APRSstat
An APRS Network Analysis Tool

Richard Parry, P.E., W9IF
rparry@qualcomm.com
http://people.qualcomm.com/rparry

ABSTRACT

APRSstat monitors network traffic by connecting to an APRS telnet server and collecting network traffic statistics. The program is intended to run continuously as a background task on a UNIX/Linux system. It collects and saves network data for periods as long as a year. The data gathered by APRSstat is plotted using the companion program APRSgraph, which creates graphs as GIF images allowing them to be integrated into a web page. APRSgraph is executed as a cron job every five minutes providing near real time updates of network usage for daily, weekly, monthly, and yearly periods. Characters per minute is used to measure network traffic. A detailed example of the San Diego, California APRS network is included which is displayed on the author's home web page. A cursory explanation of the software is also included in the paper. APRSstat and APRSgraph were written in perl and use perl modules GD and Chart for creating the graph GIF images.

KEYWORDS
Packet Radio, APRS, Linux, UNIX, Networking, Perl

INTRODUCTION

The Automatic Position Reporting System* (APRS) is one of the most popular new facets of amateur radio today. It joins a list of amateur radio specialties that is both long and broad in scope.

Although APRS can consist of a single transmitter and a remote receiving station, what makes it particularly useful is the network. This network allows stations that would normally be limited to line-of-site distances to extend the range to large metropolitan areas. Using HF (High Frequency) communication can be extended still further and when connected to the Internet, APRS networks become global.

A network is a complex subsystem that should be monitored and analyzed to spot weaknesses and/or areas that need to be improved. Commercial tools known as *sniffers* abound for monitoring various types of networks such as Ethernet networks. WinAPRS supports fairly extensive network monitoring functionality including graphs and limited statistical traffic usage. It is the purpose of APRSstat to provide additional APRS LAN traffic information. Its major strength is its ability to run continuously as a background task to collect, save, and display long term APRS network traffic. Let's look at a real life APRS LAN to understand exactly how it works and the information that it provides.

First we need to understand what we are measuring, so let's define what we mean by APRS network traffic. For our purposes, network traffic shall be measured as the number of characters received per minute. Note that characters, not packets, must be used to measure network traffic since APRS packets vary in length which makes comparisons of channel usage measured by packets impractical and less useful.

APRS traffic is transmitted at 1,200 baud, therefore a single bit requires 0.833 ms (1/1200) to send. Since an ASCII character is 10 bits, the time to transmit a single character is 8.33 ms. This translates to 120 characters per second or 7,200 characters per minute which represents the ideal theoretical maximum capacity of the channel[2]. Knowing the maximum channel capacity provides us with a basis for comparison and a metric that describes how close we may be to saturating the network.

The network usage statistics that APRSstat collects are displayed as four graphs: daily, weekly, monthly, and yearly. The wide range of time periods allows short and long term tracking of the network which can then be used to spot trends.

Figure 1 shows the daily network traffic for the author's private[3] telnet server in San Diego, CA. It displays the number of characters transmitted for a 24 hour period. The vertical axis displays characters per minute. For the daily graph, the sampling period is 5 minutes. This value is primarily a compromise between providing useful traffic resolution and limiting the size of the data log. Note that the most recent data point is on the left of the graph and the oldest data is on the right. As time passes, data moves from left to right and eventually off the graph.
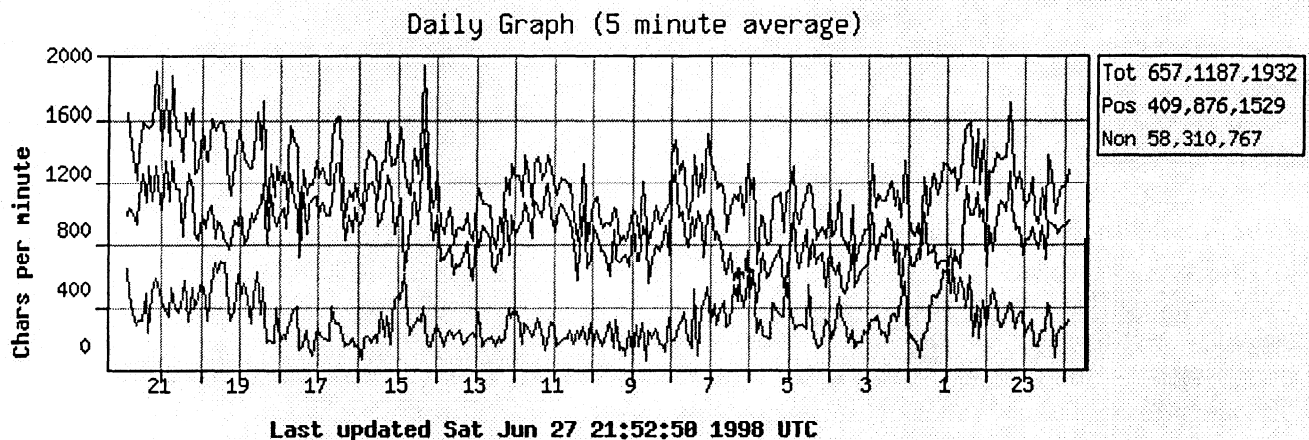

