Implementing MACA and Other Useful Improvements to Amateur Packet Radio for Throughput and Capacity

Steven Gunderson, CMoLR Project Manager
John Bonnett, KK6JRA / NCS 820

ABSTRACT

Amateur Packet Radio suffers from low throughput and "hidden terminal problem." A protocol applique within AX.25 addresses these issues. It uses unnumbered frames with sequence number extension and employs KA9Q's Multiple Access with Collision Avoidance (MACA) media access control. Throughput improved few to several-fold and network capacity increased.

1. Introduction

California coastal counties share several common characteristics: a) coastlines; b) major earthquake faults, e.g. San Andreas; c) mountains and hills; d) frequent natural disasters – e.g. wildfires, earthquakes, floods, slides and tsunamis; and e) California missions. Because of their common California mission heritage, this paper refers to them proverbially as "Mission" County.

Mission County has extensive Amateur Radio Packet Networks (packet) employed for auxiliary emergency and disaster communications with stations located in Emergency Operation Centers (EOCs), fire and police stations, and hospitals. Legacy 1200 baud packet networks are common, often with 9600 baud or faster repeater crosslinks. Outpost and Winlink 2000 (W2K) messaging software are used to communicate messages and forms.

Mission County Community Emergency Response Teams (CERT) hold annual OK Drills[1]. OK signs are distributed throughout communities; placed in windows, on doors, fences, and mailboxes visible from streets. CERT teams survey neighborhoods and report OK sign counts to CERT neighborhood Incident Command Posts (ICP), which are summarized and communicated to EOCs by voice. Some communities place additional CERT triage signs throughout neighborhoods which are discovered and reported to neighborhood ICPs[2]. CERT Form #1 damage assessment information can be included to initiate and simulate emergency communication during periods of communication outages.

Mission County sought to receive OK Drill triage information as data rather than voice, to promote speed and accuracy for Situational Awareness (SA) during major emergencies and disasters when communications may be interrupted. A Mission County with 1 million people and 300,000+ residences, organized into 150 neighborhoods of approximately 2,000 addresses each, can produce a flood of information during the "Golden Hour" when CERT neighborhood teams perform Initial Rapid Assessments (IRA).

The Communication Methodology of Last Resort (CMoLR) project was established to facilitate emergency data communications from CERT to Public Safety. Objectives were:

- Independent data communication system
- Interoperable between Amateur Radio and Land Mobile Radio (LMR)
- "Make it work with what we have"

Amateur packet radio physical (2FSK) and data link standards (AX.25) over legacy analog FM radios were used to promote interoperability between Amateur and Public Safety communities. FCC narrow-band guidelines were followed.

Preliminary 1200 baud packet radio demonstration sent Depiction mapping elements with locations and emergency status properties in one long APRS® packet within a few seconds, several times faster than previous tests and demonstrations employing messaging software through packet networks. Subsequent speed and throughput tests compared Amateur Radio messaging software (e.g. Outpost and Winlink 2000) operating through packet networks with extended APRS messaging optimized for CERT communications.

2. **CONNECT and UNPROTO Speed and Throughput**

Two packet modes were used for speed and throughput testing: CONNECT and UNPROTO. Connect is reliable and UNPROTO is unreliable.

CONNECT mode was used for messaging tests. Winlink 2000 tests most likely used PACLEN 128 and MAXFRAME 4 defaults. Outpost tests used MAXFRAME 6 to reduce TX/RX turnarounds for higher throughput. Winlink 2000 used LZH compressed binary files and Outpost used ASCII text files. Binary file transfers invoked High-level Data Link Control (HDLC) bit stuffing expansion that may have contributed to lower throughput than ASCII transfers.

Winlink 2000 reported[3,4] 1200 and 9600 baud packet normalized throughput 4-14x slower than ideal speed, accounting for header overhead (bps/10), with 4,000 byte compressed binary file, Table 1.

Table 1. Winlink 2000 reported 1200 and 9600 baud packet throughput

Winlink 2000 Binary – (4,000 bytes)	**Time** min	**Time** seconds	**Binary CPS** 4,000/seconds	**Ideal Speed** CPS	**Throughput** %
Packet (1200) direct	2	120	33	120	28%
Packet (1200) 1 node	2.5	150	27	120	22%
Packet (9600) direct	1	60	67	960	7%

Similar results were found with Outpost tests using 2,410 byte (page of text) ASCII file, Table 2. Kantronics KPC-9612 Terminal Node Controller (TNC) was used with professional Motorola LMR radios.

