Performance Monitoring
-or-
"I Wanna Fix It, Is It Broke?"

Skip Hansen, WB6YMH
Harold Price, NK6K

Abstract

Much of the performance informati.on on Amateur Packet Radio is anecdotal and ephemeral; a subjective and non-detailed accou.nt usually limited to a gross statement of "goodness" or "badness", which is neither well documented nor long remembered. While there are several papers which describe the expected performance of CSMA-type systems, there is little actual data about the live amateur packet system.

The authors discuss the need for accumulating performance data and describe work in progress to supply performance measurement software using a C program and a TNC with KISS software.

1. Why Performance Monitoring?

Big changes are coming in amateur packet radio. In early 1987, most of the amateur packet network was based exclusively on AX.25 and digipeaters. By the end of 1988, if not sooner, much of the packet world will be made of up a conglomeration of NET/ROM, TEXNET, TCP/IP, and other systems interconnecting 40,000 AX.25 based users. Each will be implemented and installed by packeteers eager to make the network better than it was before.

Each system contains a myriad of trade-offs and compromises. Each system has several tuning knobs which can be used to modify the way it operates, affecting both local user performance and global network performance. In many cases, these knobs will be cranked by people with no data on how things are running and therefore no way to tell if anything got better. In other cases, the knobs will be tuned to optimize local performance, to the undetected detriment of the rest of the network.

Performance data is vital to a local network. It is needed before the current network can be tuned, and it should be available to those who will help specify the next network. Put simply, if you don't know what you have now, how will you know if what you get next is any better?

1.1 We've Already Missed One Chance.

We've already missed one chance to monitor a major change, and in California, we've missed a second. The first version of AX.25 did not use the Poll/Final facility of LAPB. In that version, if an acknowledgment of a data frame was not received, the data frame was re-transmitted. If multiple data frames were outstanding, only the first one was re-sent. In the second version of AX.25, the poll/final facility was implemented. In AX25v2, if a data frame is not acknowledged, a "poll" is sent out, soliciting a new acknowledgment. If that ack does not indicate that the data frame was received, the data frame is then retransmitted, otherwise transmission continues with new data frames.

Any change to a protocol like the one described above entails some cost. Whether it is the effort involved in updating and distributing new software, or the trek to a snowed in mountaintop to swap ROMS, some of our limited people resources are expended. In the poll/final update, was a improvement in network performance obtained that in some way offset the effort involved in implementing it and updating the user base?

Unfortunately, we'll never know. Since there was no network performance data before the change, and none was taken after, there is no way to tell. Our only indication is indirect; one of the original major proponents of the change to poll/final is now suggesting that poll/final not be used in some cases. [1]

For future changes, we must do better.

1.2 NET/ROM

In California, the old digipeater backbone which connected northern California, southern California, and Arizona has been largely supplanted by NET/ROM nodes. We had no data showing the performance of the old system, and we have no data on the performance of the new system. It is therefore difficult to measure the improvement.

2. Field Experience vs. Theoretical Predications.

There is a large amount of literature on the topic of packet switching systems, and on packet radio. Some is quite accessible to the average amateur, one networking textbook in particular, by Tanenbaum [2], has been cited so often that it is stocked by local amateur radio stores. There is little written, however, on packet as it is practiced in the amateur radio world. In most cases, if you notice the discussion leaning toward the way we do it, you find it given as an example of the wrong way. Actually, the word "wrong" is seldom used, "less optimal" is more common.

Much of the non-amateur networking experience of those who make up the amateur packet radio community is in the area of local area networks (LANs). Although there are a great many common problems and solutions between commercial LANs and the amateur packet network, there is a danger in assuming calculated performance parameters for the former have relevance in the later. Unfortunately, there is a tendency, with a lack of actual data, to use predicted LAN data in design and implementation discussions as if it were gospel.

A LAN, as discussed in Tanenbaum [2] page 286, generally has three distinctive characteristics:

1. A diameter of not more than a few kilometers.

2. A total data rate exceeding 1 Mbps.

3. Ownership by a single organization.

Although (1) is of importance only as it relates to propagation delay for very high data rates, (2) and (3) are worthy of note. The standard data rate in 1987 is still 1200 baud. There are 56kbps modems being beta tested now, but that is still only 6% of 1 Mbps. Ownership by a single organization is also something that is unusual in the amateur radio network. Item (3) tends to lead to either a homogeneous set of network hardware, a common set of goals, or at least a common forum for discussing those items. In the amateur world, users and implementors in northern California, Southern CA, and Arizona don't get together very often. That's another advantage of (1), in Los Angeles, one node can cover a area with a diameter of 200 miles.