Figure 1 San Diego, Daily (24 Hour) APRS Network Traffic

To provide the most useful information, APRSstat breaks traffic into two categories: cnaracrers contained in *posit* and *non-posit* packets. A posit packet is an APRS packet that contains position information (i.e., latitude and longitude), while non-posit packets are those that do not provide position information. This is an important distinction. Although APRS was originally intended to provide

---

[2] This value is an approximation since APRSstat counts payload characters only. The payload is the portion of the packet carrying user information. An AX.25 packet encapsulates the payload in a header and trailer which both require bandwidth not measured. For this reason, actual channel capacity is less then 7,200 cpm and varies with the length of the packet.

[3] This private telnet server is behind a firewall and therefore cannot be reached outside the firewall. However, the results (graphs) are available at http://people.qualcomm.com/rparry/aprsstat.

position information exclusively, it now supports other types of communication. For example, it allows keyboard to keyboard communication. It also allows the transmission of weather information including: rainfall, temperature, wind direction and speed. When displayed on a web page, the graphs are color coded to help differentiate the types of traffic. Non-posit traffic is displayed in green, posit traffic in blue, and the total of the two is plotted in red.

Non-posit traffic is the lowest data set on the graph and therefore represents the least amount of APRS network traffic. Non-posit characters vary from a low of **58** to a high of 767 characters per minute (cpm) with a mean value of 310. It is worthy to note that although non-posit packets represent the least amount of network traffic, they are substantial, representing approximately 26% (310/1,187) of network traffic.

Posit characters represent the remainder of APRS network traffic. They are displayed as the middle data set on the graph. Posit characters vary from a low of 409 to a high of 1,529 with a mean of 876 cpm. Therefore posit characters account for 74% (876/1,187) of network traffic.

The data set plotted at the top of the graph is the total of both posit and non-posit packets. In the example, total characters vary from a low of 657 to a high of 1,932 with a mean of 1,187 cpm. We can now compute network traffic usage as 16.5% (1,187/7,200). For *connection* oriented protocols that support collision and/or corrupted packet detection, the network would be considered lightly loaded. Unfortunately, an APRS network must remain lightly loaded since it uses a *connectionless* protocol consisting of UI (Unnumbered Information) frames. This is just another way of saying that if a packet is dropped, there is no means to detect the loss and therefore no means to retrieve it. To drive home the point, assume an APRS network is 50% loaded. Under these circumstances a transmitted packet has a 50% probability of colliding with another packet[4] making the network almost unusable.

