Internet wide Callsign database using LDAP

Mark Sproul
1368 Noah Road
North Brunswick, NJ 08902
KB2ICI@AMSAT.ORG

Introduction:

LDAP, Lightweight Directory **Access** Protcol, is an internet protocol for storing directories. It has many uses but the mo s t common use is for keeping track of large lists of personel. Many attributes can be associated with each entry such as name, email address and callsign.. LDAP is a standard and is easy to incorporate support for it in any application. There are libraries for use with C/C++ Java, PHP and many other languages as well as all platforms. LDAP is also supported already in many end-user applications sue h as Netscape.

There are many commercial products for call.sign lookup and there is the FCC database that is updated about once a week that c an be downloaded (at least 70 megabytes). There are also many web sites that allow a user to do a quick call sign lookup.

However, there is no easy way for a developer to add call sign support to other applications without writing the callsign lookup code. The web based systems are intended for a user to manually type in a callsign and press a button.

By implementing a universal callsign database in an internet wide standard directory, all internet users will immediately have instant access to callsign lookup and email directory for all other hams in the US.

Background:

LDAP is Lightweight Directory Access Protcol and is a directory service. This differs from a database in that the overall design is optimized for a very high read to write ratio. A normal database has to have more equal performance for read and write. Additionally

LDAP is optimized for holding lots of small bits of information instead of large data records. LDAP was developed by the University of Michigan in response to the heavyweight X-500 system.

To get a feel for LDAP, open up the NETSCAPE Address Book and click on NetCenter, then type "sproul" into the Names Containing field, you will get a long list of people from around the country.

Just as a side note, LDAP has replaced the registry in Windows 2000 for internal storage of parameters.

Project:

The LDAP-HAM project consists of an LDAP server running on a Sun Spare, this database will be populated and updated from the KC database. Additional sources of information may also be used. Once everything is in place, automatic updating from the FE database is planned on at least a weekly basis. LDAP supports database replication for optimizing performance. For example, if you have 2 large campuses, both needing high access to the LDAP server you may want to have a second LDAP server at the second campus because of limited bandwidth or because of intermittent down time. LDAP fully supports a non-real time replication system that allows one server to be the master and others to be replications.

My initial use for the LDAPHAM server will be in augmenting email capabilities of APRS, specifically the APRS to email gateway (wu2z.rutgers.edu). Previously when the email gateway sent a message from APRS, the return address was alwalys the senders callsign "@unknown.net". For example, if I send a message to someone, the FROM address would show up as kb2iciQunknown.net. This is because there is no knowledge of a persons email address within APRS. By having the LDAP server, it will be real easy for the email gateway to execute a quick lookup and get the actual email address for KB2ICI and then substitute that address (which is kb2iciBamsat.org) in place of the kb2iciQunknownnet.

Server:

The LDAP-HAM server is running slapd from the OpenLDAP Foundation (http://www.openldap.org/). OpenLDAP is full open source and runs on many platforms. There are other LDAP servers available commercially from Netscape and from Microsoft.

Web access:

There will need to be a simple access to the data for casual lookups and there will need to be one or more methods for updating changing information, primarily email address. This is implemented with an apache web server and php (http://www.php.net).

Access by others:

One of the key goals of this project is for other developers, both traditional application developers and web developers, to be able to have simple and free access to the server. Modification of the data on the server will of course be password protected but read access will be open to anyone.

Sources:

The LDAP server, slapd, is open-source and is available from the OpenLDAP Foundation. The same goes for the replication server, slurpd

The web interfaces are written in PHP, most or all of these sources will be made available, contact the author.

References:

LDAP	Lightweight Directory **Access** Protcol http://www.umich.edu/-dirsvcslldapi
OpenLDAP	OpenLDAP Foundation http://WWW.openldap.org
PHP	Hypertext Preprocessor http://wwww.php.net/

<APRSdec>™ - The G3NRW APRS Packet Decoder

by Ian Wade, G3NRW (email: g3nrvQtaprorg)

7 Daubeney Close, Harlington, Dunstable, Bedfordshire LU5 6NF, United Kingdom.

4 July 2000

Abstract

This paper describes the principal features of the <APRSdec> program.. The program fully decodes raw APRS packets, producing reports in plain English. It has proved to be an extremely useful APRS diagnostic and learning tool. <APRSdec> is written in Perl, and runs under native DOS, Windows, Unix and Linux. It is available from **http ://ww.tapr.org/-g3nm**

Introduction

<APRSdec> started life towards the end of 1999, when I was working on the APRS Protocol Specification. I needed a simple tool to decode the many different types of APRS packet seen on air, to help me verify that I understood all the formats. Later, in the spring and early summer of 2000, the program unexpectedly came into its own as a diagnostic tool, when floods of badly corrupted packets appeared on the APRS network (arising from a combination of broken APRS client software and broken IGate server software). <APRSdec> was invaluable in reporting the errors and providing clues towards identifying the culprits.