Many of the studies done on LAN performance make assumptions that are not valid in the amateur environment. One study, for example, from [2] page 289 assumed:

- All packets are of constant length
- There are no errors, except those caused by collisions.
- There is no capture effect.
- Each station can sense the transmissions of all other stations

None of these assumptions hold for our current environment.

Another difference between our network and more commonly modeled networks is in the large number of autonomous stations we have on the network, and the large number of different traffic patterns running simultaneously. During the three days that data was gathered for this paper, 371 transmitters were on the air at some time in southern California. The peak number of active transmitters on a single 1200 baud frequency in a single five minute interval was 42. Most modeled networks have higher baud rates and/or low data throughput, and assume traffic is moving between a large number of outlying stations and a central station.

Again, while there is much value in reading and modeling, we should make the attempt to measure what we have; both to feed the result back into the models, and to establish a base against which future modifications can be judged.

3. The Current State of Affairs.

There appears to be only one kind of monitoring being done in amateur packet radio today. The two most common BBS systems, by WORLI and by WA7MBL, both produce a log of BBS activities. An analysis program produces a report for the BBS operator of the number of connects from users and the number of messages forwarded, among other items. While this gives a BBS operator some idea of his local usage patterns, it does little to described total network activity, or even the throughput the BBS experiences.

For global network performance we are left with anecdotal evidence, e.g., "01 really stinks tonight" (translation: performance is less than expected), and "I had no problem with 01 today" (translation: I'm retired and was on at 10:00am).

For local user performance, we get "I can talk to Utah all night long", and "I haven't been able to connect up north all week". Obviously, we need something better.

4. It's Not Easy

There are two ways of looking at network performance, one is from the network's point of view, the other is from the user's point of view. In the first case we are interested in how the channel is performing, in the simplest view, how many bytes of data it is carrying. Is the network carrying a large number of user bytes, or is most of the capacity going to overhead or retries? Are we losing data to collisions, or to bad RF paths?

In the second case, the user's point of view, the questions are more toward what level of service an individual user is getting from the network. Is the response time from distant locations adequate? Do many connections time-out? Are some destinations unreachable due to congestions or path failures?

There are several ways of acquiring performance data. One is to have each user station collect it. As updating 40,000 user's is a non-trivial exercise, we've chosen another route. A specialized monitor station sits at a central place and looks at all the activity on the channel. Unfortunately, it isn't easy to answer any of the questions from a third party monitor station. Some of the problems are discussed below.

4.1 The Problem Is, It's Radio.

In most wire based, broadcast-type LANs, a monitor program can make the assumption that if it heard a packet, everyone else in the LAN heard the packet. More importantly, if it didn't hear a packet, no one else did either. Even if the LAN is relaying data between two other LANs, it is at least certain that for data originated on the LAN or destined for the LAN, the monitor has a high probability of having the seen the same data as the other stations on the LAN. In the amateur packet network, due to hidden terminals, the FM capture effect, and propagation, all stations do not hear the same packets.

If the monitor station heard all packets, it could easily follow the state of all connections on the LAN. For connection oriented protocols like AX.25 and TCP, and providing the monitor has been up as long as the other stations on the LAN, the monitor can tell how long a connection has been in place based on the circuit start and end protocols. In the amateur radio case, the monitor station can not be certain that it heard all packets. It may miss a circuit startup or end. It must instead be prepared to infer that a connection exists because it sees data flowing, or that a circuit has closed because it has seen no data for an interval of time. This will add uncertainty to data gathered in an RF environment but it does not invalidate the entire effort.

Although collisions can be directly detected on a wire LAN, they can not be as easily detected on radio. Due to the capture effect, a stronger FM station will completely override a weaker station such that stronger packet is received without error, even though two packets were being transmitted at the same time. A collision may be inferred if the received packet is seen again.

Some tasks then become exercises in gather as much information as possible, and then making an educated guess. Still, this is better than no data at all.

