Remote Echo Control (REC)
Version 2.00

Beta-J Release Notes

Well, there was only one buglette in Beta-I, although it was a nasty little one. I have gotten it squashed and have added the last of the new features that I expect to be in version 2.0.

Contents

This archive (REC_199J.ZIP) contains only the files that have been updated. The magic name of RECBETA will get you this archive, and it has also been hatched in to the SDS SOFTDIST file area. You need to apply the files in this archive on top of either version 1.99G or 1.99H. If do not have either of these versions, you will need to file request REC_199G.ZIP from either my own system or through the nearest SDS file distribution node carrying the SOFTDIST file echo. Please note that I have NOT included the doc's in Beta-J like I did in Beta-H. There weren't enough changes to warrant doing so, and I probably shouldn't anyways.

The updated files are listed below:

REC.EXE	Executable program
REC.OVR	Executable overlay
REGISTER.EXE	The new version of the Registration program
SAMPLE.CFG	sample configuration
BETA-H.PRN	changes for beta-H
BETA-I.PRN	changes for beta-I
BETA-J.PRN	changes for beta-J (this document)
RECREG.PRN	list of all currently registered users

Bug's fixed

The was only one real bug in Beta-I, although it was a mean little one. If an echo had any special characters in the first 8 characters of the tag name, REC would die on a Runtime Error 002 when it tried to create the file or directory for that tag. This beta filters out those special characters so they are not included in determining the DOS file/directory name.

Registration

The REGISTER program included in this archive now gives you the option of having the return message placed on hold at my system for

you to call back in and pick up. Otherwise my system will return the message to you via normal net-mail routing facilities.

The option appears after you select the type of message to send to my system (Add new, change existing, or delete). It is a simple Yes or No type question, and you have to press either "Y" or "N" before the program will continue.

Many of the registrations in the first month or two of the last beta release were lost due to database errors. I have fixed the problem temporarily and will be moving the data to more secure database in the near future. Please check the enclosed list of registered users.

If you are on the list and have not received a registration key, feel free to net-mail me and I will send you the key. Please let me know if you want it sent via net-mail routing or placed on hold. DO NOT SEND IN ANOTHER REGISTRATION MESSAGE if you already have been assigned the key. Part of the reason I need to move the data is that the current form doesn't allow for very good duplicate registration message detection. I will be working on the database conversion while this beta is being field tested.

If you are NOT on the registration list, you may either send in another registration message, or wait until 2.0 comes out.

In either case, you should be aware that I will most likely be changing the registration key computation to a more compact scheme with version 2.0. The beta-test keys you have been assigned will not be valid when 2.0 comes out. As before, the only difference between registered and unregistered versions of REC is the tear line on messages that REC creates.

Features

There were several new features or minor changes added to this version of REC. I will cover those in detail here.

1) "Forwarding messages to hubs" log and screen message will only appear when there are actually messages to forward.

2) A dump report can be generated which will show you exactly how REC interpreted your echo control file when it loaded it. The option is activated via the "/V" parameter, and it is NOT designed for everyday use. If an echo node is getting error messages when attempting to connect to an echo that already exists on your system, this option may help. It will also help in the process of converting

your system to REC. The output filename is fixed at "REC.DMP"and will appear in REC's directory

3) REC reply messages will indicate differences between accessing an echo that REC can't find, or one that it found but the echo-node doesn't have access to for forwarding. The second message is now "Echo found, unauthorized source" while the first message remains "Echo not found".

```
If Work = 'P' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgPrivate
Else If Work = 'R' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgReceived
Else If Work = 'S' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgSent
Else If Work = 'IT' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgInTransit
Else If Work = 'O' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgOrphan
Else If Work = 'KS' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgKillSent
Else If Work = 'L' then
  GlobalMsgAttr:= GlobalMsgAttr + MsgLocal;
```

4) The Sysop now change control the global attributes for all messages that REC creates. The defaults are still Private, KillSent, and Local. However this new statement will override those defaults with the ones you set.

```
MSGATTR [P] [R] [S] [IT] [O] [KS] [L]
     P= Private
     R= Received
     S= Sent
     IT= In-Transit
     O= Orphan
     KS= Kill-Sent
     L=Local
```

The vast majority of setups will not need this parameter, but there are always those few exceptions. By specifying this line at all, the default settings are ingored. That means that if you chose to add the attribute of Orhan to the global settings, you will need to specify the global settings on the line as well as the Orphan attribute. Likewise, if you wish to remove the KillSent attribute from the global defaults, you will need to specify other attributes of Private and Local on this line. If you put the line in with no attributes, REC will not put any attributes on the messages.

