Trust in Waves

An introduction to packet radio with AX.25 and elliptic curve cryptography

https://brannon.online/wopr.pdf

Data over Radio

Most people associate “radio” with voice transmissions, but from the beginning, radio has always been about transmitting arbitrary data from one place to another.

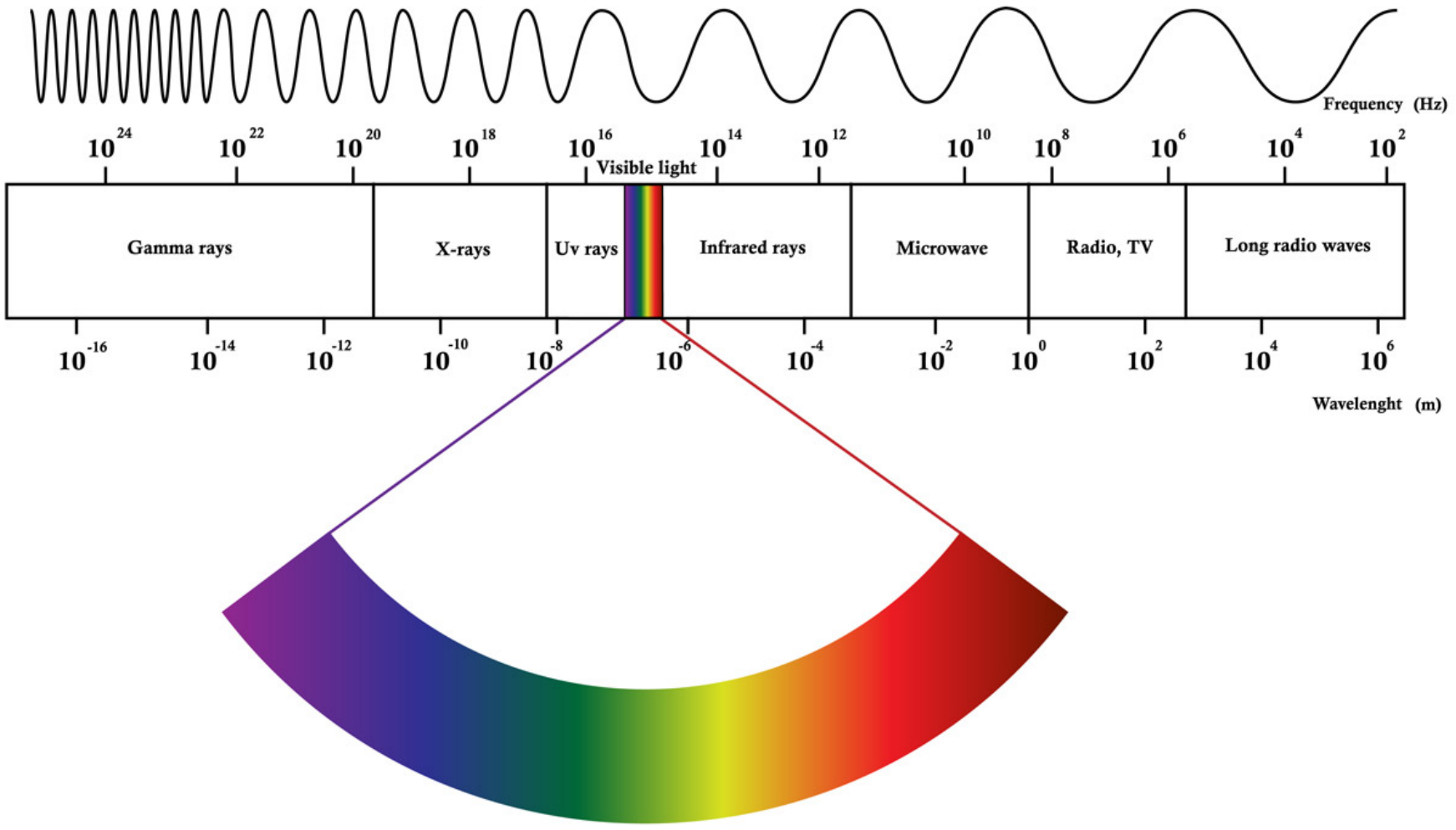
This started with with experiments in “wireless telegraphy” (which actually predate radio) that lead to standard protocols like Morse code. Today, we have a diverse range of digital radio networks like cellular CDMA + GSM, 802.11 WiFi, Bluetooth & BTLE, XigBee, GPS, Digital broadcast television, etc.

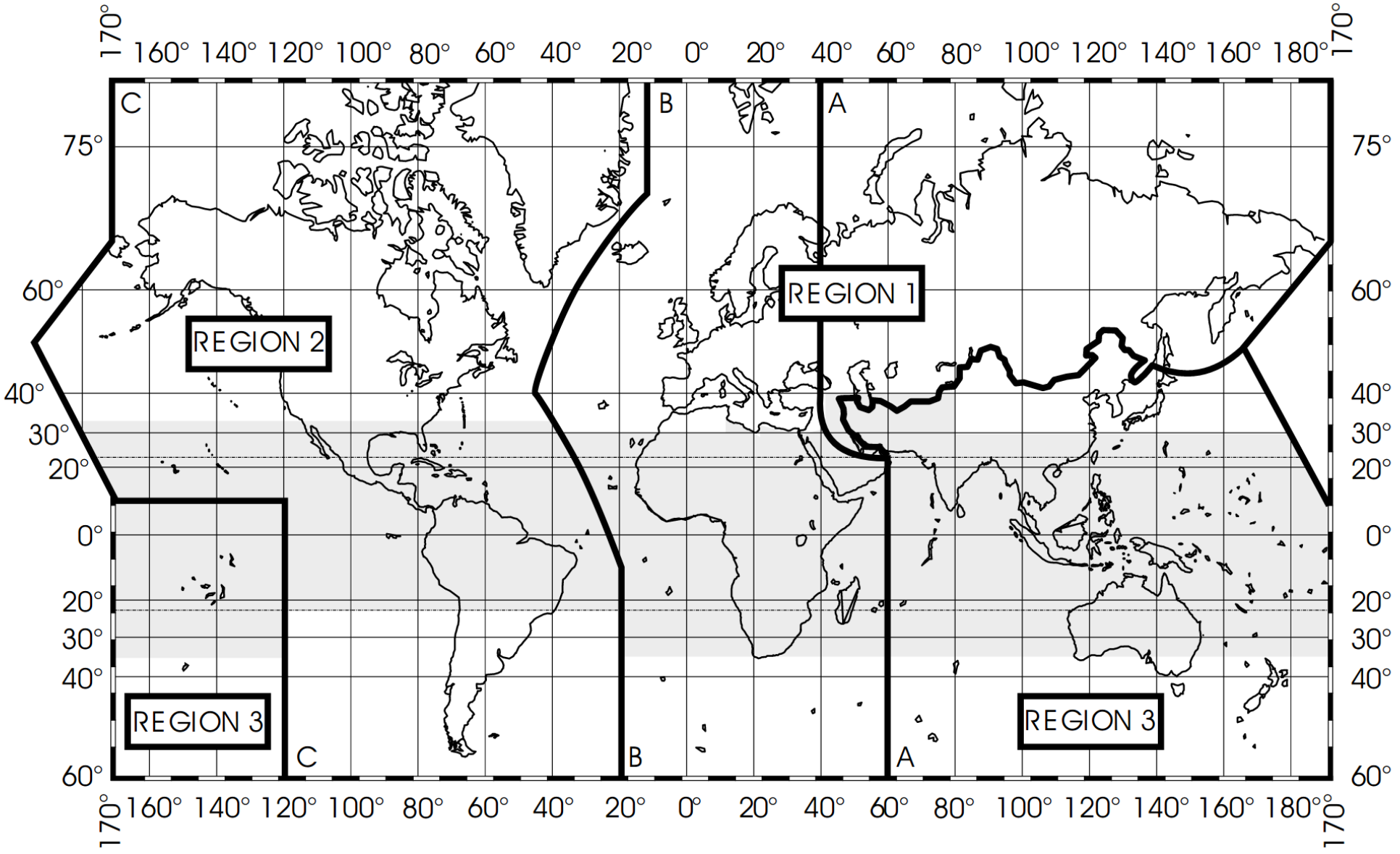
In this workshop, we’ll be taking a look at some older, inefficient, and cheap means of transmitting data over radio by encoding it first as audio.

Electromagnetic Spectrum

“Radio is the technology of using radio waves to carry information, such as sound and images, by systematically modulating properties of electromagnetic energy waves transmitted through space, such as their amplitude, frequency, phase, or pulse width.”

- 469 editors on Wikipedia 


Frequency (Hz)
10^24 10^22 10^20 10^18 10^16 10^14 10^12 10^10 10^8 10^6 10^4 10^2
Visible light
Gamma rays
X-rays
Uv rays
Infrared rays
Microwave
Radio, TV
Long radio waves
10^-16 10^-14 10^-12 10^-10 10^-8 10^-6 10^-4 10^-2 10^0 10^2 10^4 10^6
Wavelenght (m)


170°
160° 140° 120° 100° 80° 60° 40° 20° 0° 20° 40° 60° 80° 100° 120° 140° 160° 180°
170°
C
B
A
75°
60°
40°
30°
20°
0°
20°
30°
40°
60°
REGION 1
REGION 2
REGION 3
REGION 3
C
B
A

UNITED STATES FREQUENCY ALLOCATIONS

THE RADIO SPECTRUM

RADIO SERVICES COLOR LEGEND

- AERONAUTICAL MOBILE
- AERONAUTICAL MOBILE SATELLITE
- AERONAUTICAL RADIONAVIGATION
- AMATEUR
- AMATEUR SATELLITE
- BROADCASTING
- BROADCASTING SATELLITE
- EARTH EXPLORATION SATELLITE
- FIXED
- FIXED SATELLITE
- INTER-SATELLITE
- LAND MOBILE
- LAND MOBILE SATELLITE
- MARITIME MOBILE
- MARITIME MOBILE SATELLITE
- MARITIME RADIONAVIGATION
- METEOROLOGICAL AIDS
- METEOROLOGICAL SATELLITE
- MOBILE
- MOBILE SATELLITE
- RADIO ASTRONOMY
- RADIODETERMINATION SATELLITE
- RADIOLOCATION
- RADIOLOCATION SATELLITE
- RADIONAVIGATION
- RADIONAVIGATION SATELLITE
- SPACE OPERATION
- SPACE RESEARCH
- STANDARD FREQUENCY AND TIME SIGNAL
- STANDARD FREQUENCY AND TIME SIGNAL SATELLITE