4.2 Users Are Easy to Replace.

It is somewhat easier to gather user oriented data, e.g., does a path to station X exist at this time, or what is the round-trip delay for packets between Los Angeles and Salt Lake City. The monitor station can actually be a user and directly measure these values.

While data can be gathered about the performance of the channel at a specific time in this way, this alone will not supply information about the global network status at the time the measurement was taken. To be able to draw a meaningful conclusion from the data, aside from variable X was equal to Y at time T, other information is needed, such as the number transmitters on the air, and the number of other packets on the channel. In sort, both types of monitoring must be performed, direct measurement of user performance and global network measurement.

5. Monitoring Software

The software currently under development by the authors addresses the problem of global network monitoring. Other types of monitoring will be added in the future.

In this early version of the software, we are attempting to determine what sorts of questions can be answered by a program which listens to a channel and takes note of the packets it hears. Some questions, such has how many total bytes are being received at the monitor site, how many transmitters are seen, how many beacons are heard, are easy to answer.

A much more difficult question is "How many times does the average forwarding BBS send a 20k file before it goes all the way without timing out?" The type of information we're collecting, and the type of questions that can be answered, are discussed below.

5.1 Questions to Answer

There are two basic questions which are reasonably easy to answer. One is "What is the efficiency of the channel", the other is "How many users does the channel support".

We have chosen to define efficiency as the ratio of the number of unique bytes of user data on the channel verses the total number of bytes on the channel. "Unique data bytes" is our term for actual user data not including frame overhead, retransmitted copies, or digipeated copies. For example, if the string "hello" is entered, digipeated once, not acked, retransmitted, redigipeated, and acked, the total number of bytes on the channel would be 168, the number of unique data bytes is 5, an efficiency of 2.9%. If 256 user bytes are sent and directly acked, the efficiency is 88%.

To keep statistics on each user of the channel, we store pairs of Source and Destination calls from the frame header. The pair is called a circuit. A normal two-way connection would consist of two circuits. If NK6K and WB6YMH were connected, one circuit would be (TO:NK6K,FROM:WB6YMH), the other circuit would be (TO:WB6YMH,FROM:NK6K). Statistics for each circuit are maintained separately.

In addition to two basic questions, we wanted to be able to determine the number of digipeaters the circuit used, what the average size of a data frame was, the number of RNR (input blocked) frames transmitted, and similar questions. Since this required looking into the control fields of the frame, the standard TNC interface was unsuitable.

5.2 KISS

We chose the KISS TNC interface to give us access to all fields of the frame. KISS sends the entire frame, minus the checksum, to the terminal port using an async framing format. The KISS interface has been implemented on the TAPR TNC 1, the TAPR TNC 2 and clones, and on the AEA PK-232. The KISS software for the TNC 2 is included with the KA9Q TCP/IP package.

There are no modifications required to the KISS code for use in this application.

5.3 Software Design

The current implementation of the monitor package consists of three programs.

- STATS.EXE - This program monitors the received frames and accumulates data, periodically dumping the data into a log file. STATS also displays the addresses, data, control fields, and a "retry" flag in real-time as frame are received. NET/ROM and TCP/IP control fields are also displayed.
- REPORT.EXE . This program massages selected data from the log file into a form suitable for passing to a plotting program. The plotting program is not included.
- AVERAGE.EXE - This program massages the output of REPORT, combining and averages the records into larger intervals of time. This can result in clearer plots.

5.3.1 STATS.EXE

STATS collects data over a five minute interval, storing it into several different tables. These tables are then written into the log file at the end of each interval, along with a time stamp record. The tables are summarized below.

Digipeater Data.

The total number of packets and bytes heard from a digipeater is stored, along with the call of the digipeater.

Frequency Data.

Totals on bytes and packets heard on the channel without regard to source are maintained. Packet are also counted by length into five buckets: 32, 64, 128, 256, and greater than 256 bytes. The total number of ticks of the 18.2 Hz clock when the data carrier detect (DCD) line was high are recorded, as are the number of ticks when , DCD was low.

Circuit Data.