WARNING: Always use at least the Private attribute. Otherwise others people on your echo-node's system will be able to read messages from REC. Now that the downlink's password can be displayed on the

notify messages (read on for this one), this can become a major security hole.

5) REC now supports two new commands called SUSPEND and RESUME. It allows your echo-nodes to switch all their echos off for a time, and turn them all back on with a single command. This is useful for when a system is moving or otherwise going to be down for few days and they don't want to build up a ton of mail on your system.

The commands are formatted very similar to the echo report commands, meaning the echo-node just types ":SUSPEND" or ":RESUME" in a message to REC. It must be noted that these commands are performed immediately, so if the echo-node puts a Suspend command in front of an start request for a new echo, the new echo will NOT be suspended. The Resume command will automatically detect any echos that have been manually restarted since the Suspend command was issued, and not reconnect the echo-node to the echo a second time. More than one echo-node can be suspended at one time.

If the suspend command causes any echos to become dead-ends, they will be subject to Clean-up processing if you have it so enabled. Should such echos be cancelled and dropped by Clean-up processing, the Resume command will issued forwarding requests to have the echos restarted. The command will essential do the same thing as if the echo node had manually listed everyone of the desired echos.

6) The Sysop can now configure the maximum size of the messages that REC creates before it continues the messages on to a second message file. The default is 10,000 bytes and the minimum is 1,000 bytes. It MUST be realized that these numbers are approximate instead of exact limits. Messages can be up to 200 bytes larger than the limit depending on several variables, so adjust your limit accordingly.

The syntax is "MAXMSGSIZE {number of bytes}".

7) A new statement called "NotifyOptions" has been added to REC. This will allow the Sysop to control 3 optional features of REC: Putting the password and/or authorized users on the Notify messages, and placing the echo description on the Forwardable echo report. These settings are global to all messages that REC creates, and not specific nodes. If an echo node has no authorized users explictly listed in REC's configuration, they notify message will indicate that "Anyone" can submit a change request.

This can be a major security hole if implemented improperly. If you use the MsgAttr statement to remove the Private attribute from the message, any user of the echo-nodes system may be able to read the notify message. If this message gives the password and says anyone can start or stop echos, then the user can be tempted to see what they can do. Always use the Private attribute on REC's messages. If more

than one person on an echo-node's system requires a notify message, use the ExtraNotify option to send them their own message regularly or batch mode Notify command to sent it as needed.

The syntax for this statement is shown below. The parameters can be used in any order, and you do not need to use all of them. The options are case insensitive, but the spelling must be exact.

NotifyOptions [USERS] [PASSWORD] [DESC]

8) It is now possible for you to setup an EchoHub as automatic and then excempt that hub from automatic clean up. This uses the new ExemptCleanup statement, which has the syntax of "EXEMPTCLEANUP {echo-hub addr}". The echo-hub address must be specified in an EchoHub statement before this statement is found, or it will be ignored.

9) The Forwardable echo report will now indicate which echos are active, available immediate, and whether the echo-node is the source for the echo if the echo is active to the echo-node. The symbols are "-" for available immediately, "+" for active, and "*" if the echo-node is the source for the active echo. Only one symbol is shown for each echo, or a space, as the "*" automatically includes the "+" symbol.

Operations

Over half of the above new and changed features were made possible by a major change to the internal storage of the valid echo lists and the echo control file (ECF). Prior to this version, the ECF was stored as a simple list in memory in no special order. To find a specific echo meant searching the list from beginning to end. The echo-lists where read directly from disk each time they were needed, and they had to be in asscending ASCII sequence by echo tag-name. None of this is true any more.