ACTIVITY CODE

- GOVERNMENT EXCLUSIVE
- GOVERNMENT/NON-GOVERNMENT SHARED
- NON-GOVERNMENT EXCLUSIVE

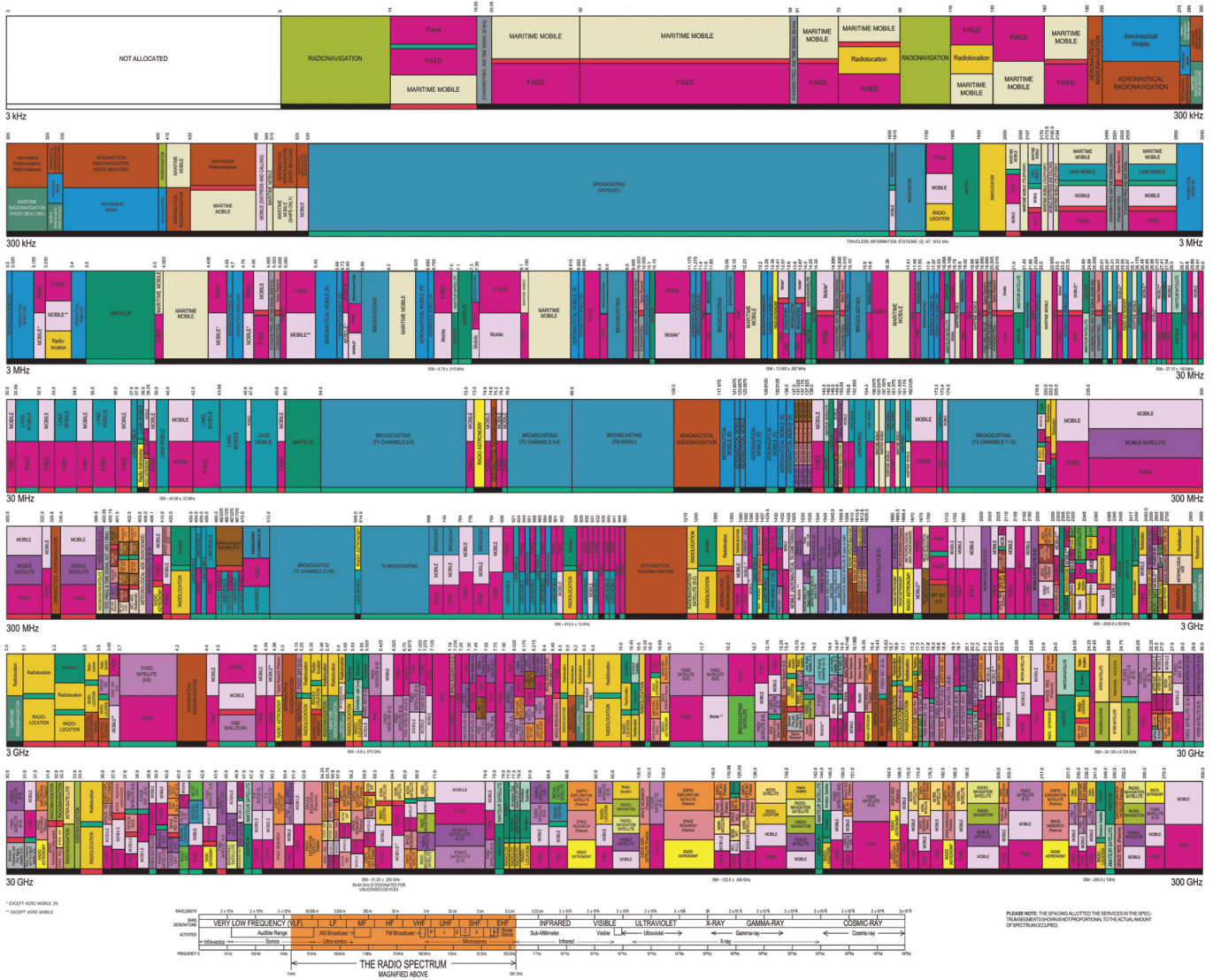
ALLOCATION USAGE DESIGNATION

SERVICE	EXAMPLE	DESCRIPTION
Primary	FIXED	Capital Letters
Secondary	Mobile	1st Capital with lower case letters

This chart is a graphic single-point-in-time portrayal of the Table of Frequency Allocations used by the FCC and NTIA. As such, it does not completely reflect all aspects, i.e., footnotes and recent changes made to the Table of Frequency Allocations. Therefore, for complete information, users should consult the Table to determine the current status of U.S. allocations.

U.S. DEPARTMENT OF COMMERCE
National Telecommunications and Information Administration
Office of Spectrum Management

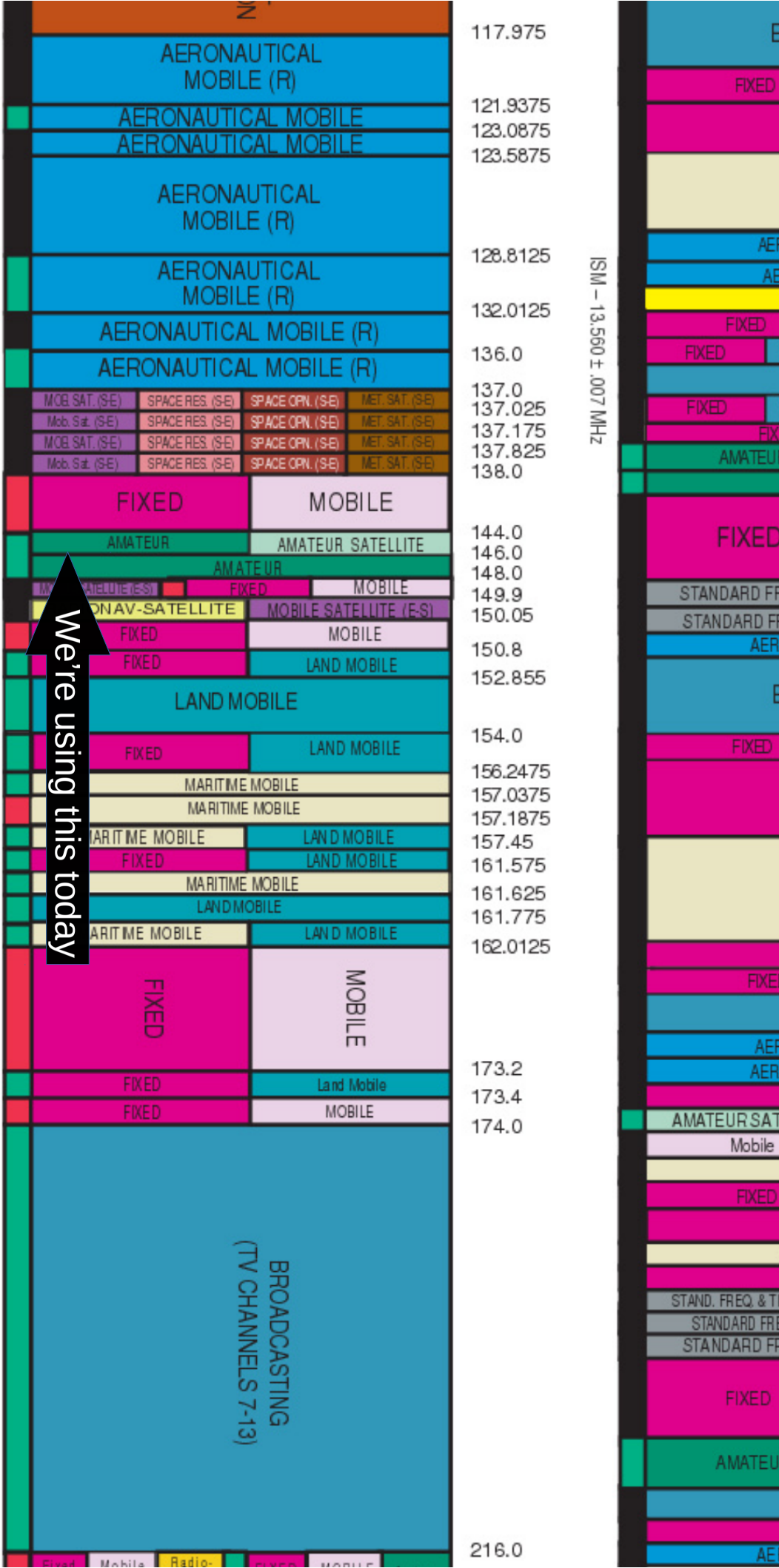
October 2003



* EXCEPT AERO MOBILE (R)

** EXCEPT AERO MOBILE

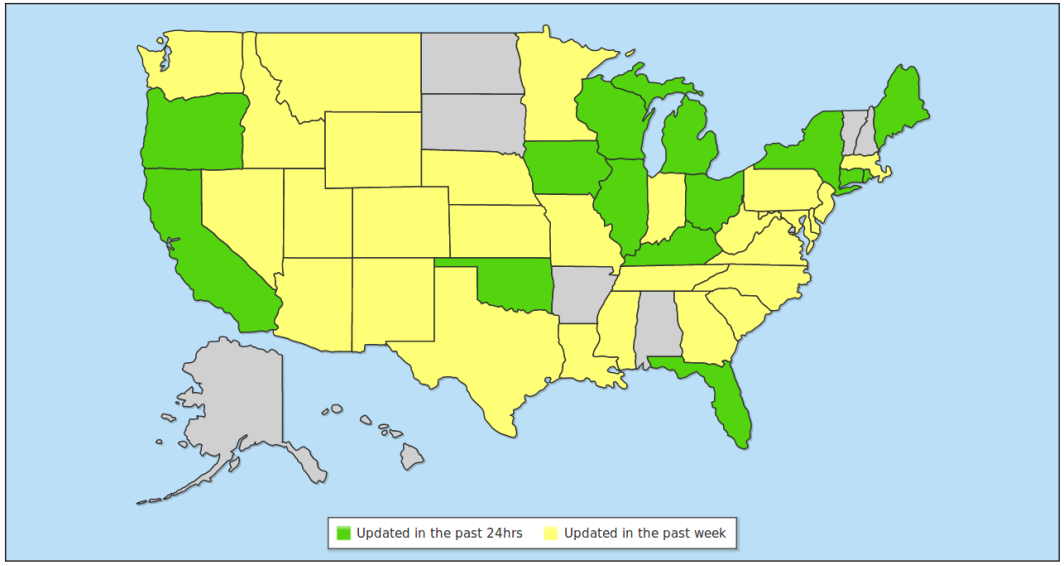
PLEASE NOTE: THE SPACING ALLOTTED THE SERVICES IN THE SPECTRUM SEGMENTS SHOWN IS NOT PROPORTIONAL TO THE ACTUAL AMOUNT OF SPECTRUM OCCUPIED.