Several items are stored for each circuit, or TO:/FROM: pair. This includes the number of digipeaters used, the Protocol ID Byte (PID) of the last I frame received in the interval, the total number of packets and bytes received, the number of unique packets and bytes received, and the number of packets and bytes ignoring those heard from multiple digipeaters. Also included is the number of unique frames heard of each frame type (sabm, ua, etc.), the number of frames with POLL, and the number of frames with FINAL. The number of I frames heard is also counted into

five buckets based on the data size: 32, 64, 128, 256, and greater than 256 bytes.

As an indication of the difficulty of accurately determining the status of a frame, the algorithm used to determine uniqueness is described below.

Uniqueness

Depending on the packet type one of three different algorithms are used to test for uniqueness.

I frames are judged to be unique if the N(s) variable matches the expected V(s), or if the locally computed checksum of the information portion of the current frame does not match the checksum of the last frame received with the same N(s). Note that the checksum is only used to resolve the ambiguity resulting from lost frames. An algorithm based solely on checksums would be confused easily by data streams containing identical consecutive lines. For example consider the transmission of text files containing multiple blank lines separating pages. In such cases several consecutive packets would contain identical information, a single carriage return, but still be unique.

S and U type frames are judged to be unique if the control field of the current frame is different than the control field the last S or U frame received. Note that this does not detect retries of frames such as multiple SABMs sent because the target station is not responding.

UI frames are judged to be unique frames if the checksum of the information field of the current frame is different than the checksum of the last UI frame which was received.

Digipeated frame filtering logic

The various "non-digipeated" counters in the software are designed to show the number of times a particular frame appears on the channel without regard to multiple retransmissions by digipeaters. The "non-digipeated" counters are advanced once and only once regardless of how many digipeater hops are observable by the monitoring station. This data is used to determine the number of retries of a packet without confusing a retry for a digipeat.

The software maintains bit maps of observed hops for its use in filtering out digipeated frames. A separate bit map of observed hops is maintained for UI, S and U frames types as well one bit map for each outstanding I frame. There are 9 bits in each map which correspond to the originating station plus up to 8 digipeaters.

A frame is considered to be "non-digipeated" when it is either heard for the first time or it is heard from a hop from which it had been previously heard. The first condition is met when a frame is first transmitted, the second condition is met when a frame is retransmitted successive times. If neither case is met the frame is a digipeated frame and is not used to increment the "non-digipeated" counters.

The digipeat bit map is cleared when either the uniqueness subroutine determines the frame is unique or when the digipeat filter subroutine determines that the frame is a retransmission.

5.3.2 REPORT.EXE

REPORT produces several output formats. The RAW format displays each field in each record. This is useful if a particular interval is being examined in detail, or when debugging STATS. Several other formats are used to produce data for plotting. One report totals all circuit data for an interval. Examples of this output are provided later.

5.4 Hardware

As discussed above in the section on KISS, TNC 1 and TNC 2 clones, and the AEA PK-232 can be used with this software. If the DCD ON and OFF times are desired, a jumper must be added.

DCD Jumpering

Since most of the current TNC designs use the DCD signal on the RS-232 interface as a connect status indicator it is necessary to modify the TNC hardware slightly to provide a true modem DCD on the RS-232 interface. The modification for the TNC 2 and clones is very simple, consisting of a single jumper wire. The jumper goes between pin 2 of the modem disconnect header (DCD output from the modem) and the pin of JMP1 which is NOT connected to +5 volts (input to the DCD driver). On the MFJ-1270B artwork the correct pin of JMP1 is the one closest to the front panel. The authors have not researched modifications to other TNC designs, but it is expected the modifications will be similar. It is NOT necessary to perform the DCD modification to run the monitoring software, it is only necessary if the statistics of DCD activity are desired.

Most terminal software used on packet will be unaffected by this modification, however most BBS software will require the jumper to be removed for normal operation.

The software was developed on an IBM PC/AT using Microsoft C 4.0. It should be easily transportable to other systems provided a suitable serial port interface is available.

A hard disk is highly recommended. Twenty-four hours of data for 145.01 MHz as monitored in southern California produced 500k bytes of log file data. This may be reduced, of course, by increasing the interval time.

6. Examples

We used the STATS program to acquire performance data on all of the active packet channels in southern California. The monitoring site used for 145.01 MHz was at 700 feet on Palos Verdes. On this frequency, the site can "see" 8 NET/ROM nodes. During the 24 hours during which 145.01 was monitored, from 00:00 to 23:59 local time on a Thursday, 105 total transmitters were seen.