Figure 2 shows weekly APRS traffic for the same San Diego, CA. network. The same information is plotted, but for a longer period. This allows one to spot trends that may occur weekly. For example, there might be more traffic on weekends than during the week or more traffic during the daytime hours than at other times. The sampling period is 30 minutes for the weekly graph. An analysis shows that non-posit traffic accounts for 25% of the total network traffic. Posit characters account for the remainder, or 75%. The most important metric is total network usage which in this case is 16.67%.
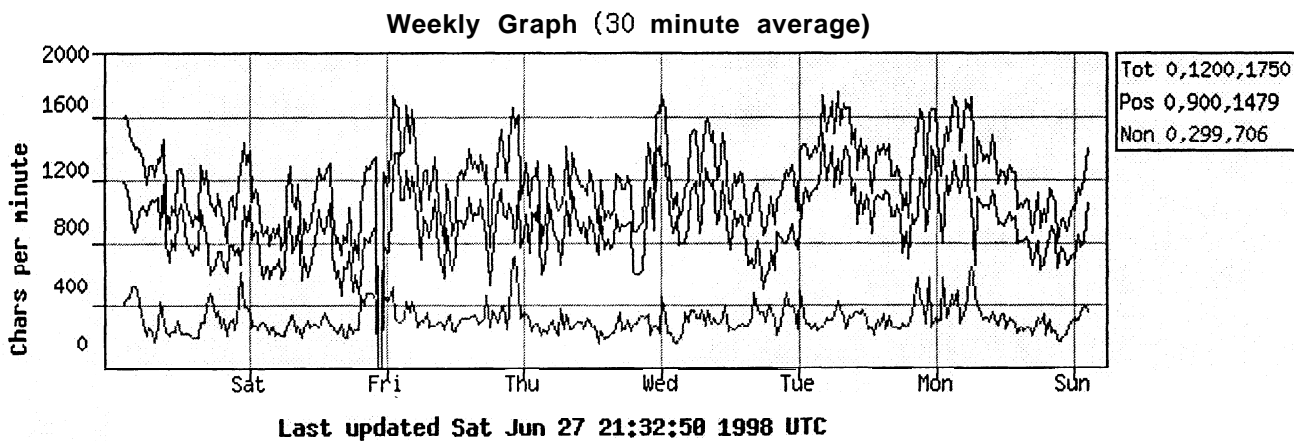

Weekly Graph (30 minute average)

Tot 0,1200,1750
Pos 0,900,1479
Non 0,299,706

Last updated Sat Jun 27 21:32:50 1998 UTC

Figure 2 San Diego, Weekly (7 Day) APRS Network Traffic

---

4   This is not completely true since the length of the packet will also determine success or failure, however the statement is sufficiently accurate for our needs.

It is important to note that the percentage of network usage by posit and non-posit traffic remains virtually constant between daily and weekly periods. This should not be a surprise since the only difference between daily and weekly data is the sampling period. Computing traffic usage for monthly and yearly periods continues to show that non-posit traffic accounts for approximately 25% of all network traffic and posit packets for 75%.

Figure 3 depicts monthly traffic. Posit and non-posit traffic for each day of the week is plotted. The sampling period is extended to 2 hours. The monthly graph allows one to spot trends that occur on a weekly basis.
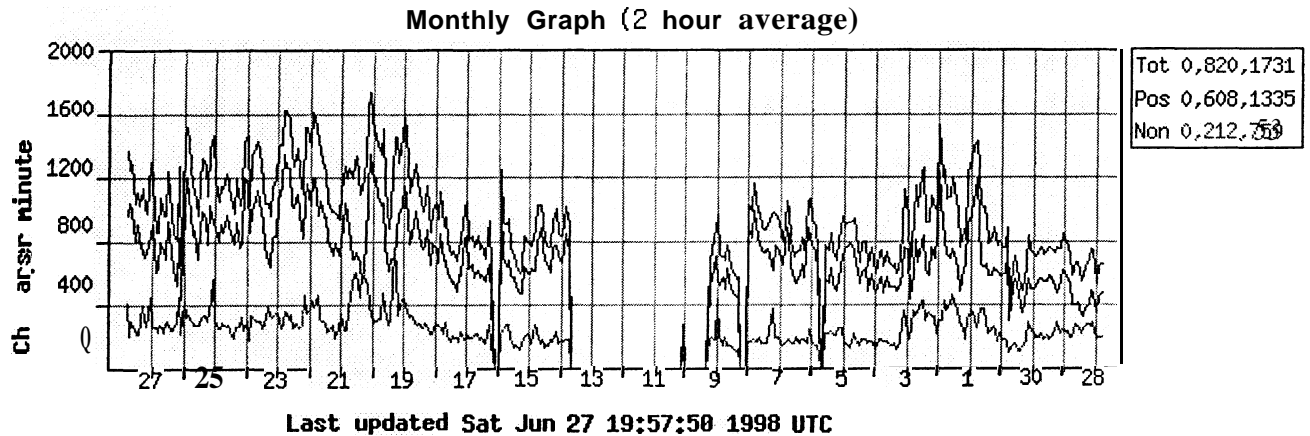

