zum vorherigen Versionsunterschied	**Version vom 2. April 2018, 16:36 Uhr (Quelltext anzeigen)** Oe1rsa (Diskussion | Beiträge) (→Zugang von Linux aus) Zum nächsten Versionsunterschied →
Zeile 59:	**Zeile 59:**
ax0 OEnxxx-1 19200 256 2 axudp interface via ...	ax0 OEnxxx-1 19200 256 2 axudp interface via ...
− ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreiber** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **das** Port gedacht ist.	+ ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreibe** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **der** Port gedacht ist.
Wird fortgesetzt (**by** OE1RSA**)**	**Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS** (**Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal**

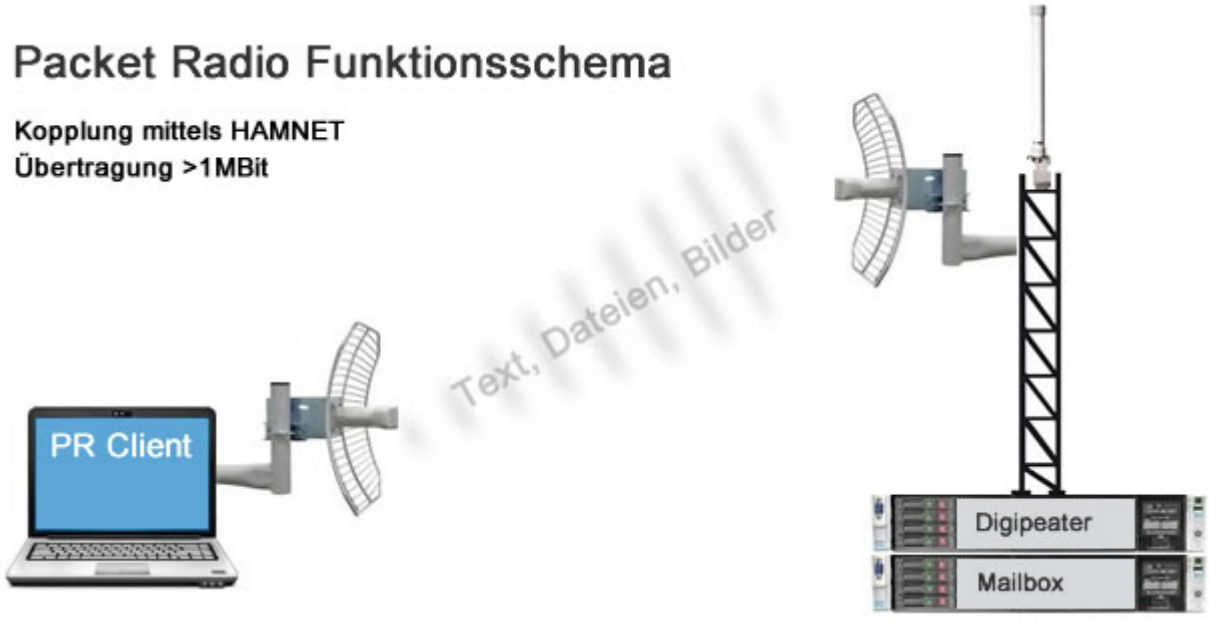
–	+ Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.
	+
	+ Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:
	+
	+ socket udp
	+ mode tnc
	+ mycall OEnxxx-1 # bitte einsetzten
	+ device /dev/ttyq0
	+ speed 9600
	+ loglevel 4
	+ broadcast NODES
	+ route OE1XAR 44.143.7.25 udp 10094 b
	+
	+ Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.
	+

+ Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.
+
+ #! /bin/sh
+
+ case "$1" in
+ start)
+ # create pseudo tty devices:
+ socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
+ socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
+ sleep 3
+
+ /usr/sbin/kissattach -l /dev/ptyq0 ax0
+ /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
+ exit 0
+ ;;
+
+ stop)
+ killall -TERM ax25ipd
+ killall -TERM kissattach
+ killall -TERM socat
+ exit 0
+ ;;

```
+
+   *)
+       echo "Usage: ax25 {start|stop}"
+       exit 0
+       ;;
+
+   esac
+
+   exit 0
+
+ Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit
+
+   sudo ax25 start
+
+ starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.
+
+ Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:
+
+   axcall ax0 OE1XAR
+
```

+	**Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel [http://xnet.swiss-artg.ch/pdf/xnet138.pdf hier] gefunden werden kann.**
+	
+	**Am Ende können wir bei Bedarf mit**
+	
+	**sudo ax25 stop**
+	
+	**die ax25 Umgebung wieder deaktivieren.**
+	
+	**Viel Erfolg wünscht Euch** OE1RSA.
== Beispiel Anleitungen ==	== Beispiel Anleitungen ==

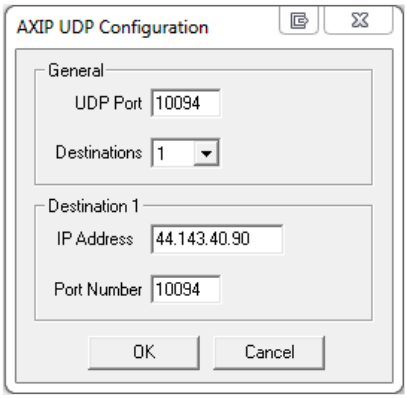
Version vom 2. April 2018, 16:36 Uhr



Inhaltsverzeichnis

PR-Zugang via HAMNET

Packet Radio kann nicht nur über die herkömmlichen 1200 bzw. 9600 Baud Zugänge oder via Internet gemacht werden. Auch im HAMNET - Highspeed Amateur Multimedia Network kann man sich Zugang zum Packet Radio Netzwerk verschaffen.



AXUDP für OE2XZR

Am OE2XZR Gaisberg bei Salzburg besteht für Benutzer bereits die Möglichkeit sich via 2,4GHz WLAN zum Accesspoint zu verbinden, und mit herkömmlicher Software wie Flexnet und Paxon Client PR Betrieb zu machen.

Lesen Sie dazu die Anleitung.