The data shown in the sample graphs is based on the five minute interval data from STATS, which was then processed by REPORT. The output of REPORT was then averaged by AVERAGE into 15 minute samples. Each point plotted represents the average of three five minute intervals.

Figure 1 shows the number of user circuits seen in an interval. A "user circuit" is a subset of the total circuits; beacons, repeater ids, and other circuits consisting of a small number of UI frames have been removed. The peak of data just after midnight is caused by forwarded BBS traffic, large broadcast messages such as newsletters are restricted to being forwarded between 00:00 and 8:00 by local custom.

Figure 2 shows the total bytes per minute. Further analysis of the data would show the distribution of bytes in the peaks; how much is destined for local users, and how much is going "overhead", passing through the backbone to other NET/ROM locations. The major NET/ROM path through to Arizona is still on 145.01, this should change before the summer is out. It will be interesting to see what effect moving the backbone will have on this graph.

Figure 3 shows the efficiency of the channel, computed as discussed above.

Figure 4 is a plot of efficiency vs. the number of user circuits on the channel. The distribution on a plot of efficiency vs. total packets is similar to this one. The occurrence of low efficiency over the entire range of users (and number of packets) shows that there are causes of low efficiency other than congestion. One interpretation would be that more data is lost due to poor RF paths than to collisions. Another would be that hidden terminals are causing problems. Further analysis of the data, coupled with a knowledge of the geography and stations involved, might result in information that could be used to improve the network.

7. Other Uses / Future Goals

Once a basic set of data gathering tools and formats has been define, the applications are boundless. For example:, STATS can be used to make improvements to the current 14.109 HF forwarding scheme. For example, data. gathered during the day on Friday, July 29, shows that of the two top stations in. terms of the total bytes transmitted, shows that one had 30% better efficiency than the other. If monitoring was continued, and the trend continued over time, it may mean that the less efficient station is trying to reach stations beyond its range, or that there are local receiver problems. It also means that the monitor station was hearing more data frames from the transmitting station that the target station was, perhaps the mail between those stations should be re-routed.

STATS can be used to check propagation between the monitor station and other stations. Figure 5 shows then number of bytes received on 14.109 MHz in a 24 hour period. It can also be used to infer propagation between other station. For example, if you hear a station in Indiana sending packets to Seattle and the efficiency is high, then a path must exist between those two points, even if you do not hear packets from Seattle at the monitor site.

The "unique" subroutine can be used filter retries out of a monitored connection as the data of the connection is displayed. AEA offers a similar feature on some of its TNCs. STATS will be updated in the future to allow the capture of filtered text from each circuit into files for later review. This can serve several purposes, as a diagnostic aid, a periodic check for intruders on the amateur network as required by the FCC, or to satisfy the standard urge to "read the mail".

This type of data collection could also assist in message traffic analysis, e.g., how many bytes are in the average connection? Are most of the BBS messages forwarded on a channel destined for users in the local area or are they just passing through?

Currently, STATS monitors at the link-layer level. Higher layer protocols such as TCP/IP and NET/ROM add additional complications to traffic analysis, primarily in determining the actual

origination and destination point. Work remains to be done in this area.

8. Conclusion

There is much good to be gained from gathering and analyzing performance data. It can tell us where we are and suggest where we might go. It will also help determine if we like where we've gone once we get there. The work discussed here is a start toward developing tools to aid in this task. Others are invited to participate.

9. Availability

The software described in this paper is available in source form from the WB6YMH-2 BBS on 145.36 in southern California. This BBS is also available by phone for those not in the local area at (213) 541-2503. Updates will periodically be sent to the HAMNET BBS on Compuserve.

10. Acknowledgments

Thanks to Craig Robins, WB6FVC, for his help in the preparation of this paper. Thanks also to those who have implemented the KISS code for the TNC 1 and TNC 2, and the folks at AEA.

11. References.

[1] Karn, P., KA9Q, "Proposed Changes to AX.25 Level 2", informal paper circulated on various mail systems and reprinted in the July/August 1987 NEPRA PacketEar, the newsletter of the New England Packet Radio Association.