117.975
AERONAUTICAL MOBILE (R)
121.9375
AERONAUTICAL MOBILE
123.0875
AERONAUTICAL MOBILE
123.5875
AERONAUTICAL MOBILE (R)
128.8125
AERONAUTICAL MOBILE (R)
132.0125
AERONAUTICAL MOBILE (R)
136.0
AERONAUTICAL MOBILE (R)
137.0
MOB. SAT. (S-E)
SPACE RES. (S-E)
SPACE OPN. (S-E)
MET. SAT. (S-E)
137.025
Mob. Sat. (S-E)
SPACE RES. (S-E)
SPACE OPN. (S-E)
MET. SAT. (S-E)
137.175
MOB. SAT. (S-E)
SPACE RES. (S-E)
SPACE OPN. (S-E)
MET. SAT. (S-E)
137.825
Mob. Sat. (S-E)
SPACE RES. (S-E)
SPACE OPN. (S-E)
MET. SAT. (S-E)
138.0
FIXED
MOBILE
144.0
AMATEUR
AMATEUR SATELLITE
146.0
AMATEUR
148.0
FIXED
MOBILE
149.9
ONAV-SATELLITE
MOBILE SATELLITE (E-S)
150.05
FIXED
MOBILE
150.8
FIXED
LAND MOBILE
152.855
LAND MOBILE
154.0
FIXED
LAND MOBILE
156.2475
MARITIME MOBILE
157.0375
MARITIME MOBILE
157.1875
ARITIME MOBILE
LAND MOBILE
157.45
FIXED
LAND MOBILE
161.575
MARITIME MOBILE
161.625
LAND MOBILE
161.775
ARITIME MOBILE
LAND MOBILE
162.0125
FIXED
MOBILE
173.2
FIXED
Land Mobile
173.4
FIXED
MOBILE
174.0
BROADCASTING (TV CHANNELS 7-13)
216.0
ISM – 13.560 ± .007 MHz
We're using this today
FIXED
STANDARD FR
AMATEUR SAT
Mobile
STAND. FREQ. & TI
FIXED
AMATEU

RadioReference.com
Your Complete Reference Source

The Radio Reference Database (United States)

Choose Country: United States



Retrieve by Location

Retrieve by State
Alabama
Retrieve

Retrieve by Metro Area
Albany-Capital District
Retrieve

Search for a US City/Location
Enter City or Location
Search

Retrieve by US zipcode
Zip
Retrieve

Search Identified Frequencies

Bally's/Wild Wild West

System Name:	Bally's/Wild Wild West
Location:	Atlantic City, NJ
County:	Atlantic
System Type:	Motorola Type II Smartnet
System Voice:	Analog
Last Updated:	July 13, 2012, 11:48 pm (Updated Talkgroups (11 Updated))

https://bit.ly/ballys-freqs

Site	Name	Freqs						
001 (1)	Site-1	935.6375	936.725	937.1625c	937.6625c	937.700	939.225	939.2375

All Talkgroups

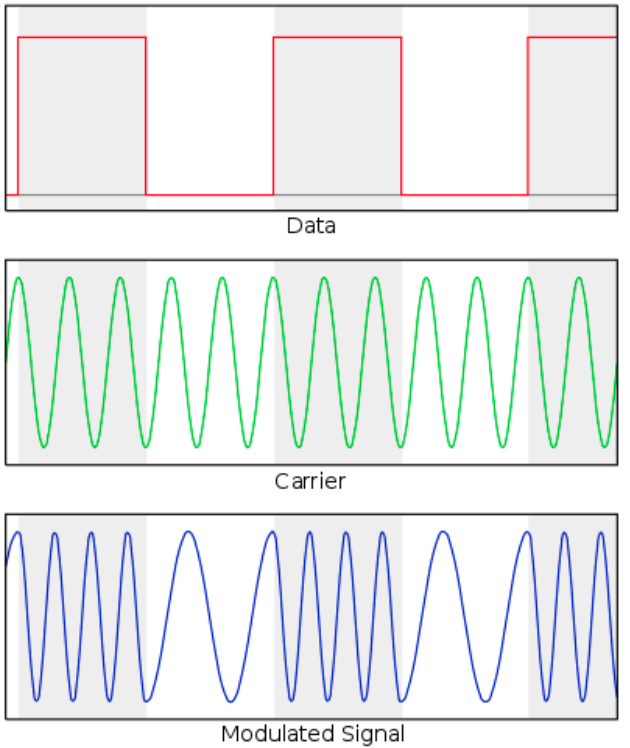
DEC	HEX	Mode	Alpha Tag	Description	Tag
112	007	A	BallySlot112	Slots	Business
144	009	A	BallySlot144	Slots	Business
176	00b	A	BallySlot176	Slots	Business
240	00f	A	Bally Eng	Engineering	Business
272	011	A	Bally Sec	Security	Security
336	015	A	BallySecSurv	Security - Surveillance	Security
400	019	A	BallySlotTch	Slot Technicians	Business
528	021	A	BallyH/K Pub	Housekeeping - Public Areas	Business
560	023	A	Bally Food	Food	Business
848	035	A	Bally Trnsp?	Transportation?	Business
1168	049	A	Bally H/K	Housekeeping	Business

Audio Frequency Modulation

Audio frequency-shift keying (AFSK) is a form of digital modulation that represents binary 1s and 0s by changes in the pitch of an audio tone.

AFSK defines the modulation technique but it doesn't define the transmission medium. Old telephone modems use the exact same type of AFSK that we'll use over radio waves.

Common bit rates for AFSK encoded data transmission of radio include 300, 600, and 1200 baud.



Encoding digital data as audio is slow, but it allows us to re-purpose existing systems that can receive or play audio, like cheap hand-held radios and walkie talkies.

All you need for a simple packet radio transmission setup is:

1. A cheap VHF/UHF radio like the Baofeng UV-5R.

2. A computer with audio input and output.

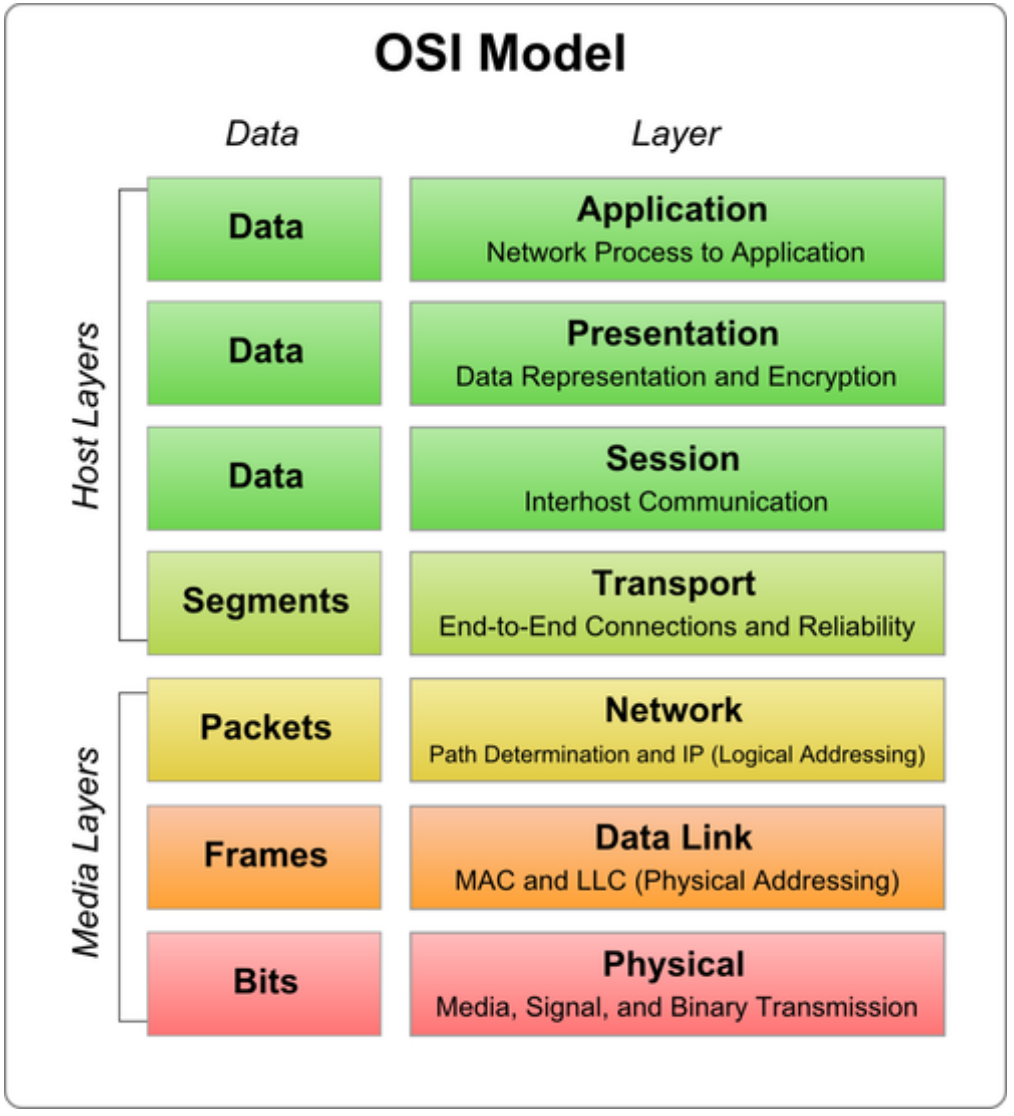
3. An audio cable that connects the computer's output to the radio's input and vice versa.