Das benötigte HF WLAN Equipment wird ebenfalls im Bereich Digitaler Backbone näher beschrieben.

AXUDP Zugangspunkte:

Station	IP	UDP Port
OE1XAR Bisamberg	44.143.7.25	10094
OE2XZR Gaisberg	44.143.40.90	10094
OE5XUL Ried/Geiersberg	44.143.105.158	10094

Transport von AX25 - Packet Radio im HAMNET

Unter Anwendung des OSI-Modells können AX.25 Datenpakete mittels AXUDP oder AX-over IP Paketen „per Rucksack“ im HAMNET transportiert bzw. eingebettet werden. Die Geschwindigkeit übertrifft dabei ein vielfaches der bestehenden 23cm 9k6 oder 19k2-FSK-Technik.

Die AX.25 Pakete können über Schnittstellen zu RMNC-Digipeatern (zb.: KISS-Karte) oder direkt an neueren Knotenrechnern (z.B: DLC7 mit XNET) in das HAMNET eingespeist und auf den Protokollschichten „huckepack“ genommen werden.

So können Linkstrecken zwischen Digipeatern auch über HAMNET-HF-Strecken zusammengeschaltet werden. Es ist auch möglich, als Funkamateur über einen HAMNET-HF-Userzugang in das Packet-Radio-Netz einzuloggen.

Eine bisher gebräuchliche Art des Huckepackverkehrs war der umgekehrte Fall, das sogenannte „IP over AX25“ oder oft auch „TCP/IP over AX“ genannt. Hierbei können über PR- Usereinstiege auch Webseiten oder andere IP-Dienste in z.T. langsamer Geschwindigkeit genutzt werden. Da AMPR einen TCPIP Stack über das AX25 Packetradio Netz benötigt, muss eine entsprechende Software wie Flexnet, AGW, WAMPES oder ax25-Linux vorhanden sein. Dabei ist der TCPIP-Stack für die jeweilige Anwendung transparent und es können diverse gewohnte Anwendungen verwendet werden.

In beiden Fällen ("IP over AX" für AMPR – sowie für das "AX over IP" im HAMNET) werden IP-Adressen benötigt.

Zugang von Linux aus

Linux erlaubt es praktisch mit "Bordmitteln" den Zugang einzurichten. Die Schwierigkeiten liegen dabei eher in der nicht immer leicht auffindbaren Dokumentation. Eine hilfreiche Quelle ist AXUDP-Gateways im Hamnet nutzen von DB0OVN. Ich habe diese Doku aber erst gefunden, nachdem ich es geschafft hatte die Verbindung herzustellen. Da ich eine alternative Methode verwendet habe beschreibe ich sie hier zusätzlich:

Das verwendete System ist Ubuntu 17.10. Ich setzte Kenntnisse im Umgang mit der Kommandozeile und im Erstellen von Shell-Skripten voraus.

Zunächst installiert man die Pakete ax25-apps und ax25-tools:

```
sudo apt-get install ax25-apps ax25-tools
```

In die Datei /etc/ax25/axports trägt man ein:

```
ax0 OEnxxx-1 19200 256 2 axudp interface via ...
```

ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder beschreibe ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür der Port gedacht ist.

Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS (Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.

Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:

```
socket udp
mode tnc
mycall OEnxxx-1 # bitte einsetzten
device /dev/ttyq0
speed 9600
loglevel 4
broadcast NODES
route OE1XAR 44.143.7.25 udp 10094 b
```

Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.

Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.

```
#! /bin/sh

case "$1" in
  start)
      # create pseudo tty devices:
      socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
      socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
      sleep 3

      /usr/sbin/kissattach -l /dev/ptyq0 ax0
      /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
      exit 0
      ;;

  stop)
      killall -TERM ax25ipd
      killall -TERM kissattach
      killall -TERM socat
      exit 0
      ;;

  *)
      echo "Usage: ax25 {start|stop}"
      exit 0
      ;;

esac

exit 0
```

Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit

```
sudo ax25 start
```

starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.

Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:

```
axcall ax0 OE1XAR
```

Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel hier gefunden werden kann.

Am Ende können wir bei Bedarf mit

```
sudo ax25 stop
```

die ax25 Umgebung wieder deaktivieren.

Viel Erfolg wünscht Euch OE1RSA.

Beispiel Anleitungen

- Packet Radio Zugang im HAMNET am OE2XZR Gaisberg
- Packet Radio via Mailclient Lesen und Antworten von Packet Radio Nachrichten via Mailclient (bspw. MS Outlook) im HAMNET am OE2XZR Gaisberg

Packet Radio via HAMNET: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 2. April 2018, 15:08 Uhr (Quelltext anzeigen) Oe1rsa (Diskussion | Beiträge) ← Zum vorherigen Versionsunterschied	**Version vom 2. April 2018, 16:36 Uhr (Quelltext anzeigen)** Oe1rsa (Diskussion | Beiträge) (→Zugang von Linux aus) Zum nächsten Versionsunterschied →
Zeile 59:	**Zeile 59:**
ax0 OEnxxx-1 19200 256 2 axudp interface via ...	ax0 OEnxxx-1 19200 256 2 axudp interface via ...
− ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreiber** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **das** Port gedacht ist.	+ ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreibe** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **der** Port gedacht ist.
Wird fortgesetzt (**by** OE1RSA**)**	**Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS** (**Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal**