<APRSdec> is a particularly useful tool for:

- APRS software development and testing.
- PIC development and testing.
- Network fault-finding and diagnostics.
- Learning about the APRS protocol.

<APRSdec> Features

Runs under native DOS, Windows 95/98 and LinuxAJnix. (It will almost certainly run under Windows NT and Windows 2000 as well, but this has not been tested).

Accepts raw input in TNC/IGate terminal output format.

Accepts raw input in UI-View two-line log format, and decodes the 15-digit UI-View timestamp.

Performs rigorous format checking, with detailed error reporting.

Understands position ambiguity, reporting the bounding box in which the station is located.

Compares lat/long position against a prefix/country database - if the station's position appears to be outside the country, <APRSdec> reports a possible anomaly.

Reports data values in imperial, nautical and metric units.

Fully decodes Mic-E and compressed position formats.

Recognizes data from Kenwood TH-D7 and DM-700 radios, and changes the incorrect DM-700 APRS Data Type Identifier to "Current Mic-E Data".

Provides detailed weather station reports, including the calculation of windchill and dew point.

Decodes storm data.

Decodes bearing and range data.

Decodes DX Cluster reports, showing the data as it will appear on TH-1~7 and DM-700 screens.

Some Examples of <APRSdec> Output

A simple lat/long report:

```
------------------------------------------------------------
Record #170
KE4SZT>APRS,W4NHV-9,WIDE~,WIDE/2:~27O7OOz34O8.76N/O7924~6OW- John in Marion, SC-793-
 APRS Data Type= Posit w/ time. With APRS
 Day= 27 Time= 07 hours 00 mins UTC
 Lat= 34 deg 08.76 min N  Long= 79 deg 24.60 min W
 Icon= House QTH VHF   Overlay= (none)
-w-w-- ------------------------------------------------------
```

A Mic-E report. CAPRSdew recognizes this comes from a Kenwood TM-D700 radio and automatically corrects the APRS Data Type to “Current Mic-E data”:

```
------------------------------------------------------------
Record #8051
KC7EQL>VE7VAN-3~~>WIDE3-1>T8PWPW:'37]lIe-/]~SS~Mark at Home awaits Sunsat
 APRS Data Type= Current Mic-E data
 Radio= Kenwood TM-D700
 Message Type= /M2/In Service
 Lat= 48 deg 07.07 min N Long= 123 deg 27.65 min W
 Icon= House QTH VHF   Overlay= (none)
 Course= 173 deg   Speed= 4 knots (4.6 mph 7.4 kph 2.1 m/s)
 Altitude= 397 feet (121 meters)
-------------------------------- --------------------------
```

A compressed lat/long position, within an APRS Object report:

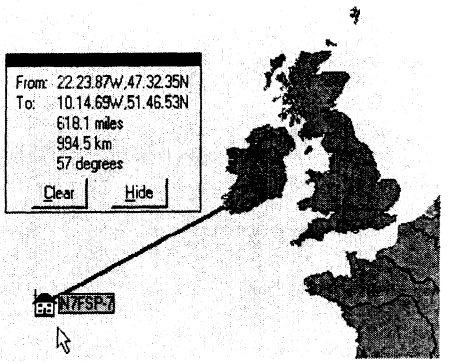
```
------------------------------------------------------------
Record #1763
N9IDH>APRK11,AE9A-10~~,WIDE3-2:;AO16      ,111826z\%H+!3vA!Sk~!R=05280/437.051/13Khz/APRStk
 APRS Data Type= Object
 Object Name= 'A016      ' Object Status= Killed
 Day= 11 Time= 18 hours 26 mins UTC
 Lat= 81 deg 14.20 min N  Long= 14 deg 04.30 min W
 CPS Fix= Old (last)   NMEA Source= Other   Compression Origin= Compressed
 Course= 296 deg   Speed= 1018 knots (1171.5 mph 1885.3 kph 523.7 m/s)
 Icon= Satellite/PAC   Overlay= (none)
-e--w-- ------------------------------------------------------
```