Figure 3 San Diego, Monthly (31 day) APRS Network Traffic

Figure 4 provides network traffic usage for a year. As previously mentioned, to limit the size of the log, the sample rate is extended to 24 hours. The graph begins in March since that is when APRSstat was completed and first came on line. It has been running continuously with brief periods of downtime, such as the loss of data from June 9 to 14, shown in both Figure 3 and 4. The yearly graph allows spotting long term trends that occur monthly.
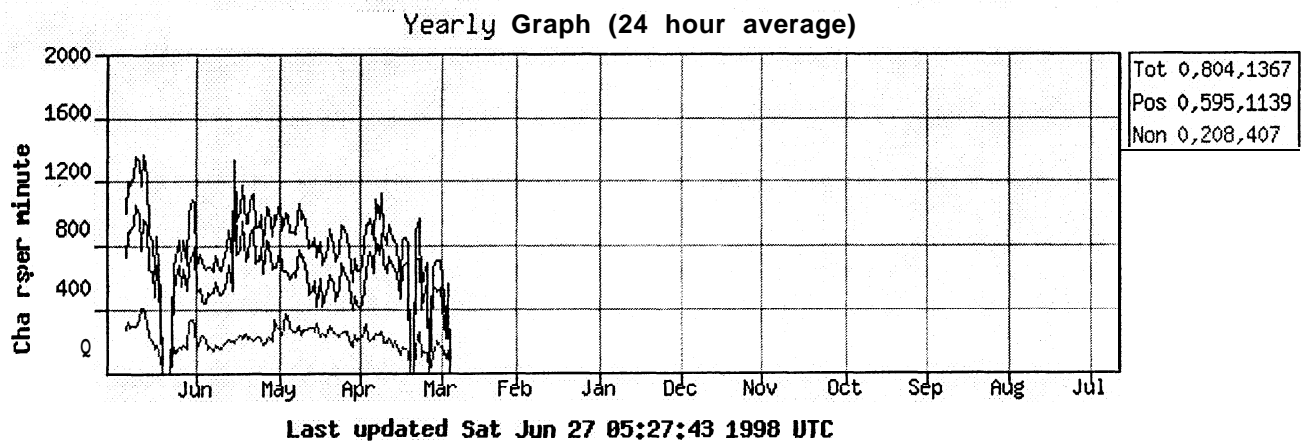

Figure 4 San Diego, Yearly (12 Months) APRS Network Traffic

74

WHAT DOES IT ALL MEAN

The purpose of APRSstat is to provide data in a form that is easily analyzed, and ultimately to draw useful conclusions. An examination of the graphs allows us to draw the following conclusions for the San Diego network:

1. Total channel usage is approximately 16%. It is measured by using the mean cpm and dividing by the total theoretical maximum capacity of the channel.
2. Non-posit packets represent a significant portion of APRS traffic, approximately 25%. The remaining 75% of traffic on the network comes from packets with position information.
3. It is interesting to note that daily network traffic is relatively constant. One might think that network traffic would vary with the time of day. For example, more traffic during daytime hours and less at night. However, when one realizes that most traffic contains information that is automatically transmitted (no human intervention) the conclusion makes sense. For example, position beacons are transmitted at 30 minute intervals both day and night. The same is true for packets carrying weather information. Only mobile tracker beacons and keyboard to keyboard packets are more prevalent during daytime hours. These packets are initiated manually, and therefore account for a minority of the network traffic.
4. Weekly and Monthly traffic is also relatively constant for the same reasons described above. For example, as the weekly and monthly graphs show, there is no significant increase in network traffic on weekends.
5. There is insufficient data for yearly traffic analysis since we do not have a full year's worth of data. However, with nearly 4 months of data, network traffic appears to be relatively constant. There are some monthly variations but no clear pattern is discernible. As APRS networks continue to grow, the predicted pattern should show a steady rise.