[2] Tanenbaum, A., "Computer Networks", Englewood Cliffs, NJ: Prentice Hall, 1981.

Fig 1. User Circuits

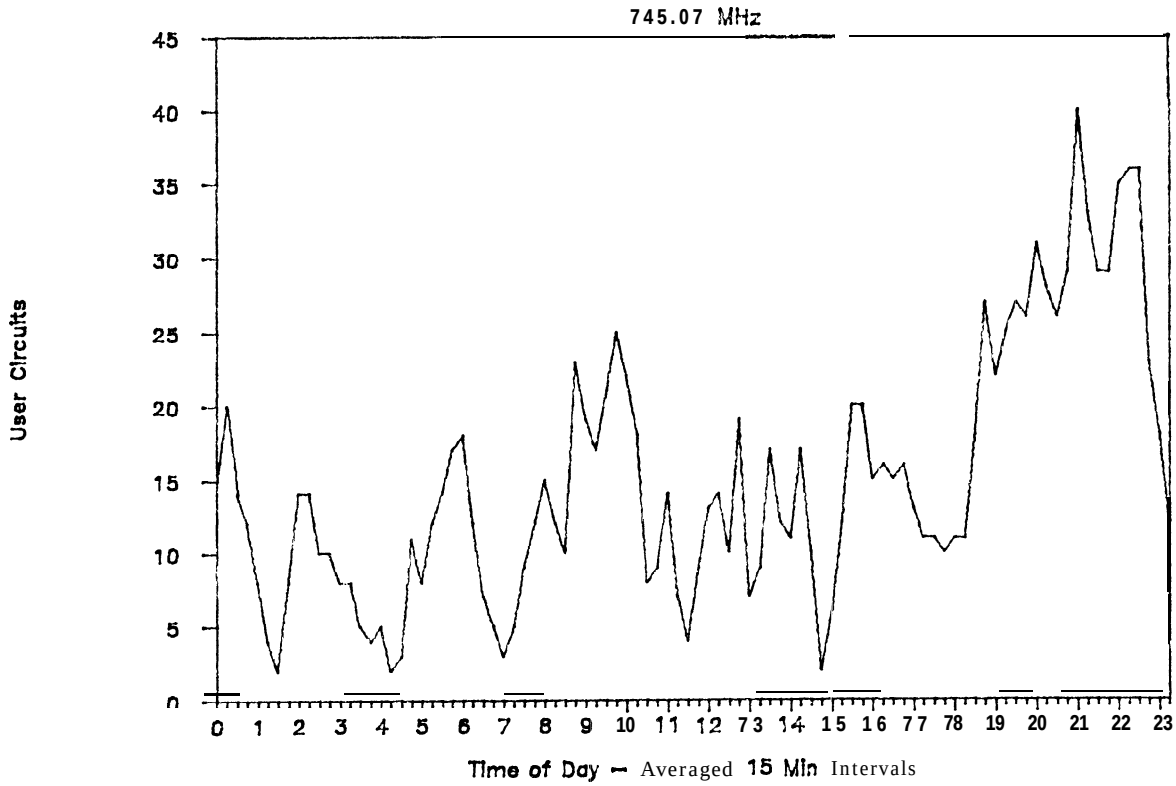