Table 2. Outpost measured 1200 and 9600 baud packet throughput

Outpost ASCII – (2,410 bytes)	**Time** sec	**ASCII CPS** 2,410/time	**Ideal Speed** CPS	**Throughput** %
Packet (1200) – KPC-9612 / Motorola	53	45	120	38%
Packet (9600) – KPC-9612 / Motorola	20	120	960	12%

CONNECTed packet network throughput was lower than ideal speed. 9600 baud did not provide expected several-fold speed increase and fell well short of ideal speed. Lower throughput can be attributed to frequent TX/RX turnarounds from short packet size and limited frame length. TX/RX turnaround time exceeded data transfer time.

UNPROTO throughput tests were run with ZIP compressed ASCII files, base64 encoded. Multipurpose Internet Mail Extensions (MIME)[5] uses base64[6] resulting in 4/3 expansion. Encoded files were evenly divided into segments ending with <CR> to optimally stuff packets and trigger packet transfer. Base64 encoding avoids AX.25 header flag field character hex 7E (ASCII ~) and uses text characters. UNPROTO does not support binary and some TNCs do not support binary in CONNECT mode, therefore, ZIP compress and encode is the safest approach.

UNPROTO tests used PACLEN 256. Additional sliding window sequence control bytes were inserted within data payload, reducing maximum data payload to 250 bytes. Larger frames (188 and 35343) using 1 or 2 bytes support longer sliding windows to minimize TX/RX turnarounds during long transfers and/or over fast links. Preliminary 1200 baud UNPROTO tests sent ACKs every 6 packets, same as Outpost tests, with similar file size to Winlink 2000 tests. Longer windows (16-19) were tested and a default of 16 was settled upon as a compromise to improve throughput, but not to exceed transmitter duty cycle.

Compressed/encoded and uncompressed file sizes are reported, Table 3. Larger files compress more, evident by 2,410 byte ASCII file used for Outpost tests, it compressed 2X, then, expanded 4/3 when encoded. ZIP typically compresses larger ASCII files up to 4X. ZIP also compresses UNICODE 16bit/character files, 7X typical. Encoded ZIP is 3X overall.

Table 3. 1200 baud CONNECT and UNPROTO throughput

1200 Baud Mode	**PACLEN / Frame**	**File Size** cmpr (uncmp)	**Time** sec	**CPS**	**Ideal Speed** CPS	**Throughput** %
CONNECT – W2K	128 / 4?	4,000 (binary)	120	33	120	28%
CONNECT – Outpost	128 / 6	2,410 (ASCII)	53	45	120	38%
UNPROTO	256 / 6	1,588 (2,410)	24	66	120	55%
UNPROTO	256 / 6	4,520 (7,959)	62	72	120	60%
UNPROTO	256 / 19	4,520 (7,959)	57	82	120	68%
UNPROTO – Simplex	256 / 16	8,285 (22,495)	78	101	120	84%
UNPROTO – Analog Rptr	256 / 16	8,285 (22,495)	84	94	120	78%

Larger UNPROTO packets and longer windows improved throughput compared with CONNECT, effectively doubling to tripling throughput. Larger files increased overall throughput as session initiation and transfer close contributions were minimized. Private Line (PL) tone time delays were added for analog repeater tests, decreasing throughput in comparison with simplex tests.

Tests were rerun at 9600 baud comparing CONNECT and UNPROTO, Table 4. Other parameters stayed the same. Larger files and longer windows to increase on-air time and to minimize session initiation and transfer close contributions also were tested.

Table 4. 9600 baud CONNECT and UNPROTO throughput

9600 Baud Mode	**PACLEN / Frame**	**File Size** bytes	**Time** sec	**CPS**	**Ideal Speed** CPS	**Throughput** %
CONNECT – W2K	128 / 4?	4,000 (binary)	60	67	960	7%
CONNECT – Outpost	128 / 6	2,410 (ASCII)	20	120	960	12%
UNPROTO	256 / 7	8,285 (22,495)	20	412	960	43%
UNPROTO	256 / 17	8,285 (22,495)	16	514	960	53%
UNPROTO – KPC-9612	256 / 96	21,621 (77,745)	26	830	960	87%
UNPROTO – Internal Kenwood TM-D710	256 / 96	21,621 (77,745)	36	600	960	62%
UNPROTO – Internal Kenwood TM-D710G	256 / 16	21,621 (77,745)	43	487	960	51%
UNPROTO – Internal Kenwood TM-D710G	256 / 128	21,621 (77,745)	37	566	960	59%

UNPROTO achieved few to several-fold higher throughput than CONNECT at 9600 baud. Longer windows and larger files were needed to achieve expected several-fold throughput increase at 9600 baud. Normalized UNPROTO throughputs (%) were similar at 1200 and 9600 baud.