Packet Radio

Packet Radio builds on top of digital modes like AFSK to group information into logical packets and frames, similar to TCP/IP. In fact, TCP/AX.25 is very common in the packet radio scene.

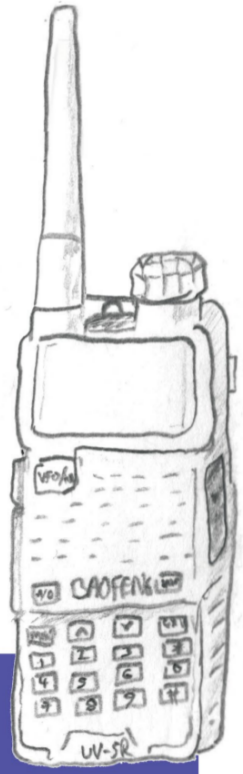
AX.25 (Amateur X.25) is the link-layer protocol of choice. It provides both *connected* and *connectionless* modes and uses amateur radio call signs as addresses.

First Bit Sent						
Flag	**Address**	**Control**	**PID**	**Info.**	**FCS**	**Flag**
01111110	112/560 Bits	8 Bits	8 Bits	N*8 Bits	16 Bits	01111110


OSI Model
Data
Layer
Host Layers
Media Layers
Data
Application
Network Process to Application
Data
Presentation
Data Representation and Encryption
Data
Session
Interhost Communication
Segments
Transport
End-to-End Connections and Reliability
Packets
Network
Path Determination and IP (Logical Addressing)
Frames
Data Link
MAC and LLC (Physical Addressing)
Bits
Physical
Media, Signal, and Binary Transmission

a thing or two about

2nd Edition
Updated and Revised



Messing Around With

Packet Radio

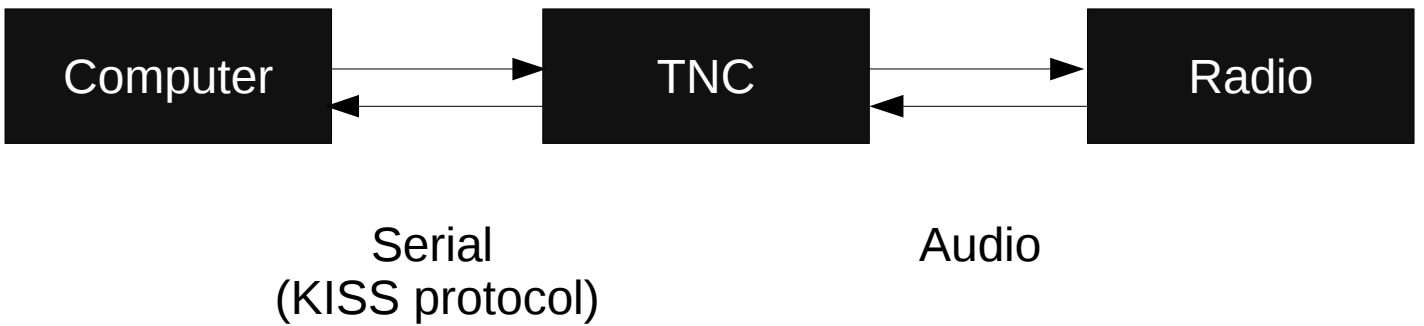
Dennis de Bel and
Roel Roscam Abbing

O'REALLY?

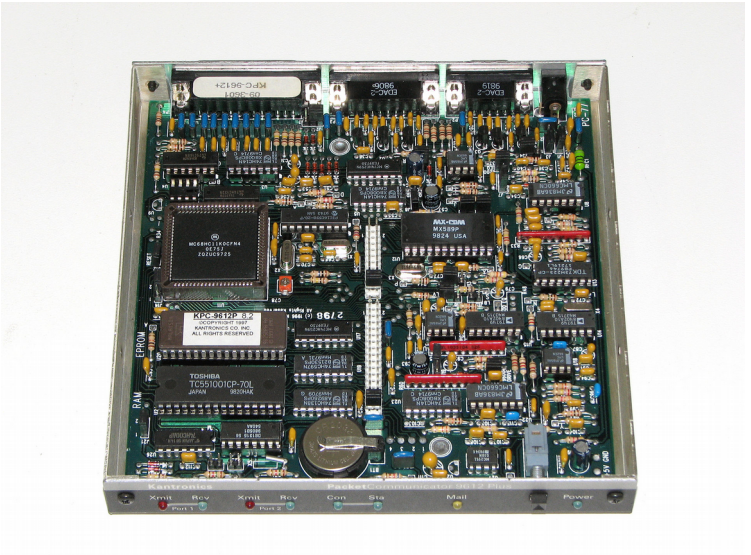
TNC

A *Terminal Node Controller* (TNC) is required for packet radio operation. The TNC is the modem that converts data to audio and vice versa.

A TNC also acts like a network switch, assembling and dissasembling frames and packets. You send the TNC a packet of data, and it monitors the simplex channel and decides when to send that packet over radio.




METRIC
MEASURES UP
NIST Special Publication 376
Revised April 1993
COMMON UNITS
Unit Symbol Measure
millimeter mm
centimeter cm length
meter m
milligram mg
kilogram kg mass
liter L volume
COMMON PREFIXES
Prefix Symbol Multiple
micro μ 1/1 000 000
milli m 1/1 000
centi c 1/100
kilo k 1 000
mega M 1 000 000
giga G 1 000 000 000
THE MODERNIZED METRIC SYSTEM
The "modernized" metric system is the International System of Units or SI (from the French "Le Systeme International d'Unites," which is abbreviated SI). For purposes of international trade, the metric system is more than just SI. It includes the product standards and preferred sizes that are accepted by industries and governments throughout the world.
NOTE: Scales and conversions are approximate (≈)
NIST
U.S. Department of Commerce
Technology Administration
National Institute of Standards and Technology
For sale by the Superintendent of Documents
U.S. Government Printing Office
Washington, D.C. 20402-9325
Advanced Electronic Applications, Inc.
Model PK-232MBX
THRESHOLD
DCD
TUNE
STATUS
MODE
RADIO 1
OFF


KPC-9612P 8.2
TOSHIBA
TC551001CP-70L
JAPAN
EPROM
RAM
Kantronics
Xmit
Rcv
Con
Sta
Mail
Power

Dire Wolf is a software "soundcard" AX.25 packet modem/TNC and APRS encoder/decoder. It can be used stand-alone to observe APRS traffic, as a tracker, digipeater, APRStt gateway, or Internet Gateway (IGate). For more information, look at the bottom 1/4 of this page and in https://github.com/wb2osz/direwolf/blob/dev/doc/README.md

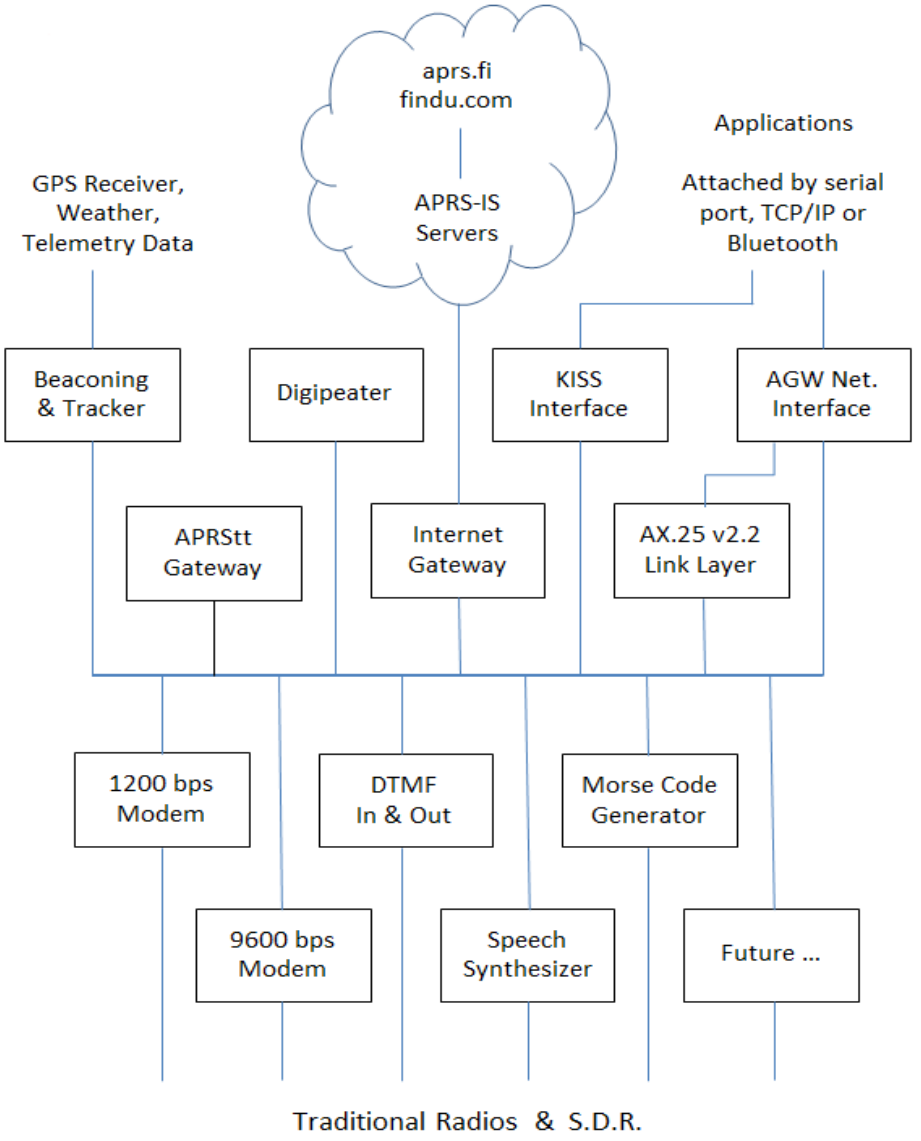
tnc igate ham-radio packet-radio raspberry-pi digipeater aprs ax25