Both the ECF and the valid echo lists are now stored internally using Binary Search Trees. This accounts for most of the increase in size of this release, but it also allowed for the elimination of a major amount of memory storage, so there was very little real cost with this solution in terms of memory usage and program size. But it did result in some major changes to the way REC operates. The changes allowd the easy random searching and access of these files, but it also changed their loading process quite drastically.

Echo Control File (ECF)

The new process for loading of the ECF is 3 steps. First it attempts to find and load in a key listing file. Second it will load in the actual ECF data. Third it will check the efficiency or "Balance" of the tree, and if the tree is less than 50% efficient it will automatically optimize (or "re-balance") the tree. If the tree had to optimized, REC will re-create the key listing file to be used next time.

The first time this version of REC runs, it will not have this key file. As a result REC will end up creating a very poorly balanced tree with an efficiency most likely less than 10% (depending on the sequence of your ECF). REC will have to rebalance the tree on it's "madien" run, and after that the tree's efficiency will be maintained automatically.

The tree can become out of balance over time. Each time a new echo is added or an existing one dropped, if affects the balance of the tree. REC will automatically create a new key file everytime it adds or drops echo tags, but it will not re-balance the tree until it becomes inefficient. However, if you go in to your ECF and manually add or remove echos, you don't need to maintain the key file. REC will automatically detect this and adjust the tree accordingly. This step is called "Pruning" the tree of dead-keys. Depending on which echos you manually add or delete, it can be from a minor to a major change in the balance of the tree. Any time REC re-balances the tree, the Log file will show the old and new efficieny ratings.

Where is the key file stored? It is stored in the same directory as the ECF. In fact it has the same file name but uses the extension of ".KEY". This file is stored in standard ASCII format, BUT DO NOT ATTEMPT TO MANUALLY EDIT THIS FILE!!!. At first glance the list may not appear to be in any specific order, but it is actually in a VERY specific order. If you modify the file, REC will most likey end-up re-balancing the tree more often. This re-balancing process is time consuming and disk intensive, which is why the key file is used as a "short-cut" to speed up the process. If you loose or delete this key file, REC will simply rebuild it from scratch after re-balancing the tree.

Valid Echo-Lists

The valid echo-lists work very similar to the ECF but with a few noticeable changes. These lists consists of only 2 pieces of information, the echo's tag name and description. The majority of the time REC is only interest in knowing if the echo exists, and it

doesn't care about the actual descriptions. This explains the difference in how REC treats these lists.

When REC goes to load the valid echo lists, it will look for the same type of key file, but also for a random access file that holds the echo descriptions. It then compares the file date/time stamps between these two files against the file date/time stamp of the original echo-list. If all 3 files don't have the same date/time stamp, the echo-list is "re-compiled". This is how REC detects a new raw echo-list as well as any possible changes to the "compiled" listings.

To "compile" the echo-list, REC will simply read in the entire original or "raw" echo-list, building the tree, and building the data file that holds the descriptions. If an echo doesn't have a description then it won't have an entry in the data file, which saves on disk space since each description is a fixed length. After the tree is built, REC will re-balance the tree and write out a key. Lastly REC will set the date/time stamps on the key and data files to same date/time stamp as on the echo-list. At this point, both they echo tags and the descriptions are considered "loaded". Because of this type of storage and retreival, the raw lists no longer need to be sorted in to ASCII sequence prior to being used.

Assuming the file date/time stamps do match, then REC just loads in the key file in to memory. The description file is not even opened much less read. Should an echo-node request a forwardable echo-report, AND you have enabled the DESCription option of the NotifyOption statement, REC will load the descriptions. However, REC again takes a short cut. REC only loads a pointer to the actual description on disk instead of the entire description. This means that REC has to actually read the description from the disk every time it is needed, but each entry in the valid echo lists only takes about 15 bytes plus the length of the echo tag in memory, rather than including the description in that total as well. The extra disk access, rare as it will be if used as all, is well worth the enormous savings in memory usage on every run. And since the data file is in a random access format (RRDS for those programmer types!), disk access is extrememely fast. I will doubt that if you see a performance drop no matter what speed your disk is. In fact you will most likely see a speed increase!