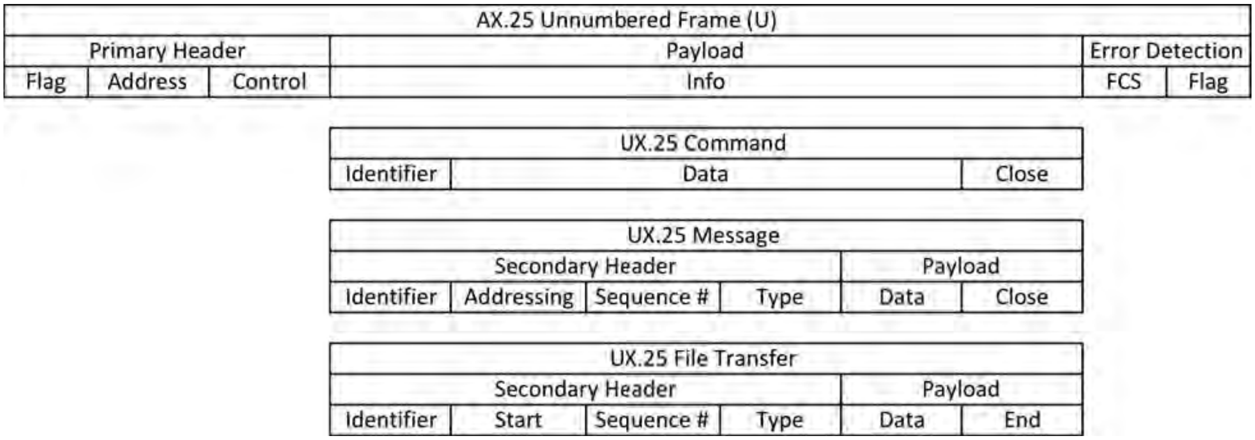
The last two tests, comparing 16 and 128 windows, added 2 second transmit time delays, decreasing throughput. They improve heterogeneous network interoperability with mixed equipment and TNC's. These delays also allow other stations to "break the net" with higher priority traffic, before channels are tied up for longer periods.

Kantronics KPC-9612 TNC with professional Motorola radios achieved high throughput. In one test, 920 cps was observed during with long windows with no transmission errors. Kenwood internal Tasco TNCs were slower than Kantronics TNCs at 9600 baud. Kenwood TNCs pause every few seconds for a fraction of a second, reducing throughput. Kenwoods, however, have good 9600 baud link reliability, approaching that of 1200 baud[7].

AX.25 throughput was improved by using UNPROTO with extensions. The resulting protocol was code named UX.25 for UNPROTO AX.25.

3. UNPROTO AX.25 – UX.25

UNPROTO AX.25 (UX.25) follows APRS conventions[8] and is formatted as experimental APRS packets. AX.25 unnumbered frames (U) serve as a wrapper around a secondary UX.25 packet riding inside AX.25 data payload, Figure 1. AX.25 headers provide frame synchronization, network addressing, control, and error detection. These functions are not duplicated by UX.25.



AX.25 Unnumbered Frame (U)					
Primary Header			Payload	Error Detection	
Flag	Address	Control	Info	FCS	Flag

UX.25 Command		
Identifier	Data	Close

UX.25 Message					
Secondary Header				Payload	
Identifier	Addressing	Sequence #	Type	Data	Close

UX.25 File Transfer					
Secondary Header				Payload	
Identifier	Start	Sequence #	Type	Data	End

Figure 1. AX.25 and UX.25 formats

UX.25 provides secondary header with identifier character, addressing, sequence number, packet type, data, and close. UX.25's secondary headers are small to minimize impact on data payload, using no more than 6 bytes including identifier character and closing carriage return <CR> character leaving 250 byte data payload.

UX.25 packets come in three types with multiple options:

Table 5. UX.25 packet types

	Broadcast		**Directed**	
Type	Addressed	Unsequenced	Addressed	Sequenced & ACK'd
Command	Optional	X	X	X
Message	Optional	X	X	X
File Transfer				X

UX.25 packets can be broadcast (unsequenced) or directed (sequenced and ACK'd). File transfers are always sequenced. Broadcast packets are identified by identical destination and source call signs in AX.25 address fields and directed packets are identified by different source and destination call signs in AX.25 primary header. Message and file transfer payloads start with APRS Data Type Identifier (DTI) unused character. Secondary source and destination addresses are included in data payload for messages. Secondary addresses are optional for broadcast packets.

AX.25 CONNECT's sequence numbers and error control are not used by UX.25, it provides its own sequence numbers and error control to support larger windows with advanced error control strategies. UX.25 uses single and double byte sequence numbers for message packets and file transfers respectively. Single byte provides 188 sequence numbers suitable for short data (47 KB), and double byte provides 188 X 188= 35344 sequence numbers, able to support a 8.8MB file with 250 byte packets.

As an aside, AX.25 specification[9] includes modulo 128 integers with up to 127 sequence numbers, but most commercially available TNCs do not support this capability, they are limited to modulo 8 frame size of 7. Increased modulo 128 sequence numbers (127) may have addressed most observed AX.25 throughput limitations. Larger sequence numbers minimize RX/TX turn arounds, but unavailability of modulo 128 sequence numbers in most commercial TNCs, "make it work with what we have" and interoperability requirements led to an interior appliqué with AX.25 unnumbered packets.