194 commits | **3** branches | **17** releases | **7** contributors | View license

Branch: **master** | New pull request | Create new file | Upload files | Find file | Clone or download

wb2osz Issue 196 - Compatibility with GPSD API 7. Latest commit `a1e2d1c` 7 days ago

doc	Add link to separate repository with presentations.	3 months ago
geotranz	Uninitialized variables found by static analysis.	a year ago
man1	Time stamps and documentation for kissutil.	a year ago
misc	Compatibility with minGW gcc 5.3.0	2 years ago
regex	Fix compile warnings found when adding -Wall and others.	2 years ago
telemetry-toolkit	Add script to generate telemetry sequence numbers.	3 years ago
.gitattributes	Rewrite GPS handling. Lots of other clean up.	3 years ago
.gitignore	ignore file tweaks	11 months ago

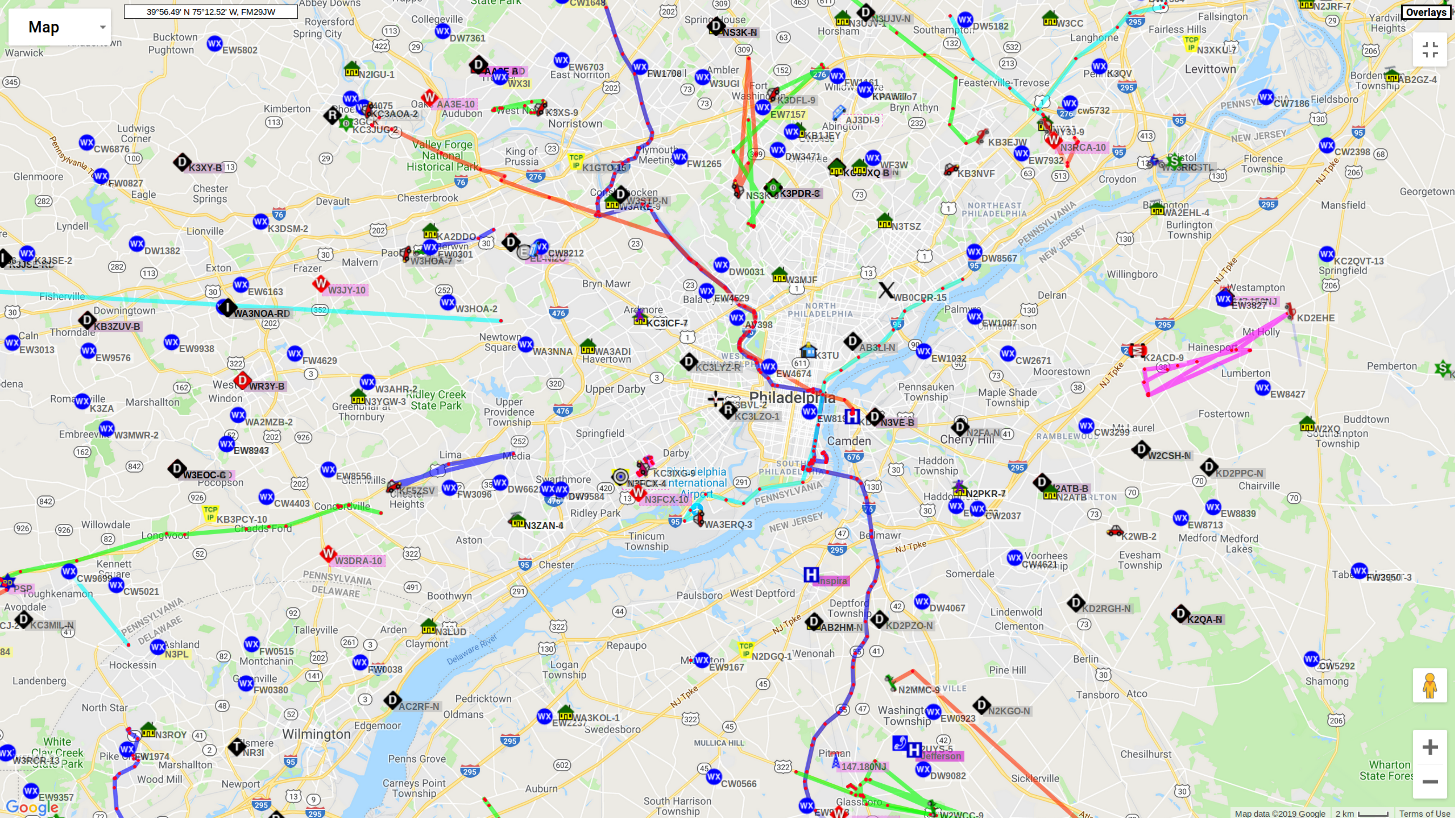

aprs.fi
findu.com
APRS-IS
Servers
Applications
Attached by serial
port, TCP/IP or
Bluetooth
GPS Receiver,
Weather,
Telemetry Data
Beaconing
& Tracker
Digipeater
KISS
Interface
AGW Net.
Interface
APRStt
Gateway
Internet
Gateway
AX.25 v2.2
Link Layer
1200 bps
Modem
DTMF
In & Out
Morse Code
Generator
9600 bps
Modem
Speech
Synthesizer
Future ...
Traditional Radios & S.D.R.

Automatic Packet Reporting System

A global packet radio network supporting GPS, weather station telemetry, text messages, announcements, bulletin boards and more. APRS data is often displayed on a map, showing stations, objects, tracks of moving objects, and direction finding data.

144.39 MHz

1200 baud

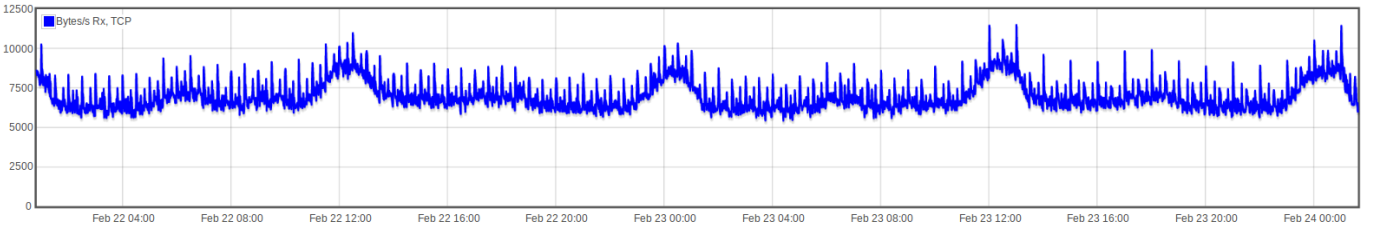

Map
39°56.49' N 75°12.52' W, FM29JW
Overlays
Philadelphia
Camden
Wilmington
Norristown
Levittown
Cherry Hill
Chester
Upper Darby
Bryn Mawr
Media
Valley Forge National Historical Park
Ridley Creek State Park
Philadelphia International Airport
Delaware River
PENNSYLVANIA
NEW JERSEY
DELAWARE
NORTHEAST PHILADELPHIA
NORTH PHILADELPHIA
SOUTH PHILADELPHIA
Map data ©2019 Google
2 km
Terms of Use
Google

T2KA aprsc status
2019-02-24 01:38:38z

APRSC

Server

Server ID	T2KA
Server admin	Bernd Strehhuber, DM8BS
Software	aprsc 2.1.4-g408ed49
Software features	epoll posix_cap clock_gettime gcc_atomics zlib ssl sctp
Uptime	83d5h
Server started	2018-12-02 20:17:48z
Operating system	Linux i686



Zoom

Totals

Clients	249	0/s
Connects	22151828	0.10/s
Bytes Tx TCP	273056815218	32780/s
Bytes Rx TCP	49563191642	6307/s
Packets Tx TCP	2677287144	329/s
Packets Rx TCP	495021523	65/s
Bytes Tx UDP	0	0/s
Bytes Rx UDP	267196	0/s
Packets Tx UDP	0	0/s
Packets Rx UDP	2404	0/s
Bytes Tx SCTP	0	0/s
Bytes Rx SCTP	0	0/s
Packets Tx SCTP	0	0/s
Packets Rx SCTP	0	0/s

Duplicate filter +

Duplicate packets dropped	22374468	3.9/s
Unique packets seen	470102173	62/s

Port listeners

Proto	Address	Name	Clients	Peak	Max	Connects	Conn/s	Packets Tx	Packets Rx	Bytes Tx	Bytes Rx	Tx/Rx bytes/s
tcp	[::]:14580	Client-defined filter	245	304	2000	22059429	0.10	693995054	31269576/20847300/2215811	77989898608	3832760900	8895 / 405
udp	[::]:14580		0	0	10	0	0	0	0/0/0	0	0	0 / 0
tcp	[::]:10152	Full feed	4	12	100	92399	0	1974593910	664178/166601/3540	194214064186	147850054	23755 / 15
udp	[::]:10152		0	0	10	0	0	0	0/0/0	0	0	0 / 0

Chattervox

Chattervox is a packet radio chat protocol with support for digital signatures and binary compression; think IRC over radio waves.

It’s a new protocol with a reference implementation and command-line interface written in TypeScript.