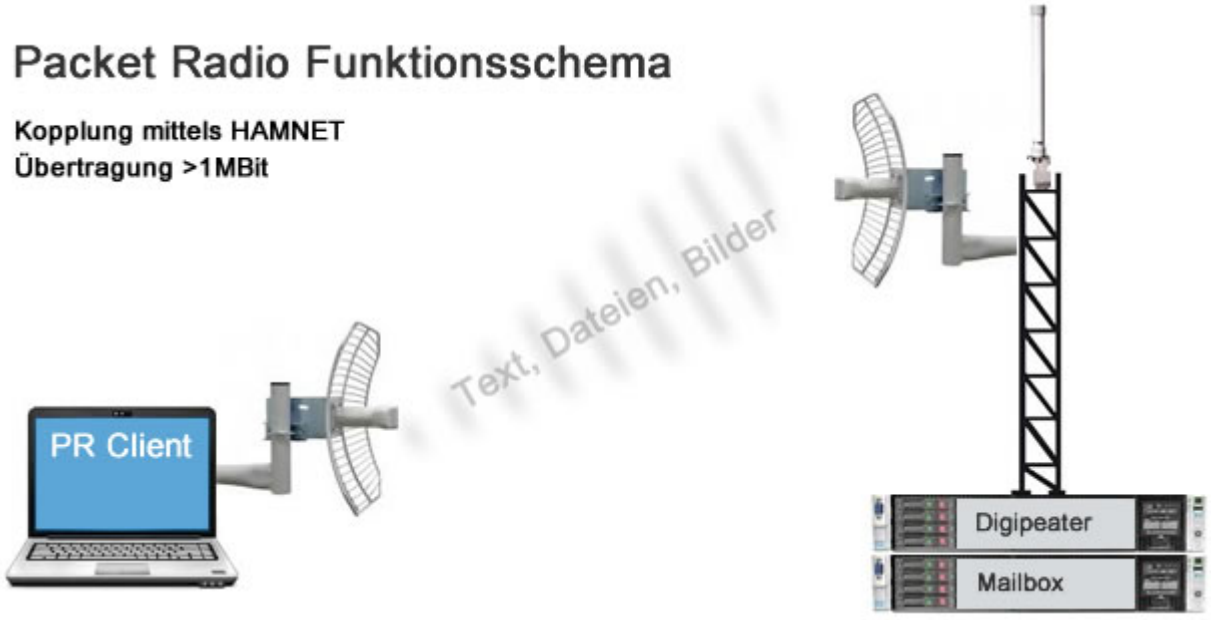
–	+ Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.
	+
	+ Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:
	+
	+ socket udp
	+ mode tnc
	+ mycall OEnxxx-1 # bitte einsetzten
	+ device /dev/ttyq0
	+ speed 9600
	+ loglevel 4
	+ broadcast NODES
	+ route OE1XAR 44.143.7.25 udp 10094 b
	+
	+ Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.
	+

+ **Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.**
+
+ **#! /bin/sh**
+
+ **case "$1" in**
+ **start)**
+ **# create pseudo tty devices:**
+ **socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &**
+ **socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &**
+ **sleep 3**
+
+ **/usr/sbin/kissattach -l /dev/ptyq0 ax0**
+ **/usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip**
+ **exit 0**
+ **;;**
+
+ **stop)**
+ **killall -TERM ax25ipd**
+ **killall -TERM kissattach**
+ **killall -TERM socat**
+ **exit 0**
+ **;;**

```
+
+   *)
+       echo "Usage: ax25 {start|stop}"
+       exit 0
+       ;;
+
+   esac
+
+   exit 0
+
+ Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit
+
+   sudo ax25 start
+
+ starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.
+
+ Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:
+
+   axcall ax0 OE1XAR
+
```

	+ **Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel [http://xnet.swiss-artg.ch/pdf/xnet138.pdf hier] gefunden werden kann.**
	+
	+ **Am Ende können wir bei Bedarf mit**
	+
	+ **sudo ax25 stop**
	+
	+ **die ax25 Umgebung wieder deaktivieren.**
	+
	+ **Viel Erfolg wünscht Euch** OE1RSA.
== Beispiel Anleitungen ==	== Beispiel Anleitungen ==

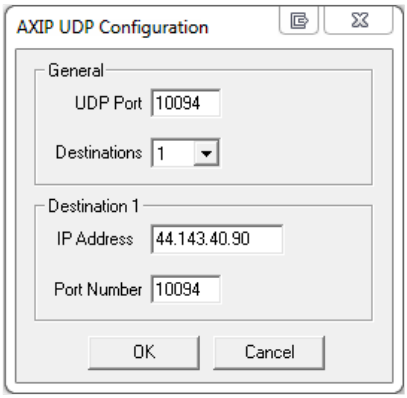
Version vom 2. April 2018, 16:36 Uhr



Inhaltsverzeichnis

PR-Zugang via HAMNET

Packet Radio kann nicht nur über die herkömmlichen 1200 bzw. 9600 Baud Zugänge oder via Internet gemacht werden. Auch im HAMNET - Highspeed Amateur Multimedia Network kann man sich Zugang zum Packet Radio Netzwerk verschaffen.



AXUDP für OE2XZR

Am OE2XZR Gaisberg bei Salzburg besteht für Benutzer bereits die Möglichkeit sich via 2,4GHz WLAN zum Accesspoint zu verbinden, und mit herkömmlicher Software wie Flexnet und Paxon Client PR Betrieb zu machen.

Lesen Sie dazu die Anleitung.

Das benötigte HF WLAN Equipment wird ebenfalls im Bereich Digitaler Backbone näher beschrieben.

AXUDP Zugangspunkte:

Station	IP	UDP Port
OE1XAR Bisamberg	44.143.7.25	10094
OE2XZR Gaisberg	44.143.40.90	10094
OE5XUL Ried/Geiersberg	44.143.105.158	10094

Transport von AX25 - Packet Radio im HAMNET

Unter Anwendung des OSI-Modells können AX.25 Datenpakete mittels AXUDP oder AX-over IP Paketen „per Rucksack“ im HAMNET transportiert bzw. eingebettet werden. Die Geschwindigkeit übertrifft dabei ein vielfaches der bestehenden 23cm 9k6 oder 19k2-FSK-Technik.

Die AX.25 Pakete können über Schnittstellen zu RMNC-Digipeatern (zb.: KISS-Karte) oder direkt an neueren Knotenrechnern (z.B: DLC7 mit XNET) in das HAMNET eingespeist und auf den Protokollschichten „huckepack“ genommen werden.

So können Linkstrecken zwischen Digipeatern auch über HAMNET-HF-Strecken zusammengeschaltet werden. Es ist auch möglich, als Funkamateur über einen HAMNET-HF-Userzugang in das Packet-Radio-Netz einzuloggen.