UX.25's error control strategy employs a combination of ACK and selective NAK (SNAK). ACK packets list highest received sequence number (+seq), and are used for command, login, message, and file transfer packets. NAK transmits highest received packet sequence number (+seq) and missing packets (e.g. +seq-seq-seq), minimizing retransmission of received packets. This approach places burden on receiving stations to disregard duplicate packets and to reassemble out of order packets. It also supports up to 75% packet loss without stalling.

Broadcast command packets are similar to APRS packets with simple commands plus optional data. Table 6 lists broadcast command packet types.

Table 6. Broadcast command packet types

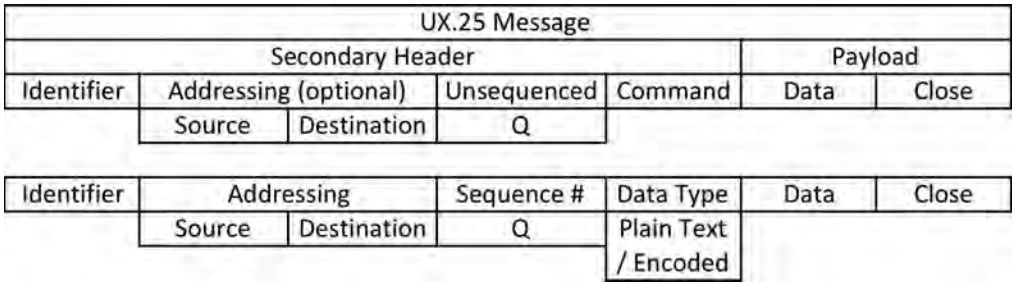
Command	Description
CQ?	Who's my repeater?
SH	I'm your smart host, e.g. repeater
CALLSIGN	My call sign
GPS	Coordinates
TIME	GMT, etc.

Directed command packets follow message format with secondary header and data.

Table 7. Directed command packet types

Command	Description
BROADCAST	Broadcast request for repeater
CALLSIGNS	My call sign + call signs heard
CERT	Address + damage assessment
MAP	KML, KMZ, SHP

Message packets include a secondary header with addressing; Q field (unsequenced / sequence #); command / data type; and data (Figure 3).



<table>
<tr><td colspan="7">UX.25 Message</td></tr>
<tr><td colspan="5">Secondary Header</td><td colspan="2">Payload</td></tr>
<tr><td>Identifier</td><td colspan="2">Addressing (optional)</td><td>Unsequenced</td><td>Command</td><td>Data</td><td>Close</td></tr>
<tr><td></td><td>Source</td><td>Destination</td><td>Q</td><td></td><td></td><td></td></tr>
</table>

<table>
<tr><td>Identifier</td><td colspan="2">Addressing</td><td>Sequence #</td><td>Data Type</td><td>Data</td><td>Close</td></tr>
<tr><td></td><td>Source</td><td>Destination</td><td>Q</td><td>Plain Text / Encoded</td><td></td><td></td></tr>
</table>

Figure 3. UX.25 message formats

Secondary source and destination addresses are two byte index numbers for directory service. Q field is one byte and includes unsequenced flag (-) or encoded sequence number (1-188). This supports messages extending over multiple packets.

Table 8 lists unsequenced commands. They are represented by a single character.

Table 8. Unsequenced commands

Command	Description
SYN	Sync, i.e. Login
ACK	Login: +seq or OK, data: +seq
NAK	Login, unknown user, bad passwd, file too large, data: +seq-seq-seq
SY / SN	Send Yes / No
DAT	Data
EOF	End-of-file
CLO	Close

Sequenced message data types include:

#	Plain text
*	Encoded compressed

SYN (sync, i.e. login) packet is core to file transfer and combines multiple functions into one packet, Figure 4. This approach avoids lengthy session protocol exchanges with TX/RX turnarounds and follows Unix-to-Unix CoPy (UUCP) conventions.

```
     Login      Passwrd   SN      Job Name    Org File Name  Cmd  Pkts Zip Org Jobs Notify
user@domain.net|LetMeIn|sernum|1309D100502000|TestData2.txt|uucp|  4  |706|956|  0  |notify|
       1           2       3          4              5         6    7    8   9   10    11
```

Figure 4. SYN (Sync) packet format

Table 9 lists SYN (Sync) packet fields and types. First three fields are for authentication, and password is encrypted. User account and domain names can be use in lieu of station call signs. AX.25 header includes call signs which function similar to UUCP host names.

UUCP job control fields follow UUCP X.file conventions. The remaining fields were added for packet. Expected packets and file size are useful for controlling radio links and notifying stations how long file transfer transmissions will last. Both compressed and original file sizes are needed for Zip decompression.