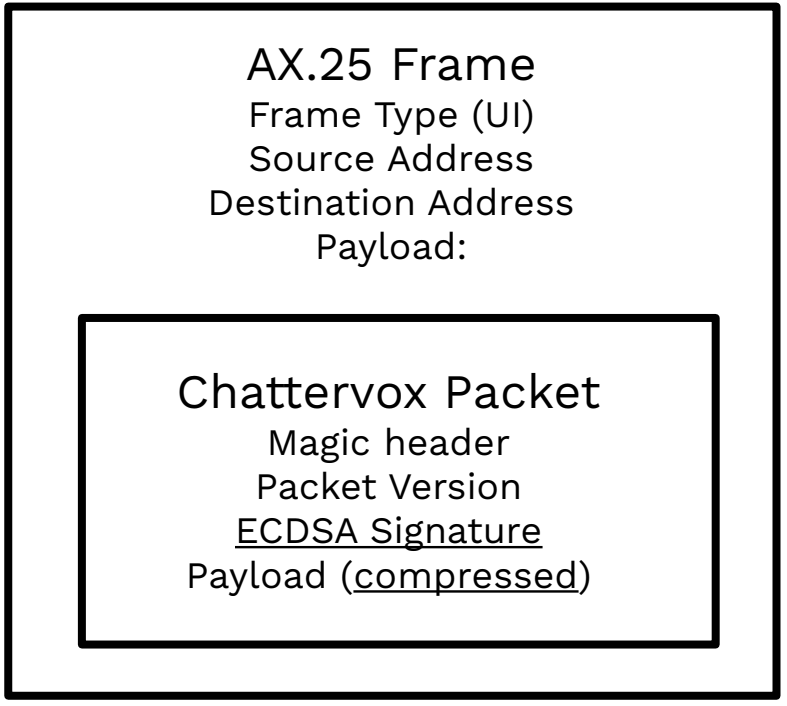
In the United States, it's illegal to broadcast encrypted messages on amateur radio frequencies. Chattervox respects this law, while using elliptic curve cryptography and digital signatures to protect against message spoofing.

FCC Title 47 Part §97.113

Prohibited transmissions

Section (4) Music using a phone emission except as specifically provided elsewhere in this section; communications intended to facilitate a criminal act; messages encoded for the purpose of obscuring their meaning, except as otherwise provided herein; obscene or indecent words or language; or false or deceptive messages, signals or identification.

Packet Encapsulation



Legal

It's illegal for non-licensed individuals to transmit on amateur frequency bands. There are, however, clauses that allow *unlicensed* individuals to speak, key/type, or otherwise transmit communication on behalf of a licensed amateur while they are under the direct supervision of a licensed control operator (*Third Party Traffic)*. In other words, unlicensed peeps can group up with licensed folks.

Alternatively, the Multi-User Radio Service (MURS) provides 5 frequency channels in the VHF band that are *licensed by rule*, meaning anyone can use them without a license, provided they follow the rules.

Some MURS rules:

Blue Dot MURS-4, 154.570 MHz
Green Dot MURS-5, 154.600 MHz

2 Watts Power
Minimize interference

Voice, data, image, telemetry allowed
Repeaters & signal boosters NOT allowed

Frequencies

144.39 MHz
APRS

145.0 MHz
Chattervox 1

145.5 MHz
Chattervox 2

APRS CTVX 1 CTVX 2 MURS 4 MURS 5

154.57 MHz
Blue Dot MURS 4

154.6 MHz
Green Dot MURS 5

Baofeng UV-5R

The best (and only) RX/TX radio money can buy for $25.

Made by a Chinese company that doesn't have to manufacture their radios to US standards.

Probably the most controversial radio on the scene. People either love them or hate them. In August 2018 the FCC issued citations to US distributors on Amazon.

Frequency Ranges:
- 136-174MHz (VHF)
- 400-520MHz (UHF)

Bandwidth:
- 25KHz on WIDE
- 12.5KHz on NARROW

Power Output:
- 5W on HIGH
- 1W on LOW

Features:
- VOX & PTT
- 128 Programmable Channels
- Flashlight

https://brannon.online | @brannondorsey

VOX & Squelch

Simplex radios like the Baofeng can't receive and transmit at the same time. They can either be listening or speaking, but not both. All transmitters in a conversation share one frequency and they have to take turns speaking.

VOX stands for *Voice Operated Transmit*. It uses a radio's microphone to conditionally trigger transmit functionality as soon as input is detected. This is in contrast to *Push to Talk* (PTT) operation where the user (or computer) explicitly signals that they'd like to transmit.

Squelch sets a noise floor that must be broken by a transmitter in order for a receiving radio to sonify the signal.

If VOX is on squelch must be on as well.

Installing Chattervox

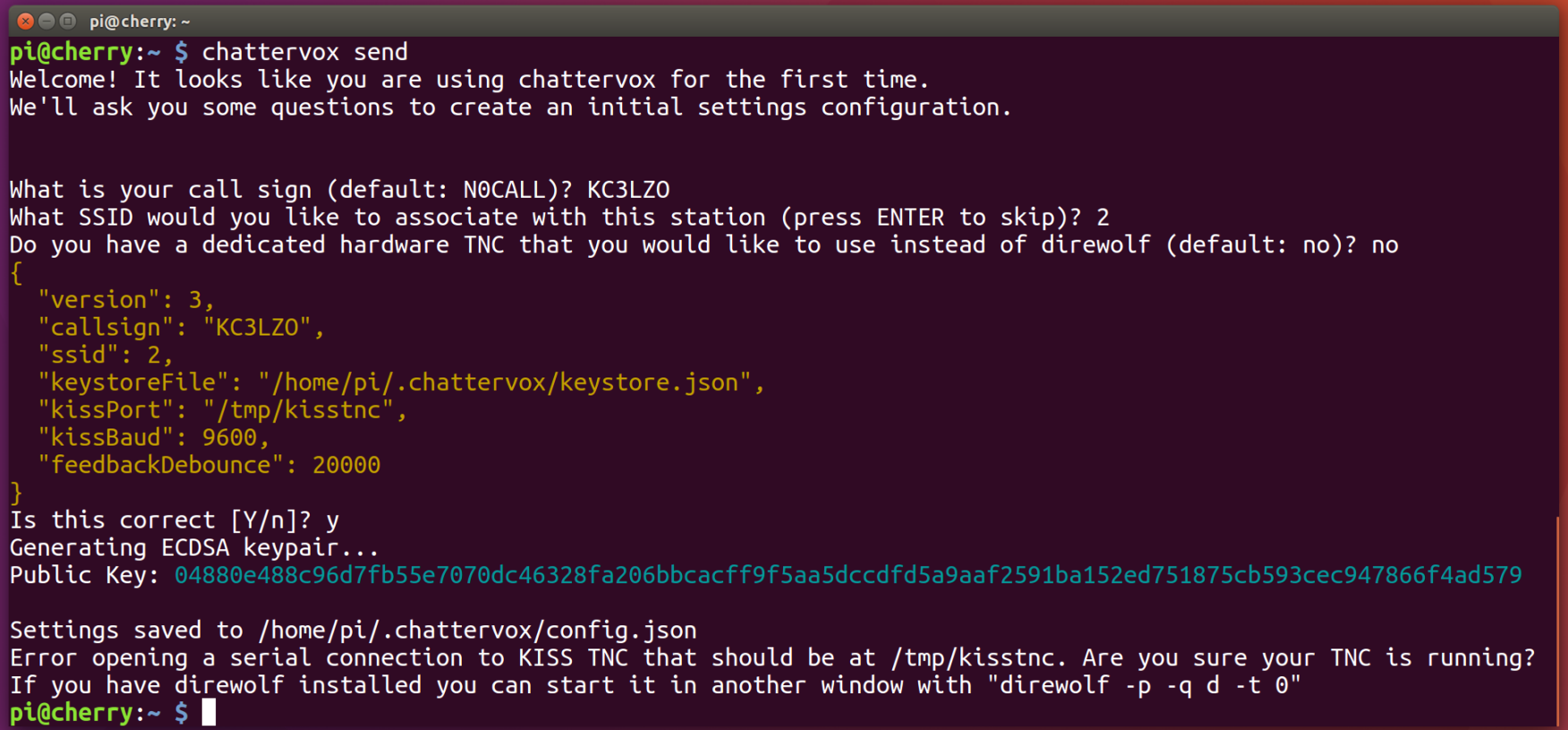
Chattervox is currently fully supported on Linux, *kind of* supported on MacOS*, and not yet supported on Windows. Chattervox requires a software or hardware TNC to operate.

*https://bit.ly/chattervox-macos

```
# clone, build, and install the Direwolf TNC
git clone https://github.com/wb2osz/direwolf
cd direwolf
make
sudo make install
make install-conf

# install node via the node version manager
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.34.0/install.sh | bash
source ~/.bashrc # or ~/.bash_profile
nvm install v8 # install node version 8

# install chattervox
npm install -g --cli chattervox
```


pi@cherry: ~
pi@cherry:~ $ chattervox send
Welcome! It looks like you are using chattervox for the first time.
We'll ask you some questions to create an initial settings configuration.
What is your call sign (default: N0CALL)? KC3LZO
What SSID would you like to associate with this station (press ENTER to skip)? 2
Do you have a dedicated hardware TNC that you would like to use instead of direwolf (default: no)? no
{
"version": 3,
"callsign": "KC3LZO",
"ssid": 2,
"keystoreFile": "/home/pi/.chattervox/keystore.json",
"kissPort": "/tmp/kisstnc",
"kissBaud": 9600,
"feedbackDebounce": 20000
}
Is this correct [Y/n]? y
Generating ECDSA keypair...
Public Key: 04880e488c96d7fb55e7070dc46328fa206bbcacff9f5aa5dccdfd5a9aaf2591ba152ed751875cb593cec947866f4ad579
Settings saved to /home/pi/.chattervox/config.json
Error opening a serial connection to KISS TNC that should be at /tmp/kisstnc. Are you sure your TNC is running?
If you have direwolf installed you can start it in another window with "direwolf -p -q d -t 0"
pi@cherry:~ $

```
pi@cherry:~ $ direwolf -p -q d -t 0
Dire Wolf version 1.5 (Feb 16 2019) Beta Test 4
Includes optional support for:  cm108-ptt

Reading config file direwolf.conf
Audio device for both receive and transmit: plughw:1,0  (channel 0)
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, E+, 44100 sample rate / 3.
Note: PTT not configured for channel 0. (Ignore this if using VOX.)
Ready to accept AGW client application 0 on port 8000 ...
Ready to accept KISS TCP client application 0 on port 8001 ...
Virtual KISS TNC is available on /dev/pts/2
Created symlink /tmp/kisstnc -> /dev/pts/2
[0L] KC3LZO-2>CQ:z9<0x01><0x02>705<0x02><0x18>=n}o<0x17><0x03>#.0x02><0x19><0x00>0x0
a>)6atla<0x0d><0x1c>.UThis is an example chattervox message!
[0L] KC3LZO-2>CQ:z9<0x01><0x02>705<0x02><0x19><0x00><0x07>z<0x14><0x17>zKx'<0x1c>Ɔ<0
x02><0x18>2b?Dm4<0x16>DGI0x1b>0x0c>WDYou'll see that longer messages usually get com
pressed.
[0L] KC3LZO-2>CQ:z9<0x01><0x03>604<0x02><0x18>s?<0x0e>0x0a>grV<0x13>nꞤU%7Rw<0x02><0x
18>P<0x12><0x08>HNh!i]<0x13>G<0x13><0x01><0x7f><0x0b>,QKUH/H*-Q(,<0x06><0x0b><0x16><
0x14>%&*g(($<0x16><0x14><0x16><0x17>0x01><0x00>
[0L] KC3LZO-2>CQ:z9<0x01><0x03>604<0x02><0x18>F<0x1c>Ǩ<0x02><0x18><0x0d>lķĩMbv<0x18>
QKs-(J-.⌋TN.-*VS(U0x05>J%*d<0x16>+<0x14><0x16><0x01>U%*$C0x01><0x00>
[0L] KC3LZO-2>CQ:z9<0x01><0x03>604<0x02><0x18><0x09>G1<0x13>.S?<0x1a>jpz<0x02><0x1e>
<0x02><0x18>{<0x00>ri<0x1d>!<0x17>gk<0x18><0x1d>0x1a>~x-*NUH,)I-*˯<0x00>2S<0x15>K<0x
14>J0x15><0x0b>R<0x15>j<0x15>rT=<0x00>
```

```
KC3LZO-2: This is an example chattervox message!
KC3LZO-2: You'll see that longer messages usually get compressed.
KC3LZO-2: That one didn't, but this one probably will get compressed.
KC3LZO-2: Compression only occurs if the message is smaller once compressed.
KC3LZO-2: Otherwise chattervox chooses not to compress the message.
KC3LZO-2:
```

```
[0] 0:node*                                                    "cherry" 20:41 24-Feb-19
```