Eine bisher gebräuchliche Art des Huckepackverkehrs war der umgekehrte Fall, das sogenannte „IP over AX25“ oder oft auch „TCP/IP over AX“ genannt. Hierbei können über PR- Usereinstiege auch Webseiten oder andere IP-Dienste in z.T. langsamer Geschwindigkeit genutzt werden. Da AMPR einen TCPIP Stack über das AX25 Packetradio Netz benötigt, muss eine entsprechende Software wie Flexnet, AGW, WAMPES oder ax25-Linux vorhanden sein. Dabei ist der TCPIP-Stack für die jeweilige Anwendung transparent und es können diverse gewohnte Anwendungen verwendet werden.

In beiden Fällen ("IP over AX" für AMPR – sowie für das "AX over IP" im HAMNET) werden IP-Adressen benötigt.

Zugang von Linux aus

Linux erlaubt es praktisch mit "Bordmitteln" den Zugang einzurichten. Die Schwierigkeiten liegen dabei eher in der nicht immer leicht auffindbaren Dokumentation. Eine hilfreiche Quelle ist AXUDP-Gateways im Hamnet nutzen von DB0OVN. Ich habe diese Doku aber erst gefunden, nachdem ich es geschafft hatte die Verbindung herzustellen. Da ich eine alternative Methode verwendet habe beschreibe ich sie hier zusätzlich:

Das verwendete System ist Ubuntu 17.10. Ich setzte Kenntnisse im Umgang mit der Kommandozeile und im Erstellen von Shell-Skripten voraus.

Zunächst installiert man die Pakete ax25-apps und ax25-tools:

```
sudo apt-get install ax25-apps ax25-tools
```

In die Datei /etc/ax25/axports trägt man ein:

```
ax0 OEnxxx-1 19200 256 2 axudp interface via ...
```

ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder beschreibe ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür der Port gedacht ist.

Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS (Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.

Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:

```
socket udp
mode tnc
mycall OEnxxx-1 # bitte einsetzten
device /dev/ttyq0
speed 9600
loglevel 4
broadcast NODES
route OE1XAR 44.143.7.25 udp 10094 b
```

Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.

Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.

```
#! /bin/sh

case "$1" in
  start)
      # create pseudo tty devices:
      socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
      socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
      sleep 3

      /usr/sbin/kissattach -l /dev/ptyq0 ax0
      /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
      exit 0
      ;;

  stop)
      killall -TERM ax25ipd
      killall -TERM kissattach
      killall -TERM socat
      exit 0
      ;;

  *)
      echo "Usage: ax25 {start|stop}"
      exit 0
      ;;

esac

exit 0
```

Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit

```
sudo ax25 start
```

starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.

Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:

```
axcall ax0 OE1XAR
```

Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel hier gefunden werden kann.

Am Ende können wir bei Bedarf mit

```
sudo ax25 stop
```

die ax25 Umgebung wieder deaktivieren.

Viel Erfolg wünscht Euch OE1RSA.

Beispiel Anleitungen

- Packet Radio Zugang im HAMNET am OE2XZR Gaisberg
- Packet Radio via Mailclient Lesen und Antworten von Packet Radio Nachrichten via Mailclient (bspw. MS Outlook) im HAMNET am OE2XZR Gaisberg

Seiten in der Kategorie „Digitaler Backbone“

Folgende 45 Seiten sind in dieser Kategorie, von 45 insgesamt.

7

- 70cm Datentransceiver für HAMNET

A

- Adressierung in OE
- Anwendungen am HAMNET
- Arbeitsgruppe OE1
- Arbeitsgruppe OE3
- Arbeitsgruppe OE4 OE6 OE8
- Arbeitsgruppe OE5
- Arbeitsgruppe OE7
- Arbeitsgruppe OE9

B

- Backbone
- Bandbreiten digitaler Backbone
- BigBlueButtonServer

D

- D4C - Digital4Capitals
- Dokumentationen
- Domain Name System
- DXL - APRSmap

E

- Einstellungen Digitaler Backbone
- Email im digitalen Netz

F

- Frequenzen Digitaler Backbone

H

- HAMNET HOC
- HAMNET Service Provider
- HAMNET Vorträge
- HAMNET-70

L

- Linkberechnung
- Linkkomponenten digitaler Backbone
- Links
- Linkstart - Konfiguration vor dem Aufbau
- Livestream

R

- Routing - AS-Nummern
- Routing digitaler Backbone

S

- SAMNET

T

- TCE Tinycore Linux Projekt
- Teststellungen Gaisberg Gernkogel
- Teststellungen OE5

U

- Userequipment HAMNETmesh
- Userequipment HAMNETpoweruser
- Userzugang-HAMNET

V

- VoIP - HAMSIP
- VoIP Codec Uebersicht
- VoIP Einstellungen

- VoIP Rufnummernplan am HAMNET

W

- WXNET-ESP

X

- X ARCHIV IP Adressen OE
- X ARCHIV Koordinaten
- X ARCHIV Messungen digitaler Backbone

Packet Radio via HAMNET: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 2. April 2018, 15:08 Uhr (Quelltext anzeigen) Oe1rsa (Diskussion | Beiträge) ← Zum vorherigen Versionsunterschied	**Version vom 2. April 2018, 16:36 Uhr (Quelltext anzeigen)** Oe1rsa (Diskussion | Beiträge) (→Zugang von Linux aus) Zum nächsten Versionsunterschied →
Zeile 59:	**Zeile 59:**
ax0 OEnxxx-1 19200 256 2 axudp interface via ...	ax0 OEnxxx-1 19200 256 2 axudp interface via ...
− ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreiber** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **das** Port gedacht ist.	+ ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreibe** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **der** Port gedacht ist.
Wird fortgesetzt (**by** OE1RSA**)**	**Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS** (**Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal**