Table 9. SYN (Sync) packet field descriptions and types

Field #	Description	Authentication	UUCP	Packet
1	Remote Account Login	X		
2	Remote Password	X		
3	Remote Serial Number	X		
4	Job Name		X	
5	Original File Name		X	
6	Job Command		X	
7	Expected Packets			X
8	Compressed File Size (bytes)			X
9	Original File Size (bytes)			X
10	Expected Jobs		X	
11	Notify		X	

Directed commands and messages are used to set up and control file transfers. Once remote sites and repeaters agree to requested file transfers (SYN login message) with Send Yes (SY), file transfers themselves require little additional information. Figure 5 lists UX.25 file transfer packet format.

```
    1        2        3     250            ' ChrB(2),  STX   ctrl-B  Start of a Text
[-------][-------][-------](data)[-------]  ' ChrB(3),  ETX   ctrl-C  Returns to command mode
  start    seq 1    type            end     ' ChrB(4),  EOT   ctrl-D  End of a text/packet
 ChrB(7)                          ChrB(4)   ' ChrB(7),  BEL   ctrl-G  Start of a packet
 <PKT>       0        0     data   <EOT>    ' ChrB(13), CR    ctrl-M  Carriage return

ChrB(7) + IntToChrId(seq) + IntToAxSeq(type) + data + ChrB(4) + ChrB(13)

Seq:  0 - 35344 (188 * 188)

Type: 0      SYN Login/sync Packet, login password SN job file expctpkts cmprlen origlen
      10-49  DAT Packet, 10 sender pausing for ACKs/NAKs, 24-11 expect more data pkts
      150    DAT EOF packet
      170    NAK Login, unknown user, bad password, file too large
      171    NAK packet Data: +seq-seq-seq  where + is an ack, - is a nak
      172    NAK File, corrupt file data
      180    ACK Login +seq or OK
      181    ACK packet, Data: +seq
      182    ACK File
      188    CLO packet

Data: Can be up to 250 characters, at 251 characters the TNC rolls another packet

EOT:  Char(4) and Char(13) if packet less than 255.
```

Figure 5. UX.25 file transfer packet format

Stations request permission to send long multi-packet files using SYN (Sync, i.e. login) packets and are granted permission using SY (Send Yes) packets. By using UNPROTO instead of CONNECT, other stations can listen to the file transfer setup conversation and keep silent until channel is clear. These factors, provide the foundation for a new packet radio Media Access Control (MAC), Multiple Access with Collision Avoidance (MACA).

4. Multiple Access with Collision Avoidance (MACA)

Phil Karn, KA9Q, proposed "MACA – A New Channel Access Method for Packet Radio"[10] in 1990 to address hidden and exposed terminal problems. Packet radio's MAC is based on ALOHAnet[11] and Carrier Sense Multiple Access (CSMA)[12] whereas stations listen for transmissions using carrier sense (CS), and wait for a pre-determined and/or random period following transmission by other stations. Stations try not to interfere with other stations but may inadvertently transmit while others are transmitting, causing network contention.

To further compound the problem, stations A & B located on opposite sides of a hill may not be able to hear each other thus may try to transmit to a shared repeater R at the same time, thinking channel is clear. This is called the "hidden terminal" problem, Figure 6.

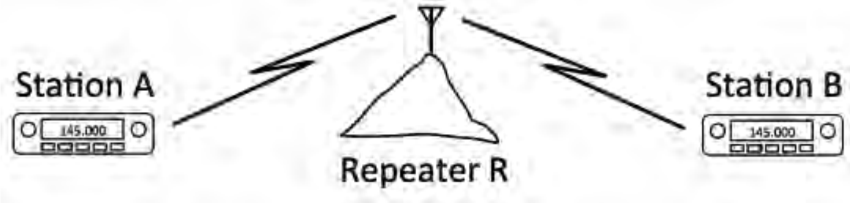


Figure 6. Hidden terminals

A similar and related problem is where a repeater R may be transmitting to one of the stations A and the other station B wants to transmit to another station C outside station A's radio range. Station B thinks the channel is in use, thus does not transmit to station C when it would otherwise be OK to do so. This is called the "exposed terminal" problem, Figure 7. (Note: Solving this requires disabling radio's carrier sense circuitry.)