Basic Usage

```
# open the chat room
chattervox chat

# send a packet from the command-line
chattervox send "this is a chattervox packet sent from the command-line."

# receive *all* packets and print them to stdout
chattervox receive --allow-all

# generate a new public/private key pair, and use it as your default signing key
chattervox genkey --make-signing

# add a friend's public key to your keyring, so that chattervox can verify their messages
chattervox addkey KC3LZO \
044da0d4c38bed6e5bc418231cb2dca4f690d858d36c38a032732553b76262a1adfccf588b6c1f9d7734b1bbce90914f82

# remove a friend's public key if it has become compromised
chattervox removekey KC3LZO \
0489a1d94d700d6e45508d12a4eb9be93386b5b30feb2b4aa07836398781e3d444e04b54a6e01cf752e54ef423770c00a6

# print all keys in your keyring
chattervox showkey
```

Chattervox Key Registry

Discussion on GitHub issues lead to the creation of a centralized “key server” where hams can register and share their keys via a secure channel.

For now, keys are added via pull requests to the chattervox-keys repository. (https://bit.ly/chattervox-keys)

The list of active and revoked keys is maintained and hosted on GitHub.

Future versions of Chattervox may allow you to automatically sync your local key store with these lists. Non-centralized key servers may also be explored in the future.

Chattervox Examples

I've created a collection of example applications and use cases for the Chattervox protocol @ https://bit.ly/chattervox-examples

Low-Fi Time Server: Broadcast a time stamp beacon at regular intervals
A weather broadcast station: Pulls local weather data from the Internet and broadcasts it via Chattervox
Breaking news headlines: Pulls breaking news headlines from the Internet and broadcasts them via Chattervox (technically illegal on Amateur Frequencies)
Remote shell: Use Chattervox to control a remote computer via Bash
Zork: Play the famous text adventure game over packet radio

Byte Offset	# of Bits	Name	Value	Description
0x0000	16	Magic Header	0x7a39	A constant two-byte value used to identify chattervox packets.
0x0002	8	Version Byte	Number	A protocol version number between 1-255.
0x0003	6	Unused Flag Bits	Null	Reserved for future use.
0x0003	1	Digital Signature Flag	Bit	A value of 1 indicates that the message contains a ECDSA digital signature.
0x0003	1	Compression Flag	Bit	A value of 1 indicates that the message payload is compressed.
[0x0004]	[8]	[Signature Length]	Number	The length in bytes of the digital signature. This field is only included if the Digital Signature Flag is set.
[0x0004 or 0x0005]	[0-2048]	[Digital Signature]	Bytes	The ECDSA digital signature created using a SHA256 hash of the message contents and the sender's private key.
0x0004-0x104	0-∞	Message	Bytes	The packet's UTF-8 message payload. If the Compression Flag is set the contents of this buffer is a raw DEFLATE buffer containing the UTF-8 message.

[] indicates an optional field.

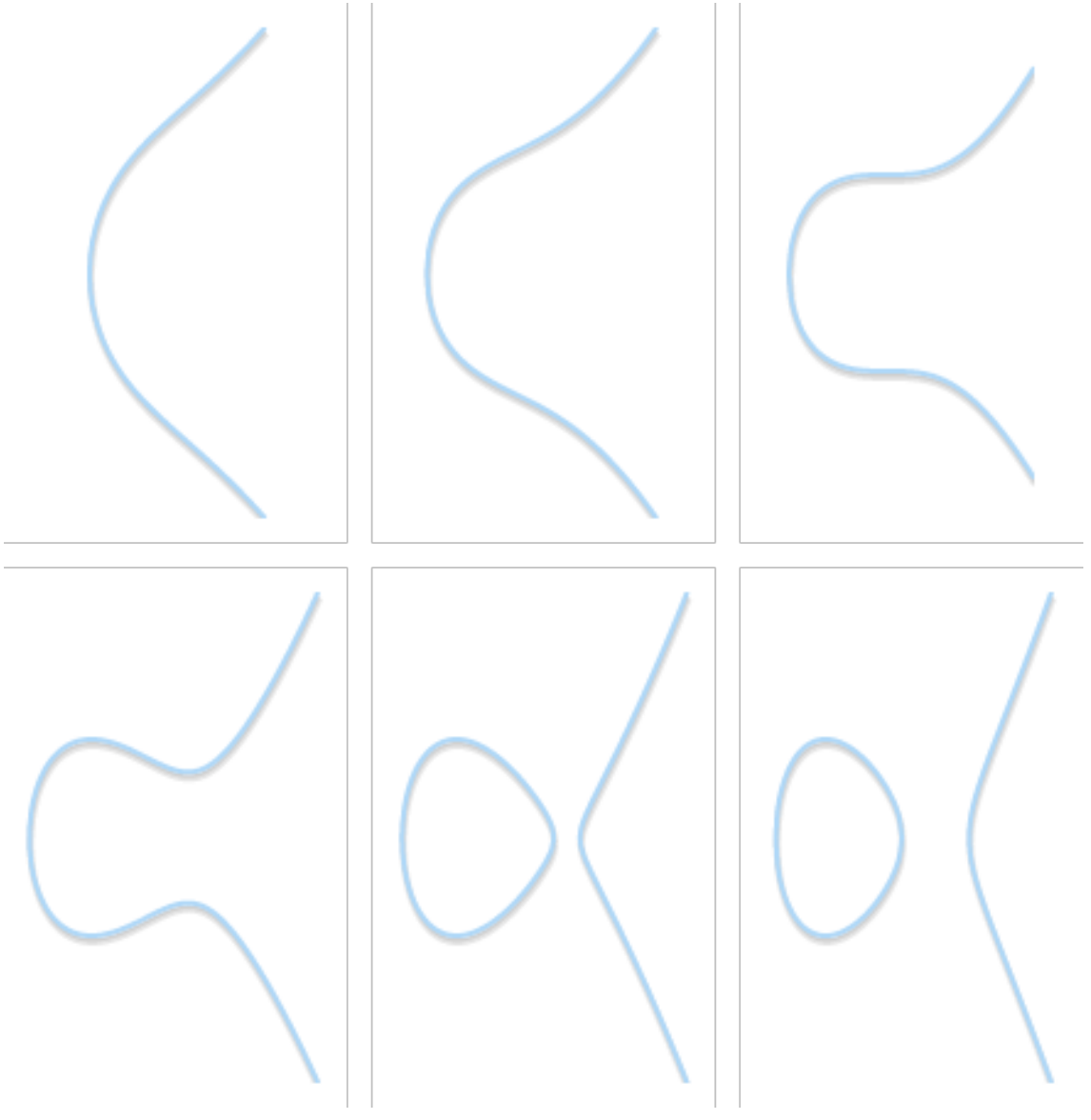
What is an Elliptic Curve?