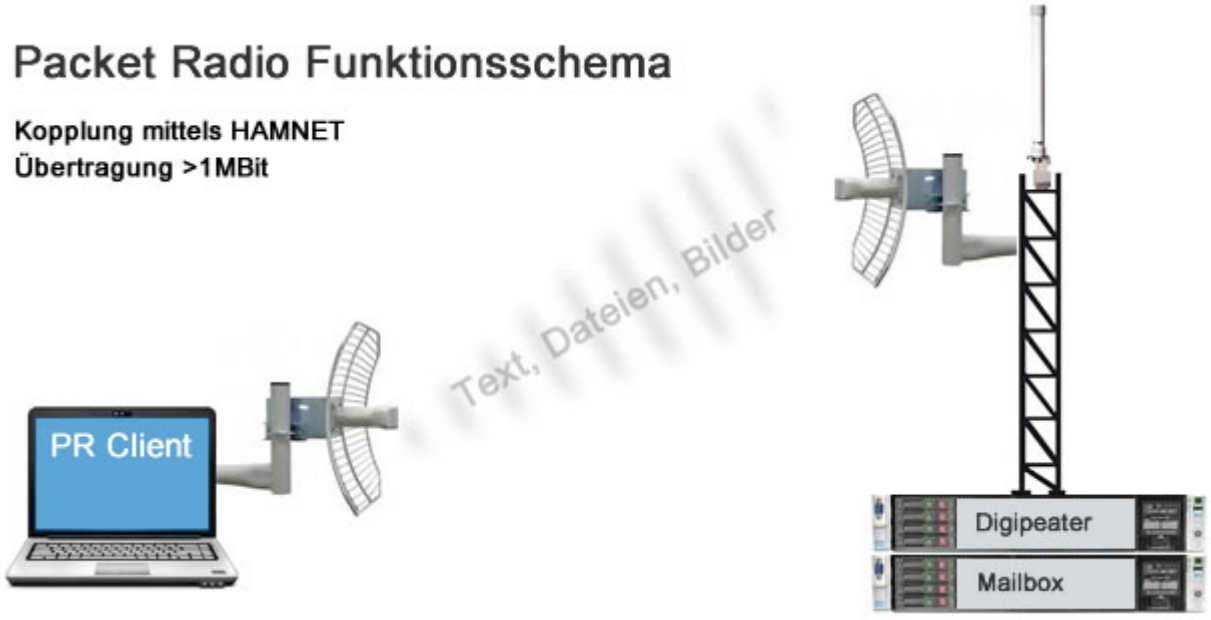
–	+ Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.
	+
	+ Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:
	+
	+ socket udp
	+ mode tnc
	+ mycall OEnxxx-1 # bitte einsetzten
	+ device /dev/ttyq0
	+ speed 9600
	+ loglevel 4
	+ broadcast NODES
	+ route OE1XAR 44.143.7.25 udp 10094 b
	+
	+ Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.
	+

+ **Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.**
+
+ **#! /bin/sh**
+
+ **case "$1" in**
+ **start)**
+ **# create pseudo tty devices:**
+ **socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &**
+ **socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &**
+ **sleep 3**
+
+ **/usr/sbin/kissattach -l /dev/ptyq0 ax0**
+ **/usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip**
+ **exit 0**
+ **;;**
+
+ **stop)**
+ **killall -TERM ax25ipd**
+ **killall -TERM kissattach**
+ **killall -TERM socat**
+ **exit 0**
+ **;;**

```
+
+   *)
+       echo "Usage: ax25 {start|stop}"
+       exit 0
+       ;;
+
+   esac
+
+   exit 0
+
+ Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit
+
+   sudo ax25 start
+
+ starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.
+
+ Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:
+
+   axcall ax0 OE1XAR
+
```

	+ **Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel [http://xnet.swiss-artg.ch/pdf/xnet138.pdf hier] gefunden werden kann.**
	+
	+ **Am Ende können wir bei Bedarf mit**
	+
	+ **sudo ax25 stop**
	+
	+ **die ax25 Umgebung wieder deaktivieren.**
	+
	+ **Viel Erfolg wünscht Euch** OE1RSA.
== Beispiel Anleitungen ==	== Beispiel Anleitungen ==

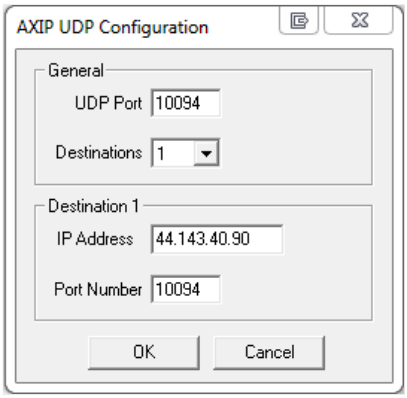
Version vom 2. April 2018, 16:36 Uhr



Inhaltsverzeichnis

PR-Zugang via HAMNET

Packet Radio kann nicht nur über die herkömmlichen 1200 bzw. 9600 Baud Zugänge oder via Internet gemacht werden. Auch im HAMNET - Highspeed Amateur Multimedia Network kann man sich Zugang zum Packet Radio Netzwerk verschaffen.



AXUDP für OE2XZR

Am OE2XZR Gaisberg bei Salzburg besteht für Benutzer bereits die Möglichkeit sich via 2,4GHz WLAN zum Accesspoint zu verbinden, und mit herkömmlicher Software wie Flexnet und Paxon Client PR Betrieb zu machen.

Lesen Sie dazu die Anleitung.

Das benötigte HF WLAN Equipment wird ebenfalls im Bereich Digitaler Backbone näher beschrieben.

AXUDP Zugangspunkte:

Station	IP	UDP Port
OE1XAR Bisamberg	44.143.7.25	10094
OE2XZR Gaisberg	44.143.40.90	10094
OE5XUL Ried/Geiersberg	44.143.105.158	10094

Transport von AX25 - Packet Radio im HAMNET

Unter Anwendung des OSI-Modells können AX.25 Datenpakete mittels AXUDP oder AX-over IP Paketen „per Rucksack“ im HAMNET transportiert bzw. eingebettet werden. Die Geschwindigkeit übertrifft dabei ein vielfaches der bestehenden 23cm 9k6 oder 19k2-FSK-Technik.