Similar results are expected for other metropolitan areas.

APRSSTAT INTERNALS

APRSstat is one of two programs that make up the system to collect and display APRS network traffic usage. It is a perl program5 that runs as a background task under Linux, but any UNIX system should work. As we shall see in the next section, APRSgraph is a separate task that also runs in the background.

APRSstat begins with a telnet connection to an APRS server. In the examples shown in this paper, the connection is to the author's San Diego, CA, APRS telnet server. Once the connection is established, the program starts receiving APRS packets and counts the number of characters contained within the packet. It then determines the type of packet (posit or non-posit) and increments separate posit and non-posit character counters. At five minute intervals, posit and non-posit counters are time and date stamped and saved to a log file. The counters are reset to zero and the process begins again.

The log file consists of four mini-logs, one for each time period (e.g., day, week, month, and year). The first section of the log file contains the daily log data (5 minute intervals), the second section holds the weekly data (30 minute intervals), the third section contains monthly data (2 hour intervals), and the last section of the log retains the yearly mini-log (24 hour interval). The reason for the different resolutions is primarily a compromise to limit the file size.

A status log file is also provided and updated as necessary to log salient events such as when the process began and the loss of a connection to the server.

---

5   A perl library to be more exact.

APRSGRAPH INTERNALS

APRSgraph is a separate program and runs independently of APRSstat. While APRSstat is collecting network statistics and storing it in a log, APRSgraph is asynchronously reading the log and creating graphs on the fly. APRSgraph is executed as a cron job6 every five minutes resulting in new GIF images which provides near real time display of network traffic.

When APRSgraph is executed, it opens the log file created by APRSstat and extracts the daily, weekly, monthly, and yearly mini-log data. Using perl modules GD.pm and Chart.pm, four GIF images are created. A web page is then used to display the information.

CONCLUSION

Networks provide the ability for two or more entities to communicate. While the entities themselves may be complex (a computer), the network can be equally complex. Understanding how the network operates is important to insure reliable communication. APRSstat was developed to provide a means to monitor APRS network traffic and ultimately to understand its behavior to allow it to be improved.

The examples provided in this paper were limited to a single APRS server located in San Diego, CA. Limited connections to other servers in metropolitan areas with significant APRS traffic show similar trends and conclusions as those outlined in this paper. It is the hope of the author to continue monitoring APRS networks as APRS activity continues to grow among amateur radio operators.

ACKNOWLEDGMENT

Thanks to Bob Bruninga, WA4APR; for permission to use the APRS trademark.

RESOURCES

1. Bruninga, Bob, "Automatic Packet Reporting System (APRS)," 73, December 1996, pp. 10-19.
2. Dimse, Steve, "javAPRS: Implementation of the APRS Protocols in Java," *ARRL and TAPR 15th Digital Communications Conference Proceedings,* Seattle, Washington, September 1996, pp. 9-14
3. Parry, Richard, "Position Reporting with APRS," *QST,* June 1997, pp 60-63.
4. Parry, Richard, "APRS Network Guidelines," 73, October 1997, pp 19-20.
5. Parry, Richard, "Position Reporting: The Global Positioning System and Linux," *Linux Journal,* July 1998, pp 46-49.
6. Parry, Richard, "XNET: A Graphical Look At Amateur Radio Packet Networks," *ARRL and TAPR 15th Digital Communications Conference Proceedings,* Seattle, Washington, September 1996, pp. 64-75.
7. Parry, Richard, "perlAPRS," *ARRL and TAPR 16th Digital Communications Conference Proceedings,* Baltimore, Maryland, October 1997, pp. 141-148.
8. Parry, Richard, "Graphing on the Fly: Generating Real Time Graphs for the Web," available from the author's web page at http://people.qualcomm.com/rparry/papers.

---

6   A cron job is a computer task that is executed on a UNIX system at specified intervals.