Remote Echo Control (REC)
Version 1.20

Sysop's Documentation

**** NOTE ****
Beta test copies are only to be distributed to persons that I have authorized as REC Beta Tester and not made available for download or file request or any other means of file transfer.

Table of Contents

Introduction

Overview

Remote Echo Control (REC for short) is simply a zone-aware, point-aware program written for echo hubs that allows your echo nodes to remotely control which echos they receive from your system. This is done with no manual intervention on your part.   Installation is quick, configuration is simple, and execution is fast, and logging is complete.

AreaFix (by George Peace) is another program that does these same basic functions. It is a fine program if it fits your needs. However, REC provides all the functions of AreaFix, and also provides several other functions that I found missing in AreaFix. These are functions that I found essential as an echo hub in multiple zones and networks, and they are Full zone & point support, complete Zmail support, and Automatic Cleanup. Both of these will be explained later.

In fact, many of the features of REC can be very advantageous to an echo hub with only one network address. If you are and echo hub in multiple zones or a gateway system, then REC will be very valuable.

Lastly, if you are using the ZMail Mail Processor, then REC is necessary since AreaFix does not completely support all of Zmail's features. In fact, REC was written specifically to work with Zmail for this very reason, yet it is flexible enough to work with nearly any mail processor.

REC is fully compatible with all forms of point addressing used by most echo mail processors. You can use the private net, point reference, or a fully qualified address. If you are not involved with point systems, don't worry about this. If you are then please read the section on point systems.

Copyright and Disclaimer

Remote Echo Control (REC) and its accompanying utilities are copyrighted by Daniel S. Fitch, and all rights are reserved internationally. The programs and documentation are not to be decompiled, altered, or re-engineered in any way. This includes any patches to the load modules, and removing the copyright notice that is displayed when the programs are executed.

You use the software at your own risk. I will not be responsible for any damages resulting from the use of REC, including but not limited to software, hardware, outages, or monetary. I have taken all reasonable steps to properly test this software, and will take all reasonable steps to correcting any problems discovered.

License for Distribution

REC may be distributed by anyone in either FidoNet or MetroNet organizations. It must be distributed with all original files present. No files may be added. None of the original files may be altered in any means. Changing the archive format is permitted, as long as the contents of all files in the archive remain the same. This distribution is to be done FREE OF CHARGE by the distributing system, except for a nominal fee to cover any charges for postage or disk media. Charges incurred by users of a Pay BBS are not included in this restriction, unless the charge is tied explicitly to REC.

General Releases of REC are distributed via the SDS system. Please contact your nearest SDS system for the most recent copy of REC. If you are unable to find it from your nearest SDS system, feel free to file request it from myself or any of the Beta Test sites. It can be requested from any of these locations by using the magic name of "REC", or requesting "REC_???.*". However, the later form may give you some duplicates.

ALPHA test versions are to be only run on my own system, and not by any other system. They are not for distribution by any means.

BETA test versions are to be passed out only to the systems that I have authorized to receive these versions. Unless I arrange otherwise, I will personally deliver all BETA test packages to my testers. The documentation (sysop, user, history, etc.) of the beta test versions may be distributed to anyone interested, along as they are not altered in any means. For information on becoming a beta tester, please read the section on Beta Test Applications.

Registration for Use

You are expected and required to register the software. This way I know how many copies are in use, and whether I should continue to maintain it. Registered users will get first notice on updates, and ONLY registered users will be allowed to be beta testers.

There is no charge for registration. REC and its accompanying utilities are not crippleware. There is no change in operation of the software by registering the software.

Registration takes only a simple net-mail message to Dan Fitch on 1:104/435@FidoNet, 200:5000/211@MetroNet, or 13:7450/101@YouthNet. I will keep all information confidential, and use it only for purposes of maintaining REC.

Include the following information:

Your legal name
Your BBS system name
Your Primary Network name and address
Date of software installation
Name of system where you got the software from
City, State, and Country where your system resides

Beta Test Applications

The more diverse a group of beta testers is, the better testing a program will get before going out to the general public. I have a wide range of Beta Testers today, but am always looking for others.