Die AX.25 Pakete können über Schnittstellen zu RMNC-Digipeatern (zb.: KISS-Karte) oder direkt an neueren Knotenrechnern (z.B: DLC7 mit XNET) in das HAMNET eingespeist und auf den Protokollschichten „huckepack“ genommen werden.

So können Linkstrecken zwischen Digipeatern auch über HAMNET-HF-Strecken zusammengeschaltet werden. Es ist auch möglich, als Funkamateur über einen HAMNET-HF-Userzugang in das Packet-Radio-Netz einzuloggen.

Eine bisher gebräuchliche Art des Huckepackverkehrs war der umgekehrte Fall, das sogenannte „IP over AX25“ oder oft auch „TCP/IP over AX“ genannt. Hierbei können über PR- Usereinstiege auch Webseiten oder andere IP-Dienste in z.T. langsamer Geschwindigkeit genutzt werden. Da AMPR einen TCPIP Stack über das AX25 Packetradio Netz benötigt, muss eine entsprechende Software wie Flexnet, AGW, WAMPES oder ax25-Linux vorhanden sein. Dabei ist der TCPIP-Stack für die jeweilige Anwendung transparent und es können diverse gewohnte Anwendungen verwendet werden.

In beiden Fällen ("IP over AX" für AMPR – sowie für das "AX over IP" im HAMNET) werden IP-Adressen benötigt.

Zugang von Linux aus

Linux erlaubt es praktisch mit "Bordmitteln" den Zugang einzurichten. Die Schwierigkeiten liegen dabei eher in der nicht immer leicht auffindbaren Dokumentation. Eine hilfreiche Quelle ist AXUDP-Gateways im Hamnet nutzen von DB0OVN. Ich habe diese Doku aber erst gefunden, nachdem ich es geschafft hatte die Verbindung herzustellen. Da ich eine alternative Methode verwendet habe beschreibe ich sie hier zusätzlich:

Das verwendete System ist Ubuntu 17.10. Ich setzte Kenntnisse im Umgang mit der Kommandozeile und im Erstellen von Shell-Skripten voraus.

Zunächst installiert man die Pakete ax25-apps und ax25-tools:

```
sudo apt-get install ax25-apps ax25-tools
```

In die Datei /etc/ax25/axports trägt man ein:

```
ax0 OEnxxx-1 19200 256 2 axudp interface via ...
```

ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder beschreibe ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür der Port gedacht ist.

Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS (Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.

Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:

```
socket udp
mode tnc
mycall OEnxxx-1 # bitte einsetzten
device /dev/ttyq0
speed 9600
loglevel 4
broadcast NODES
route OE1XAR 44.143.7.25 udp 10094 b
```

Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.

Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.

```
#! /bin/sh

case "$1" in
  start)
      # create pseudo tty devices:
      socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
      socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
      sleep 3

      /usr/sbin/kissattach -l /dev/ptyq0 ax0
      /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
      exit 0
      ;;

  stop)
      killall -TERM ax25ipd
      killall -TERM kissattach
      killall -TERM socat
      exit 0
      ;;

  *)
      echo "Usage: ax25 {start|stop}"
      exit 0
      ;;

esac

exit 0
```

Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit

```
sudo ax25 start
```

starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.

Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:

```
axcall ax0 OE1XAR
```

Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel hier gefunden werden kann.

Am Ende können wir bei Bedarf mit

```
sudo ax25 stop
```

die ax25 Umgebung wieder deaktivieren.

Viel Erfolg wünscht Euch OE1RSA.

Beispiel Anleitungen

- Packet Radio Zugang im HAMNET am OE2XZR Gaisberg
- Packet Radio via Mailclient Lesen und Antworten von Packet Radio Nachrichten via Mailclient (bspw. MS Outlook) im HAMNET am OE2XZR Gaisberg

Packet Radio via HAMNET: Unterschied zwischen den Versionen

Versionsgeschichte interaktiv durchsuchen
VisuellWikitext

Version vom 2. April 2018, 15:08 Uhr (Quelltext anzeigen) Oe1rsa (Diskussion | Beiträge) ← Zum vorherigen Versionsunterschied	**Version vom 2. April 2018, 16:36 Uhr (Quelltext anzeigen)** Oe1rsa (Diskussion | Beiträge) (→Zugang von Linux aus) Zum nächsten Versionsunterschied →
Zeile 59:	**Zeile 59:**
ax0 OEnxxx-1 19200 256 2 axudp interface via ...	ax0 OEnxxx-1 19200 256 2 axudp interface via ...
− ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreiber** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **das** Port gedacht ist.	+ ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder **beschreibe** ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür **der** Port gedacht ist.
Wird fortgesetzt (**by** OE1RSA**)**	**Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS** (**Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal**

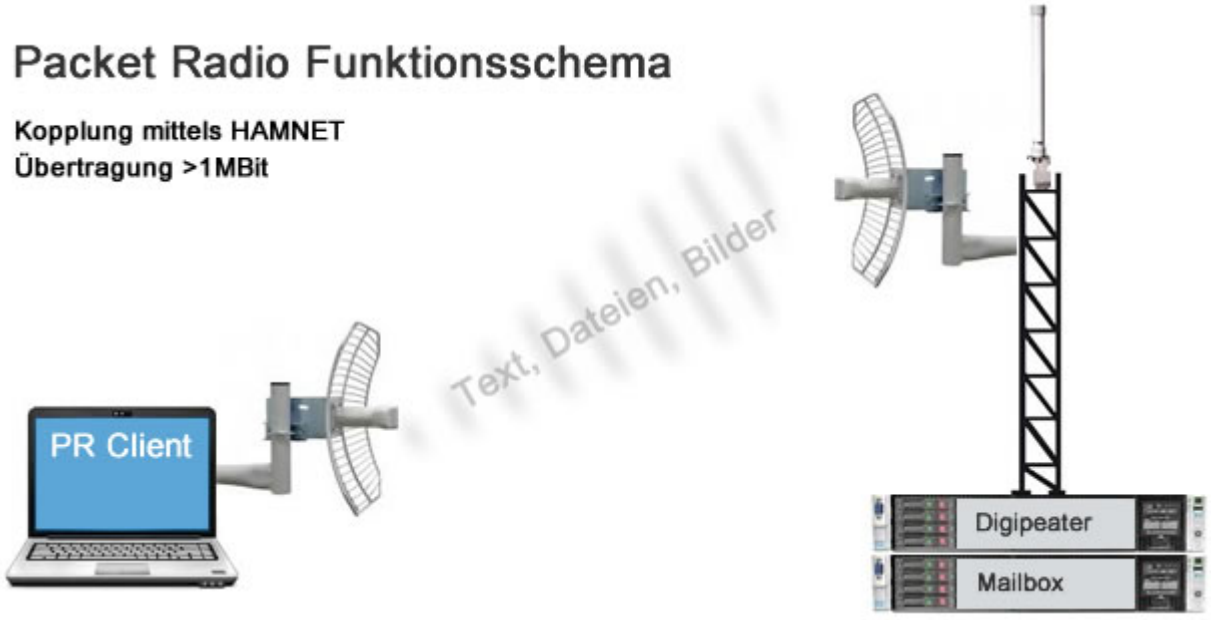
–	+ Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.
	+
	+ Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:
	+
	+ socket udp
	+ mode tnc
	+ mycall OEnxxx-1 # bitte einsetzten
	+ device /dev/ttyq0
	+ speed 9600
	+ loglevel 4
	+ broadcast NODES
	+ route OE1XAR 44.143.7.25 udp 10094 b
	+
	+ Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.
	+