A Long Way from Home

This UI-View report (complete with the decoded UI-View timestamp) shows APRS Working Group Chairman John Ackermann, N8UR, jogging towards G3NRW’s home at over 72mph -- he was on a train at the time! cAPRSdec> recognizes that John is outside the contiguous 48 states, so it questions the reported location:

```
------------------------------------------------------------
Record #622
N8UR-7"~>RELAY>WIDE>WIDE>WIDE2-2>UlUVXX (UI):'vSf"> [/>"58)
 UI-View Timestamp= 27 May 2000   12 hrs 58 mins 32 sets (PC clocktime)
 APRS Data Type= Current Mic-E data
 Radio= Kenwood TH-D7
 Message Type= /M2/In Service
 Lat= 51 deg 56.88 min N Long= 0 deg 29.74 min W
;kQQUESTION: Is this lat/long position reasonable?
            It seems a long way from home for this callsign.
 Icon= Jogger   Overlay= (none)
 Course= 1 deg   Speed= 63 knots (72.5 mph 116.7 kph 32.4 m/s)
 Altitude= 407 feet (124 meters)
----------------------- ------------------------------------
```

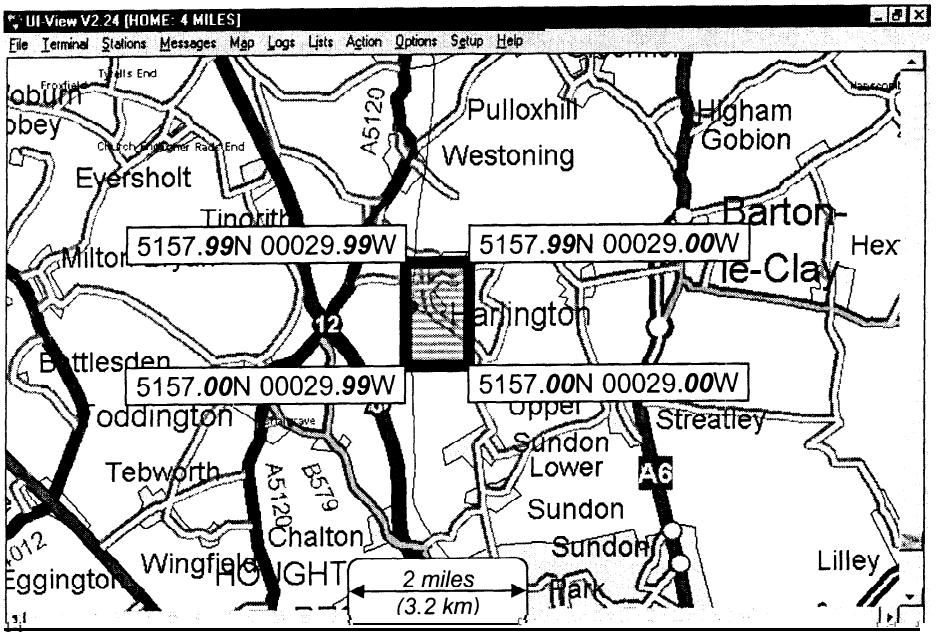
The next report is an example of how data was corrupted somewhere in the APRS network - the leading longitude digit was somehow changed from 1 to 0, making N7FSP-7 appear in the Atlantic Ocean, 600 miles off the southwest coast of Ireland:




```
Record #646
N7FSP-7>APK101,N70EP-lO~~,WIDE3-2:@111816z4732.2lN/O2222.65W-
                                   OOO/OOO/Mic-E/MO/Off  duty>
 APRS Data Type= Posit w/ time. With APRS
 Day= 11 Time= 18 hours 16 mins UTC
 Lat= 47 deg 32.21 min N Long= 22 deg 22.65 min W
%k*QUESTION: Is this lat/long position reasonable?
            It seems a long way from home for this callsign.
 Icon= House QTH VHF   Overlay= (none)
 Course= (Indeterminate) deg   Speed= 000 knots (0.0 mph
                                        0.0 kph 0.0 m/s)
-------------------------------------------------------------
```