The efficiency rating of tree is NOT checked, since the file is never changed once it is compiled. This is the major reason you NEVER EDIT THE KEY OR DATA FILES. It is possible for you to make changes to the files and then reset the date/time stamps. But if you want go out of your way to make your life miserable, I feel no reason to stop you.

The compiled files are actually stored in the same directory as the raw echo-list, but with the extensions of ".KEY" and ".RDM". I strongly recommend that you use DOS to rename all your echo-lists to

the extension of ".LST" but this is entirely up to you. If you delete the compiled echo-list files, REC will re-create them. If you delete the raw echo-list file, REC will give you configuration errors since it can't find the echo-list file as listed in the configuration file. If you delete the echo-list filename from the echo-hub's configuration statement, the compiled files will be ignored.

Performance Impacts

For those of who out there who are either mathematicians or programmers, this will interest you. The average number of searches of a straight list in not specific sequence is defined as "#entries/2", or if your list is 200 entries then each random search will take an average of 100 passes. However, in a binary tree the MAXIMUM number of searches is "SqrRoot(#entries)+1", or 15 searches for the list of 200. If you think about the FidoNet valid echo-list which has at least 600 entries, that's an average of 300 compared to a maximum of 25 searches.

What does all this mean? Well, before this release REC would process the user's message by going through the entire ECF one echo at time and seeing if any of the user's message lines had anything to do with the echo. Now, REC check the user's message lines, and then picks the specific echo (or echos) that need to be changed. The fact that REC ran so quickly in prior releases was a tribute to the tight loops I had designed. But it was still the "slow" way to do things.

REC's processing is now driven by the message or the batch commands entered. This has drastically reduced the actual number of searches required, as well the fact that each of the searches are significantly faster. The result is that even with the longer loading times for the ECF on each run, REC's overprocessing processing speed has dropped noticeably. If you would like to see a perfect example, time how long REC takes to load the ECF with and without a key file. For an even better example, time how long REC takes to "compile" the raw echo-lists as compared to how long it takes to load the keys and descriptions. Even if you don't count the balancing and prunng processes, you will notice the speed increase.

There is one drawback to this new process. The ECF file now has to be sorted each time it has to be written back to disk. The sorting method is the same as used in the previous versions, but the sort will happen more often. REC has been setup to maximize the amount of memory available for the sort and minimize the size of each record to be sorted. However a sort is still "long" process to be performed. Considering the overall benefits to each run of REC, and the fact that not ever run of REC will require a new ECF to be created, it's well worth the extra step.

Program Support

There is software support echo called REC_SUPPORT that is already on the MetroNet backbone. Any and all members of MetroNet are encouraged to pick up the echo from there regular MetroNet echo source. FidoNet nodes are encourage to pick up the echo from either a local source already receiving the echo or myself directly. If you wish to get connected to the echo, please send me a net-mail message. If there is already a node in your net receiving the echo, I'll let you know.

The only requirements for receiving the support echo is that you must be running either a live or beta version of REC. You can pass the echo off to anyone meeting this requirement without my prior permission. All I ask is that you inform me of such action either via the echo (preferred for now) or via net-mail.

To get the echo on to the FidoNet backbone, I need three things: 1) 25 nodes receiving the echo, 2) 25 messages per week, and 3) at least 2 REC's to request the echo within two weeks of when I asked for it to be placed backbone. As of this writing the traffic level on the support echo is about 45 messages per week. It's a bit on the high side due to some excellent discussions about security changes desired for REC 2.10, as well as reports on changes with the beta's. However I only have about 15 nodes currently receiving the echo.

I must limit my use of routed net-mail support for a program that can be "commercial" if I wished to charge for it. If I do not I could loose the benefit of outbound routed net-mail entirely and this would be very unfortunate for several reasons.

Conclusion

I strongly suspect that this will be last beta version of REC prior to 2.0. All other changes are being added to the plans for REC 2.10 which I hope to start close to middle of January (yes, I'm taking a break for the Holidays!). Some very significant changes are being suggested, so please join us in the REC_SUPPORT echo to help develop the program that you rely on.

Thanks again to every one using, and beta-testing, REC. Later!

Dan Fitch, Author of REC

1:104/435@FidoNet