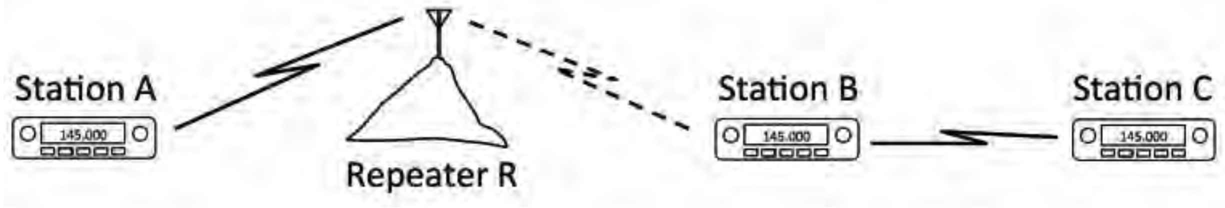


Figure 7. Exposed terminal

Phil's proposed solution was simple and elegant, employ Request to Send (RTS) and Clear to Send (CTS) as channel pilots to receive permission from remote repeaters and stations before transmissions, Figure 8. If a repeater R or station A were busy, repeater R would not respond to a request from another station B.

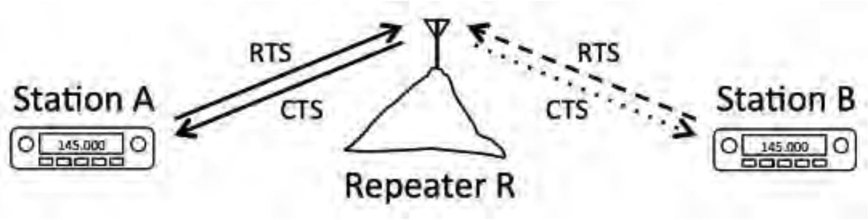


Figure 8. RTS/CTS exchange

Other stations overhearing RTS/CTS exchanges likewise know to keep silent, Figure 9. Station D overhears both sides of RTS/CTS exchange between station B and repeater R. Stations A & C each hear one side of RTS/CTS exchanges and know to keep silent. Station C hears initial RTS from station B, but does not hear CTS from repeater R. Station A does not hear initial RTS from station B (it is a hidden terminal) but hears CTS from repeater R. Stations A, C and D do not send traffic until RTS/CTS negotiated transfers are complete.

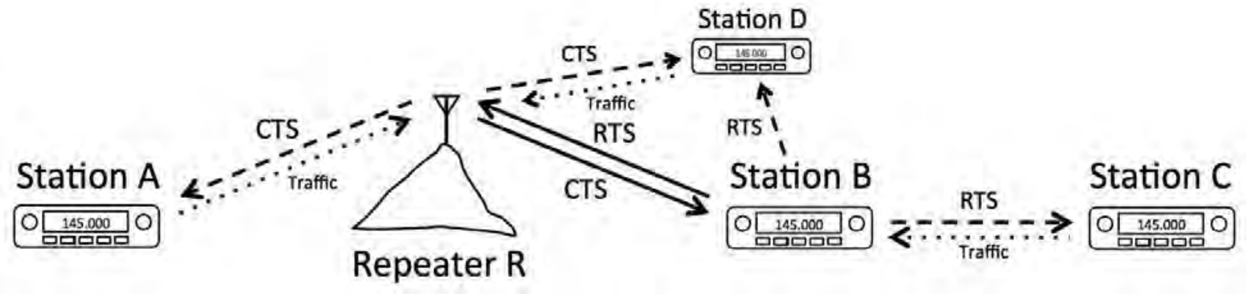


Figure 9. Overheard RTS/CTS exchange

RTS and CTS contain file sizes so that other stations can estimate wait time before transmitting. UX.25's Sync (SYN) and Send Yes (SY) packets additionally contain expected number of packets plus file sizes. Send Yes (SY) repeats Sync (SYN) expected number of packets and file sizes for stations that did not hear initial RTS packets.

RTS/CTS are not used for single command and message packets, or small groups of packets (e.g. 4) for longer messages that require less than a couple seconds to transmit. Overhead required for RTS/CTS negotiation would exceed transmission time. RTS/CTS are used for file transfers that will occupy channels for lengthy time periods.

Table 10 lists MACA and UX.25 file transfer negotiation packet equivalents.

Table 10. MACA and UX.25 file transfer negotiation packet types

Protocol	Request	Proceed	Don't Proceed	File Size	Estimate Packets
MACA	Request to Send (RTS)	Clear to Send (CTS)	—	X	—
UX.25	Sync / Login (SYN)	Send Yes (SY)	Send No (SN)	X	X

5. Directed Packet Networks

One of packet's fundamental problems is stations determine when to transmit. This works well for lightly loaded open packet networks with short transmissions, e.g. APRS. It does not work well for heavily loaded networks with mixed traffic, both short and long. Stations do not know the nature of transmissions, or how long they will last, because of carrier sense. All traffic appears to be the same, until too late. Stations sending long file transfers (e.g. e-mail) can hog the network, preventing short urgent and priority messages from network access.

Radio Amateurs have long solved this problem for directed voice networks. Their scripts are sophisticated Media Access Controls (MAC). Stations check in (CQ), identify traffic (urgent, priority, routine), and alternately receive or don't receive permission from Network Control. Short breaks allow stations with urgent and priority traffic to "break [onto] the net" ahead of routine traffic.