+ **Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.**
+
+ **#! /bin/sh**
+
+ **case "$1" in**
+ **start)**
+ **# create pseudo tty devices:**
+ **socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &**
+ **socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &**
+ **sleep 3**
+
+ **/usr/sbin/kissattach -l /dev/ptyq0 ax0**
+ **/usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip**
+ **exit 0**
+ **;;**
+
+ **stop)**
+ **killall -TERM ax25ipd**
+ **killall -TERM kissattach**
+ **killall -TERM socat**
+ **exit 0**
+ **;;**

+
+ *)
+ echo "Usage: ax25 {start|stop}"
+ exit 0
+ ;;
+
+ esac
+
+ exit 0
+
+ Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit
+
+ sudo ax25 start
+
+ starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.
+
+ Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:
+
+ axcall ax0 OE1XAR
+

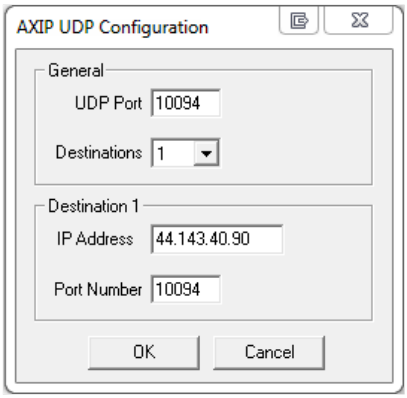
+	**Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel [http://xnet.swiss-artg.ch/pdf/xnet138.pdf hier] gefunden werden kann.**
+	
+	**Am Ende können wir bei Bedarf mit**
+	
+	**sudo ax25 stop**
+	
+	**die ax25 Umgebung wieder deaktivieren.**
+	
+	**Viel Erfolg wünscht Euch** OE1RSA.
== Beispiel Anleitungen ==	== Beispiel Anleitungen ==

Version vom 2. April 2018, 16:36 Uhr



Inhaltsverzeichnis

PR-Zugang via HAMNET



AXUDP für OE2XZR

Packet Radio kann nicht nur über die herkömmlichen 1200 bzw. 9600 Baud Zugänge oder via Internet gemacht werden. Auch im HAMNET - Highspeed Amateur Multimedia Network kann man sich Zugang zum Packet Radio Netzwerk verschaffen.

Am OE2XZR Gaisberg bei Salzburg besteht für Benutzer bereits die Möglichkeit sich via 2,4GHz WLAN zum Accesspoint zu verbinden, und mit herkömmlicher Software wie Flexnet und Paxon Client PR Betrieb zu machen.

Lesen Sie dazu die Anleitung.

Das benötigte HF WLAN Equipment wird ebenfalls im Bereich Digitaler Backbone näher beschrieben.

AXUDP Zugangspunkte:

Station	IP	UDP Port
OE1XAR Bisamberg	44.143.7.25	10094
OE2XZR Gaisberg	44.143.40.90	10094
OE5XUL Ried/Geiersberg	44.143.105.158	10094

Transport von AX25 - Packet Radio im HAMNET

Unter Anwendung des OSI-Modells können AX.25 Datenpakete mittels AXUDP oder AX-over IP Paketen „per Rucksack“ im HAMNET transportiert bzw. eingebettet werden. Die Geschwindigkeit übertrifft dabei ein vielfaches der bestehenden 23cm 9k6 oder 19k2-FSK-Technik.

Die AX.25 Pakete können über Schnittstellen zu RMNC-Digipeatern (zb.: KISS-Karte) oder direkt an neueren Knotenrechnern (z.B: DLC7 mit XNET) in das HAMNET eingespeist und auf den Protokollschichten „huckepack“ genommen werden.

So können Linkstrecken zwischen Digipeatern auch über HAMNET-HF-Strecken zusammengeschaltet werden. Es ist auch möglich, als Funkamateur über einen HAMNET-HF-Userzugang in das Packet-Radio-Netz einzuloggen.

Eine bisher gebräuchliche Art des Huckepackverkehrs war der umgekehrte Fall, das sogenannte „IP over AX25“ oder oft auch „TCP/IP over AX“ genannt. Hierbei können über PR- Usereinstiege auch Webseiten oder andere IP-Dienste in z.T. langsamer Geschwindigkeit genutzt werden. Da AMPR einen TCPIP Stack über das AX25 Packetradio Netz benötigt, muss eine entsprechende Software wie Flexnet, AGW, WAMPES oder ax25-Linux vorhanden sein. Dabei ist der TCPIP-Stack für die jeweilige Anwendung transparent und es können diverse gewohnte Anwendungen verwendet werden.

In beiden Fällen ("IP over AX" für AMPR – sowie für das "AX over IP" im HAMNET) werden IP-Adressen benötigt.