Fig 2. Total **Bytes per Minute**

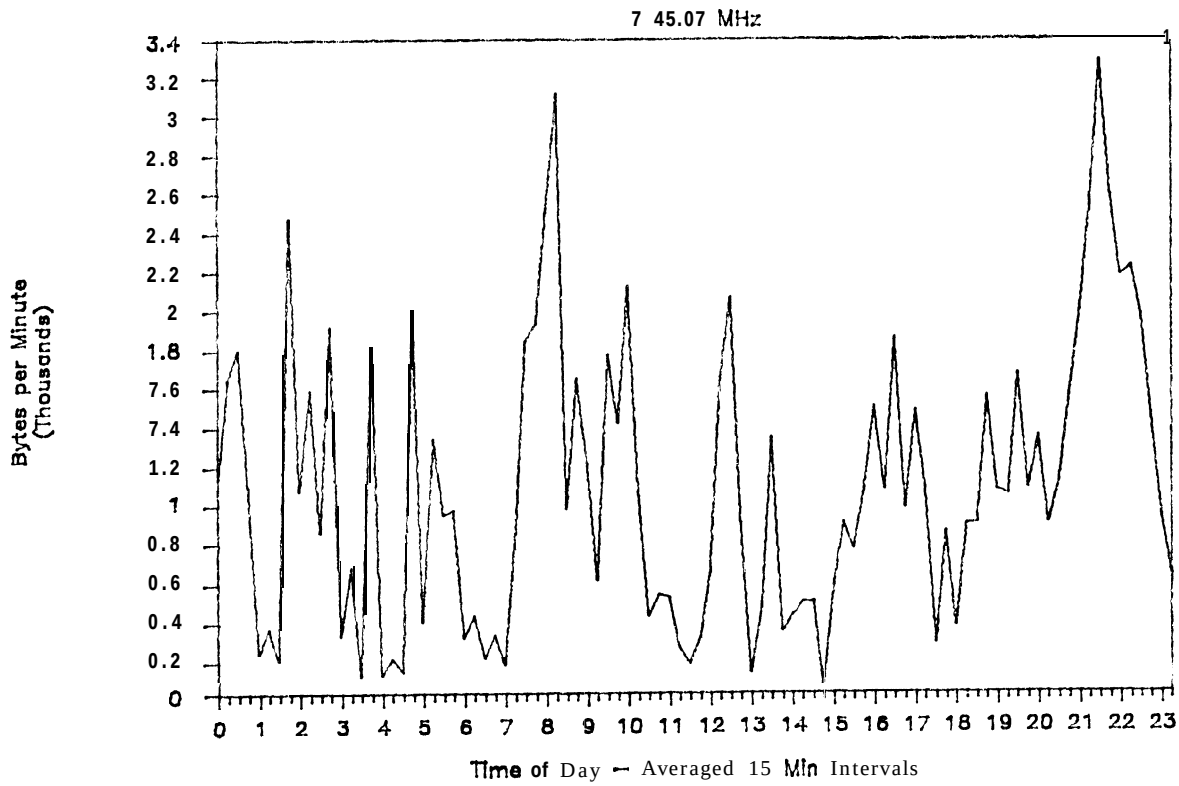

Fig 3. Efficiency