MACA lays the foundation to incorporate directed network control principles into packet networks. MACA was originally proposed for single-frequency amateur packet radio networks. It was hoped "it may *finally* make single frequency amateur packet radio networks practical. …The ability to create usable, ad-hoc, single frequency networks could be very useful in certain situations… This would be especially useful for emergency situations in remote areas without dedicated packet facilities."[10]

UX.25 extends MACA by incorporating additional information (e.g. expected packets) before sending large files (e.g. e-mail). UX.25 also incorporates UUCP's Send Yes (SY) and Send No (SN) commands. Send Yes (SY) and Send No (SN) are essential for directed packet networks. Stations no longer determine when they can transmit files, they can be told "no." Digital repeaters (digipeaters) can have authority to determine who and when they allow access, and for how long.

Station to station simplex comms, without relaying through digipeaters, is supported. MACA RTS/CTS conversations clue station's "when the coast is clear" for simplex comms. UX.25 does not limit short single packet commands and messages. At 9600 baud, they last less than half a second. Mid-length messages (e.g. 1KB) last less than 2 seconds.

6. Brevity

The simplest way to increase throughput is brevity. Urgent and priority messages should fit into one packet (250 bytes), at most 4 packets (1 KB), similar to text messages. Messages shorter than 1KB do not benefit from compression, they are best sent as text.

Electronic mail provides two-part addressing (e.g. user@host), but is inefficient for short messages, e-mail headers can add hundreds of bytes of overhead. UX.25 expands short packet message usefulness for emergency messaging, where a lost packet may result in serious loss. Reliably sending useful information in half of a sec is advantageous.

Short messages can safely be assumed more urgent than e-mail. This is borne out by text messages serving different purpose than e-mail. Text messages are more immediate and perishable. E-mail is more deliberate and archival. Messages can be considered priority and urgent, and e-mail can be considered routine.

Message length provides a natural way to prioritize traffic and encourages network etiquette and usage. Allowing messages to extend a few more packets bridges the gap with e-mail so that users are less tempted to use e-mail for mid-length messages (e.g. 1KB) thus incur additional overhead and throughput reduction by using e-mail.

Directory services provides a way to address messages without high e-mail overhead. UX.25 supports message secondary source and destination addresses using two bytes each. Directory services is hosted by repeater node controllers, and in case of multi-digipeater networks, centralized in a super-node controller / server. Stations register with and join networks to participate in directory services. Directory copies are sent to participating stations once they join networks, and are periodically updated with changes as needed.

Directory services include: call signs, domain names, individual accounts and groups. Stations send their domain and local account information after checking in. Repeater nodes and servers maintain a common directory and distribute it to registered stations in the network. Secondary addresses allow messages to be addressed to individuals and groups with familiar two-part addressing without resorting to e-mail. Directory services also makes packet networks easier to use.

7. Trunked Packet

Packet networks with multiple digipeaters can support mobile terminals moving throughout coverage areas. Fast trunks (e.g. mesh) between digipeaters support inter-digipeater communication and coordination, and packet supports "last 10 mile" links to fixed and mobile terminals, Figure 10.

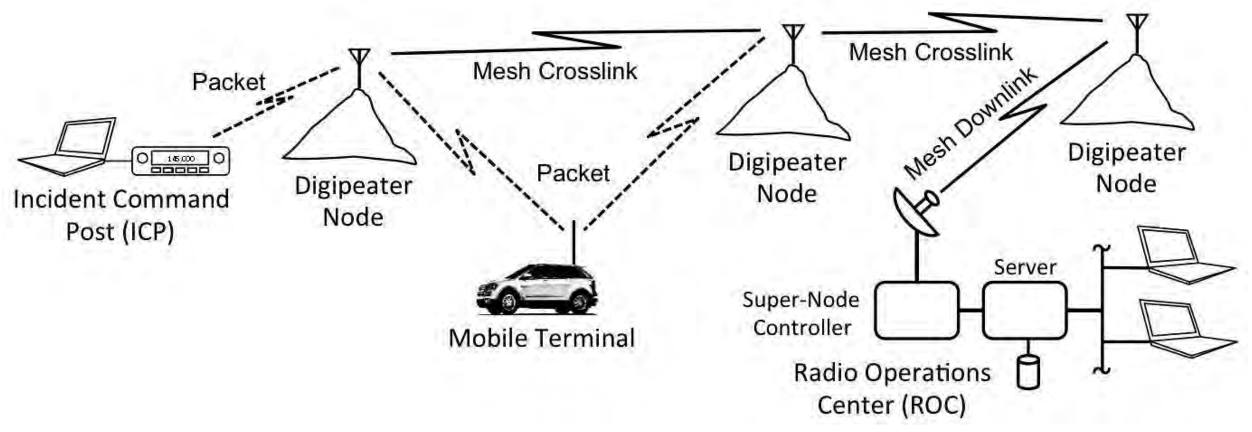


Figure 10. Trunked packet network

Stations check in with local repeater(s) and super-node servers coordinate digipeater nodes to send “I’m your smart host (i.e. digipeater)” (SH) messages. Stations then direct traffic to that digipeater. Direction can be based on proximity (e.g. relative signal strength indicator, RSSI) or network loading whereas multiple digipeaters may hear a terminal and lightly loaded digipeaters may be better able to support the load.