Zugang von Linux aus

Linux erlaubt es praktisch mit "Bordmitteln" den Zugang einzurichten. Die Schwierigkeiten liegen dabei eher in der nicht immer leicht auffindbaren Dokumentation. Eine hilfreiche Quelle ist AXUDP-Gateways im Hamnet nutzen von DB0OVN. Ich habe diese Doku aber erst gefunden, nachdem ich es geschafft hatte die Verbindung herzustellen. Da ich eine alternative Methode verwendet habe beschreibe ich sie hier zusätzlich:

Das verwendete System ist Ubuntu 17.10. Ich setzte Kenntnisse im Umgang mit der Kommandozeile und im Erstellen von Shell-Skripten voraus.

Zunächst installiert man die Pakete ax25-apps und ax25-tools:

```
sudo apt-get install ax25-apps ax25-tools
```

In die Datei /etc/ax25/axports trägt man ein:

```
ax0 OEnxxx-1 19200 256 2 axudp interface via ...
```

ax0 ist eine im Prinzip beliebige Bezeichnung für das Interface, vergleichbar mit der von der Ethernet Schnittstelle her bekannten Bezeichnung "eth0", hier aber eben für ein ax25 Port. Der nächste Eintrag spielt die Rolle der MAC Adresse, im Fall von AX25 muss hier das eigene Rufzeichen stehen. Da man nur ein Rufzeichen hat, aber durchaus mehrere "MAC Adressen" benötigt, kann man das Rufzeichen durch eine SSID (Secondary Station ID) nach einem Bindestrich ergänzen. Das nächste Feld, die Baudrate ist in unserem Fall nicht so wichtig, spielt aber eine Rolle wenn man statt des Umleitungsdaemon eine echte serielle Schnittstelle zu einem TNC anschließen möchte. Die nächsten beiden Felder beschreibe ich hier nicht, die übernehmen wir fürs Erste mal so. Am Ende kann dann noch ein Kommentar stehen wofür der Port gedacht ist.

Mit Hilfe des Programmes kissattach wird nun der eben parametrierte Port normalerweise mit einer seriellen Schnittstelle verbunden. KISS (Keep It Simple Stupid) ist dabei das Protokoll mit dem der TNC (Terminal Node Controller) angesprochen wird. In unserem Fall haben wir aber keinen echten TNC sondern verwenden ein weiteres Programm mit dem Namen ax25ipd, das einen TNC simuliert und die angebotenenen Datenpakete an eine IP Adresse weiterleitet. Eine kleine Hürde ist nun die Tatsache, dass wir keine echte serielle Schnittstelle verwenden wollen. In Linux ist dieses Problem recht einfach und elegant zu lösen: Wir verwenden ein virtuelles Terminal.

Doch bevor wir uns dem Thema mit der seriellen Schnittstelle zuwenden, legen wir die Parameterdatei für den Umpleitungdaemon ax25ipd an: Wir legen eine Datei mit dem Namen /etc /ax25/ax25udp.conf an mit folgemndem Inhalt:

```
socket udp
mode tnc
mycall OEnxxx-1 # bitte einsetzten
device /dev/ttyq0
speed 9600
loglevel 4
broadcast NODES
route OE1XAR 44.143.7.25 udp 10094 b
```

Natürlich können auch noch die anderen Routen wie weiter oben auf dieser Seite eingetragen werden. Hinter udp steht dabei die Portnummer.

Was nun noch fehlt ist ein kleines Skript, das die Programme startet und später, wenn wir sie nicht mehr benötigen auch wieder stoppt. Dazu legen wir die Datei ax25 an und markieren sie als ausführbare Datei. Auf meinem Laptop habe ich sie ins Verzeichnis /usr/local/bin kopiert damit sie von überallher aufrufbar ist.

```
#! /bin/sh

case "$1" in
  start)
      # create pseudo tty devices:
      socat PIPE:/dev/ttyq0 PIPE:/dev/ptyq0 &
      socat PTY,link=/dev/ttyq0 PTY,link=/dev/ptyq0 &
      sleep 3

      /usr/sbin/kissattach -l /dev/ptyq0 ax0
      /usr/sbin/ax25ipd -d /dev/ttyq0 -c /etc/ax25/ax25udp.conf > /tmp/axip
      exit 0
      ;;

  stop)
      killall -TERM ax25ipd
      killall -TERM kissattach
      killall -TERM socat
      exit 0
      ;;

  *)
      echo "Usage: ax25 {start|stop}"
      exit 0
      ;;

esac

exit 0
```

Das Programm socat auch das "Schweizer Messer" fürs Netzwerk genannt stellt uns dabei die Simulation der seriellen Schnittstelle her. Nun ist es fast geschafft. Mit

```
sudo ax25 start
```

starten wir die Programme. Wir müssen an dieser Stelle sudo verwenden, da wir root Rechte benötigen um die Netzwerktreiber neu zu konfigurieren und die seriellen Schnittstellen zu emulieren. Wer will kann das natürlich auch beim Hochfahren seines Systems automatisch ausführen lassen.

Nun können wir schon calls absetzten. Im einfachsten Fall benutzen wir das bei den ax25-apps vorhandene Tool axcall:

```
axcall ax0 OE1XAR
```

Nach kurzer Pause finden wir uns im "Split-Screen Terminal" verbunden mit dem Packet Knoten OE1XAR. Es handelt sich dabei um einen auf der Software (X)NET basierenden Digipeater (leider keine freie Software) dessen Bedienungsanleitung zum Beispiel hier gefunden werden kann.

Am Ende können wir bei Bedarf mit

```
sudo ax25 stop
```

die ax25 Umgebung wieder deaktivieren.

Viel Erfolg wünscht Euch OE1RSA.

Beispiel Anleitungen

- Packet Radio Zugang im HAMNET am OE2XZR Gaisberg
- Packet Radio via Mailclient Lesen und Antworten von Packet Radio Nachrichten via Mailclient (bspw. MS Outlook) im HAMNET am OE2XZR Gaisberg