Position Ambiguity

Position ambiguity allows a station to specify its approximate rather than absolute position. One way to do this is to replace coordinate digits with spaces. For example, G3NRW's home to the nearest minute of latitude and longitude could be specified as 5157.,-N and 00029.,,W (where u represents a space). This means that the latitude may be anywhere between 5 l° 57.OCJ and 5 l° 57.99' N and the longitude may be anywhere between 000' 29.00' and 000' 29.99' W. In other words, G3NRW is located somewhere inside the bounding box shown below: --



<APRSdec> understands this position ambiguity, and reports the coordinates of opposite comers of the box:

```
Record #l
C3NRW">APRS:!5157.  N/00029.  W-
 APRS Data Type= Posit w/o time. No APRS
 Ambiguous position. Opposite corners of bounding box:
  NW Corner: Lat= 51 deg 57.99 min N Long= 0 deg 29.99 min W
  SE Corner: Lat= 51 deg 57.00 min N Long= 0 deg 29.00 min W
 Icon= House QTH VHF   Overlay= (none)
-_____-____-_--c__-------------------  -------------I----- --m-
```

<APRSdec> similarly reports a bounding box for a Maidenhead grid locator:

```
------------------------------------------------------------
Record #459
CT1DBH-1>ID:[IM58IS]
 APRS Data Type= Maidenhead Grid Square
 Maidenhead locator bounding box:
  NW  Corner: Lat= 38 deg 47.50 min N Long= 9 deg 20 min W
  SE Corner: Lat= 38 deg 45.00 min N Long= 9 deg 15 min W
*"'NOTE: This Maidenhead format with square brackets is obsolete.
------------------------------------------------------------
```

Here is a full WX station report, with computed windchill and dewpoint:

```
-----------------------------------  --------------------
Record #SO34
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 APRS Data Type= Posit w/ time. With APRS
 Day= 11 Time= 19 hours 55 mins UTC
 Lat= 44 deg 47.06 min N Long= 93 deg 29.48 min W
 Icon= WX station   Overlay= (none)
 Wind Direction= 125 deg   Speed= 008 knots (9.2 mph 14.8 kph 4.1 m/s)
 Gust Speed= 9 mph (14.5 kph 4.0 m/s 7.8 knots)
 Temp= 79 degF (26.1 degC)   Windchill= 76.5 degF (24.7 degC)
 Rain: Last hour= 0 in (0.0 mm)   Last 24 hrs= 0.03 in (0.8 mm)
       Since midnight= (Indeterminate)
 Humidity= 41 percent   Dew Point= 53.2 degF (11.8 degC)
 Barometric Pressure= 1019 mbar/hP
--------------- -s-m-----------------------------------------
```

Finally, here is an example of a DX Cluster report, showing how the data appears on Kenwood radio displays:

```
-----------I-------- ---------------------------------------
Record #3852
NA4V-3>RESORC,AB4KN-2:k,WIDE3-2:DX de NA4V-3   >FO29@1509 FO20@1503                                    SATS
 DX Cluster Information:

   TH-D7 Screen 1     TH-D7 Screen 2              TM-D700
 +---------------+  +---------------     +---------------------------+
 1 Dx:FO20@1503 1 1 Dx:FO20@1503  ;    1 l:FO20@1503 FO29@1509 SATSI
 1  FO29@1509  I  I                |    |                           |
 |          SATS I 1               |    |                           |
 +---------------+  +--------------+    |                           |
                                        +---------------------------+

------------------------------------------------------------
```

Input Files

cAPRSdec> understands input data in regular TNC/Igate terminal format: e.g.

```
W9ZCU>APR846,WA4WHD-3+~,WIDE/V:=2559. N/08010. W-PHC6560/Carl's Castle!
```

and in two-line UI-View log format: e.g.

```
36673.5168518519~M1DTA>MB7UPG>GlYFF~c>T~CE3-1>APRS   [UI]:
=5143.43N\00035.88EUM1DTA Bob on Danbury Hill. Email address: mldta@yahoo.com
```

To obtain raw data from an IGate, you can telnet to it; e.g. telnet www.aprs.net. However, many APRS report lines are much longer than the 80,character line limit of most telnet clients. These lines will be split, so that some reports may occupy 2 (or even 3) separate lines. It is possible to join them up again using a text editor, but that is a long, tedious process!

Much better is to use a telnet client that can read long lines without wrapping or truncating. By far the best I haveseenis `~era~erm~~~,frOmhttp://hp.vector.co.jp/authors/VAOO2416/teraterm.html.`