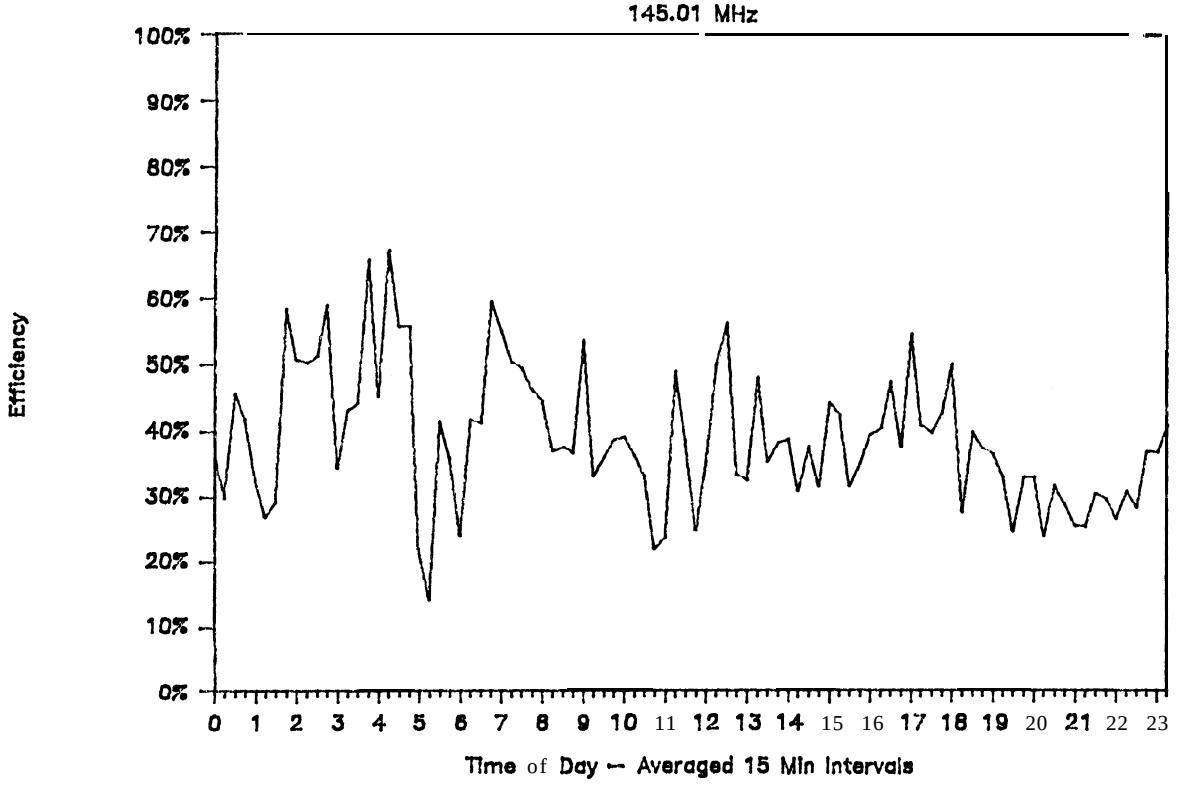

Fig 4. Efficiency Vs. Users

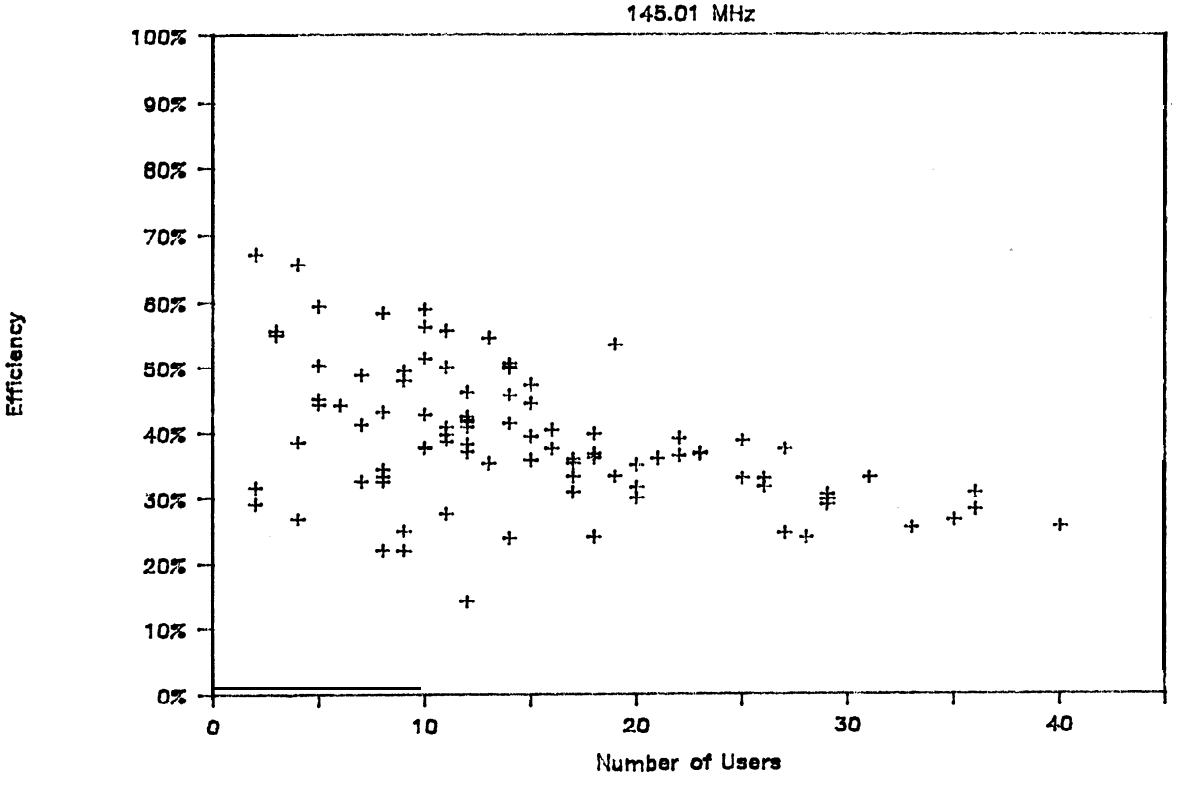

Fig 5. Total Bytes per Minute