Super-node(s) keep track of stations and forwards messages to closest digipeater node. Digipeater nodes coordinate transmissions with individual stations using media access control (e.g. MACA and directed packet). When mobile stations move between digipeaters, messages are forwarded from previous digipeater to next digipeater under super-node control. This provides network handoff functionality, similar to trunked voice networks. Messages can be replicated between super-nodes and digipeater nodes to speed handoff, and to provide network resilience.

Trunked packet operates at message level, rather than packet level. This supports message batching and compression to increase effective throughput. Batches may include messages addressed for multiple destinations inside and outside network. Short messages can also be batched (and compressed) to improve throughput, not just email. Nodes and super nodes support message addressing and routing inside and outside packet networks, without dependence upon external Internet message servers. Super-nodes are fully capable e-mail hosts and can be directly connected to the Internet, although it would be wise to use upstream smart host. This forms a “store and forward” message network.

8. Conclusion

Amateur packet radio is well suited for emergency data communications between communities and Public Safety. Extending existing packet networks provides a cost effective solution to extend emergency data communication into communities.

Packet network capacity and throughput can be improved by link protocol and media access control changes, without changing hardware. Faster packet radios (e.g. 9600 baud) provide expected several-fold throughput increase using UX.25’s improved link protocol support for longer packet sizes and windows. 20-fold throughput increase is possible.

Phil Karn’s Multiple Access with Collision Avoidance (MACA) solves the hidden terminal problem. Directed packet network solves the problem of stations “hogging the network” with long transmissions and increases packet network capacity. Directory services enable short messages to be addressed with familiar two-part addressing, encouraging using text messages for urgent and priority emergency messages, and makes packet easier to use.

Trunked packet architecture supports mobile terminals with automatic handoff between digipeaters, similar to land-mobile radio (LMR) and cellular networks. High speed mesh cross-links support trunked packet and complement slower packet radios.

These improvements support CERT data communications for Initial Rapid Assessment.

9. References

[1] "Communitywide Disaster Drill," OK Sign Program, Culver City Fire Department, https://www.culvercity.org/live/public-safety/communitywide-disaster-drill.

[2] "Op-Ed –Why Last Friday Night's Disaster Drill Was a Major Success," The Front Page online, December 11, 2007, http://www.thefrontpageonline.com/op-ed/why-last-friday-nights-disaster-drill-was-a-major-success.

[3] "Frequently Asked Questions (FAQ) about Winlink 2000 – Q&A 170," https://www.winlink.org/sites/default/files/wl2k_faq_20150314.pdf, March 14, 2015.

[4] "Winlink FAQ (Frequently Asked Questions - with Answers!)," February 28, 2018, https://www.winlink.org/content/winlink_faq_frequently_asked_questions_answers_2282018, and ftp://autoupdate.winlink.org/User%20Programs/wl2k_faq.pdf.

[5] "MIME," https://en.wikipedia.org/wiki/MIME

[6] " Binary-to-text encoding," https://en.wikipedia.org/wiki/Binary-to-text_encoding.

[7] "Comparing 1200 and 9600 in the TM-D710 during ATGP," KX4O, July 19, 2016, https://vapn.org/gear/comparing-1200-9600-kenwood-tm-d710-during-atgp.html,

[8] "Automatic Position Reporting System – APRS Protocol Reference – Protocol Version 1.0," The APRS Working Group, August 29, 2000, Filename: aprs101.pdf, http://www.aprs.org/doc/APRS101.PDF.

[9] "AX.25 Link Access Protocol for Amateur Packet Radio – Version 2.2," July 1998, https://www.tapr.org/pdf/AX25.2.2.pdf.

[10] Phil Karn, KA9Q, "MACA - A New Channel Access Method for Packet Radio," Proceedings of the 9th ARRL Computer Networking Conference, London, Ontario, Canada, 1990, http://www.ka9q.net/papers/.

[11] "ALOHAnet," https://en.wikipedia.org/wiki/ALOHAnet#The_ALOHA_protocol.

[12] "Carrier-sense multiple access," https://en.wikipedia.org/wiki/Carrier-sense_multiple_access.

10. Key words

AX.25, UX.25, KA9Q, MACA, throughput