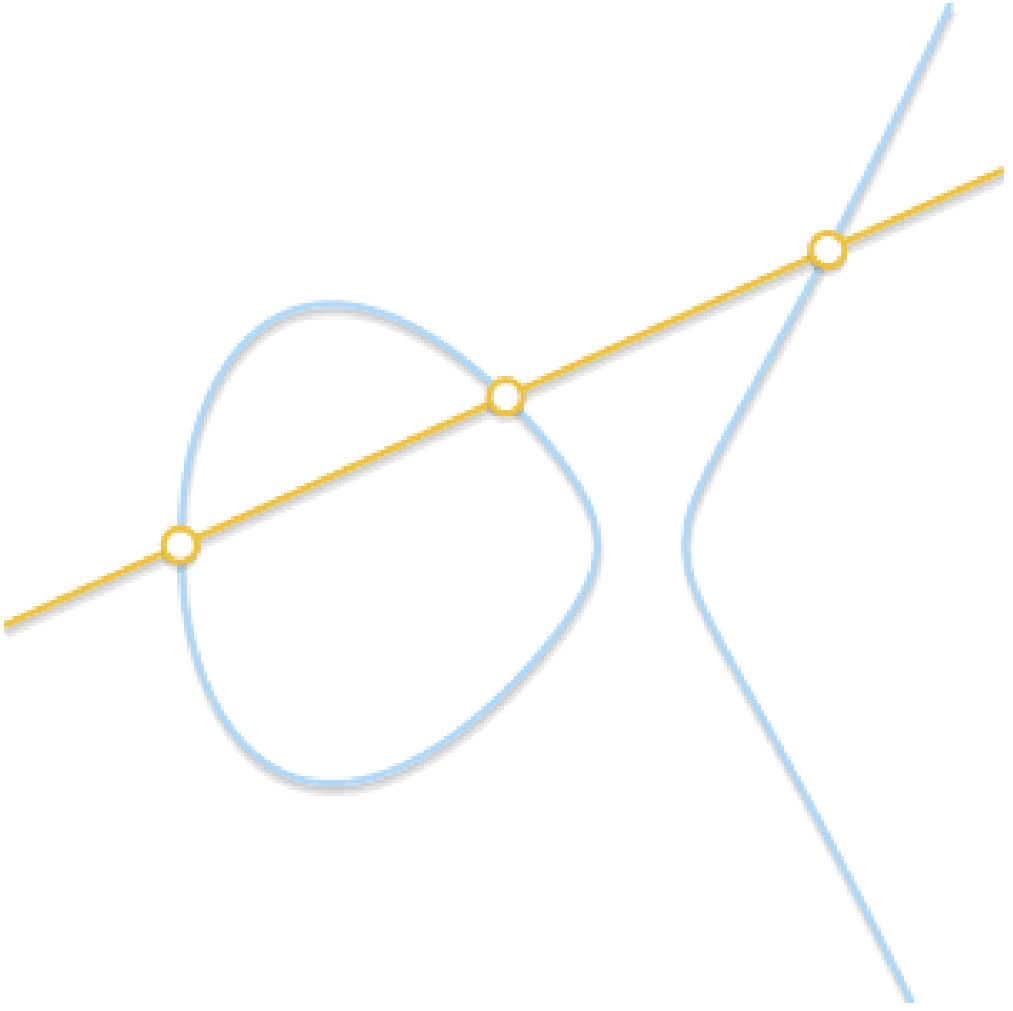
An elliptic curve is the set of points that are described by the equation...

$$y^2 = x^3 + ax + b$$

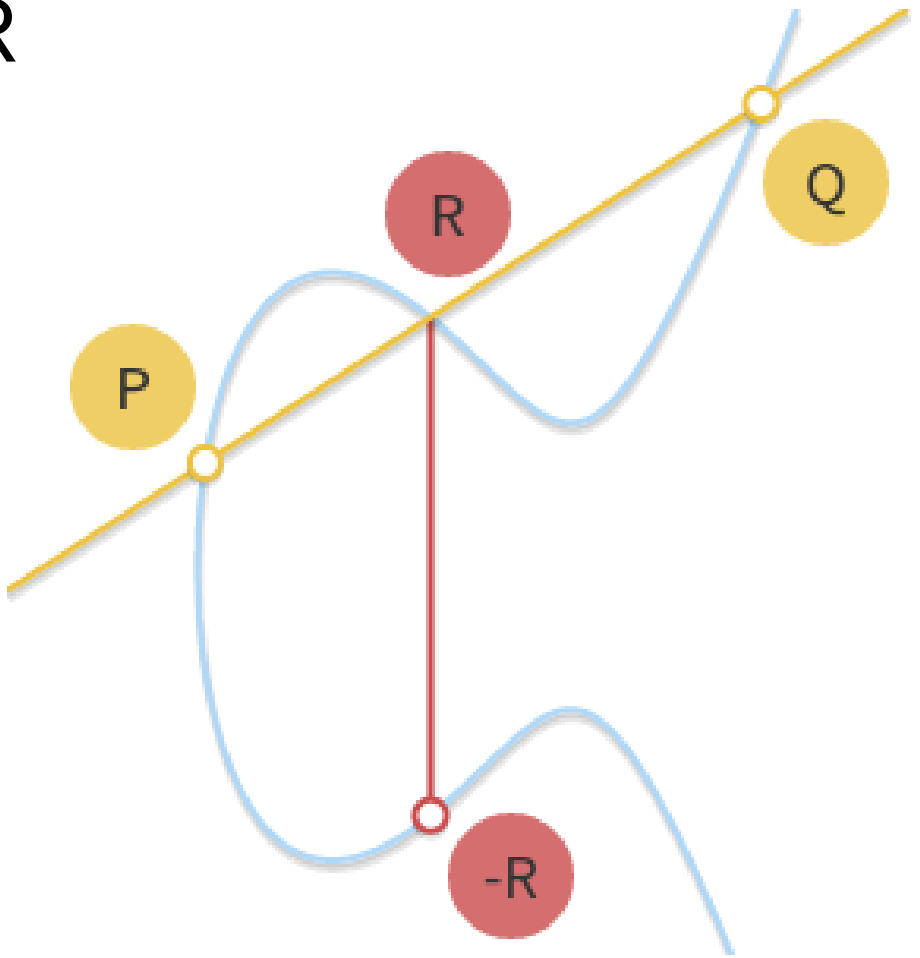
Elliptic curves are defined by two variables, **a** and **b**.

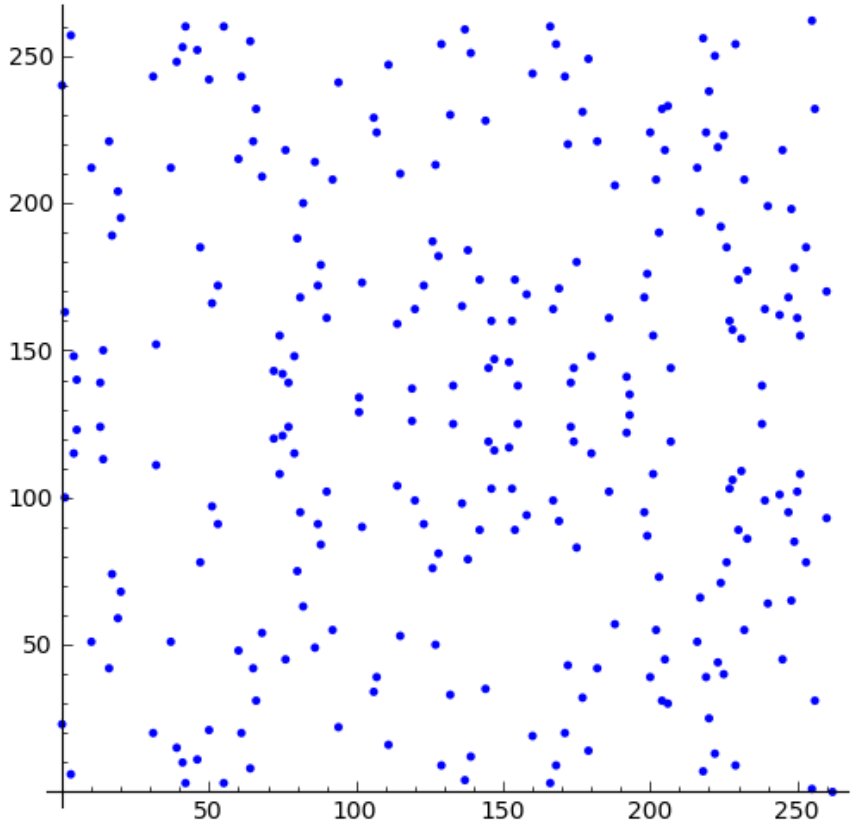
Curves are more general than functions and allow multiple *y* output values to exist for each *x* input value.





P * Q = -R




250
200
150
100
50
50
100
150
200
250

STANDARDS FOR EFFICIENT CRYPTOGRAPHY

SEC 2: Recommended Elliptic Curve Domain Parameters

Certicom Research

Contact: secg-talk@lists.certicom.com

September 20, 2000
Version 1.0

2.5.2 Recommended Parameters secp192r1

The verifiably random elliptic curve domain parameters over $\mathbb{F}_p$ `secp192r1` are specified by the sextuple $T = (p, a, b, G, n, h)$ where the finite field $\mathbb{F}_p$ is defined by:

$$p = \texttt{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF}$$
$$= 2^{192} - 2^{64} - 1$$

The curve E: $y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ is defined by:

$$a = \texttt{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFC}$$
$$b = \texttt{64210519 E59C80E7 0FA7E9AB 72243049 FEB8DEEC C146B9B1}$$

That's an "E" 😮

E was chosen verifiably at random as specified in ANSI X9.62 [1] from the seed:

$$S = \texttt{3045AE6F C8422F64 ED579528 D38120EA E12196D5}$$

The base point G in compressed form is:

$$G = \texttt{03 188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012}$$

and in uncompressed form is:

$$G = \texttt{04 188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012}$$
$$\texttt{07192B95 FFC8DA78 631011ED 6B24CDD5 73F977A1 1E794811}$$

Finally the order n of G and the cofactor are:

$$n = \texttt{FFFFFFFF FFFFFFFF FFFFFFFF 99DEF836 146BC9B1 B4D22831}$$
$$h = \texttt{01}$$

ECDSA

The ECDSA *Signature Algorithm* creates a point (*r*, *s*) using a message, a private key (d_A), and a **curve**.

$e = \text{SHA256}(\text{message})$
$z = \text{left\_bits}(e, \text{bit\_length}(\mathbf{n}))$
$k = \text{random}(1, \mathbf{n}-1)$
$(x, y) = k * \mathbf{G}$
$r = x \bmod \mathbf{n}$
$s = k^{-1} * (z + r * d_A) \bmod \mathbf{n}$

$\text{signature} = (r, s)$

The ECDSA *Verification Algorithm* verifies that **curve** point (*r*, *s*) was derived from a message, and the unknown private key (d_A) associated with a known public key (Q_A).

$e = \text{SHA256}(\text{message})$
$z = \text{left\_bits}(e, \text{bit\_length}(\mathbf{n}))$
$w = s^{-1} \bmod n$
$u_1 = z * w \bmod n$
$u_2 = r * w \bmod n$
$(x, y) = u_1 * \mathbf{G} + u_2 * Q_A$

$\text{is\_valid if } r \equiv x \bmod n$

Private Key (scalar)

fccefae6a899e93b68c0ce7d6552449b6ef5b61ef0d26d78

Puplic Key (vector)

pubic_key = curve_base_point * private_key

04427306E1725abb009991a132bc5a9346de5531915d8946d591fdfa8825bbc037ad8f818b043e6d9994ad58e64a405368

public_key_format = concat(“04”, x, y)

Thanks

https://github.com/brannondorsey

https://brannon.online

@brannondorsey