If you wish to be a beta tester, please send a separate net-mail to the same address that you did for registration. Only registered user's will be permitted to become Beta Testers. Your application should include:

Your legal Name
Your BBS system name
Your Primary Network name and address
How long you have been running REC
Name/Version of your Front-End mailer
Name/Version of your Mail-Processing software
Name/Version of your BBS software
Your position in all the network's that you echo hub for
Format of your message base (Hudson, .MSG, etc.)
Any general comments that you have

I will try to get back to you as soon as possible, but I will make no guarantees on the time frame or who I will accept. I reserve the right to accept or reject any application with as much or as little explanation as I feel is needed. Please understand that not everyone can be a beta tester.

Beta Test Sites

I would be lost without all the helpful suggestions and <gasp> bug reports from my beta testers. In an effort to thank them, I have listed them here below in alphabetical order. I will apologize now for any mis-spellings I may make, and will correct it as soon as I am told about it.

Alex Cleynhens	Boortmeerbeek	2:292/500@FidoNet
Pete Franchi	Droitwich AB	2:253/157@FidoNet
Paul Fullerton	Denver CO	1:104/440@FidoNet
		200:5000/100@MetroNet
Michael Hughes	Pueblo CO	1:307/12@FidoNet
		13:7400/100@YouthNet
Joe Lindstrom	Calgary AB	99:9305/55@Eggnet
		200:5500/55@MetroNet
		111:1110/55@FantasyNet
Denis Marchand	Ayer MA	1:104/322/2@FidoNet
Jason Steck	Denver CO	1:104/424@FidoNet
		200:5000/400@MetroNet
Jeff Tensley	Denver CO	1:104/477@FidoNet
Charles Whitlatch	Phoenix AZ	1:114/115@FidoNet

Thanks again to you all!!

Implied Consent

If you do not agree to these restrictions, then do NOT use this software or make it available for others to access it. Use of this software indicates that you agree to these terms. Distribution of the software, or making it available for others to get it via any means of file transfers also implies consent and agreement to the terms specified in the

License for Distribution.

Trouble Reports

For technical support, you can contact me in one two ways. The easiest is to use the ZMAIL support echo. The moderator is Jason Steck and he has given me permission to use the echo for support of REC since it is tied close specifically with Zmail. The echo is available via either the MetroNet distribution system, or the FidoNet Echo Backbone. The second option is to use net-mail to any of my addresses in FidoNet, MetroNet, or YouthNet. The primary addresses in each of these networks are listed at then end of this document.

No matter how much testing is done, any software is bound to have glitches or bugs present. This includes the documentation. You may be asked to send your configuration and echo control files to the author at the name and address listed above for registration. Please be sure to supply the include the following information:

Complete hardware description
Compete software description
Which release of REC you are running
Any program terminations messages
Copy of relevant portion of the log file
Complete copy of your configuration file
A short narrative of the problem
Anything else you think might be needed
A voice number to reach you if necessary

Installation

Well, that is enough of the legal matters. Now let's get on to the business at hand and get this program working!

Definition of Terms

Just so everyone is using the same terminology, I will briefly describe the terms I will be using in this documentation. This is not intended as a short course, but rather as a translation.

ECHO - The actual message area that is passed from system to system.

ECHO HUB - The system that sends echo mail to your system for you to distribute.

ECHO NODE - The node receiving echo mail from your system.

LOCKOUT - Blocking an echo from receiving mail on an certain echo. This could be needed for several reasons, including Network Policy, local Net policy, or echo moderator decree.

CROSSZONE - Allowing certain echo nodes to access echos from zones other than there own. This is not very common, and is subject to a variety of policies in all networks involved. REC gives you the capability, and you have the responsibility.

Packing List

While you can place REC in any directory on your system that you wish, I suggest that you place it in its own directory. This will make program updates easier and keep your hard disk somewhat more organized.

The first step is to un-archive the distribution file into the directory that you wish to use. The original distribution file contains the following files:

REC.EXE - The main program
RECSIZE.EXE - Displays REC's memory requirements
REC.CFG - A sample configuration file
RECSYSOP.PRN - Sysop documentation, ready to print
RECUSER.PRN - User documentation, ready to print
HISTORY.PRN - Revision History, ready to print
BETA.DOC - Changes from the last beta (Beta Versions only)
RELEASE.DOC - Release letter for all users
NOTIFY.HDR- A sample notify message header
REPLY.HDR - A sample reply message header

The distribution file also contains a few little utility programs that you may find useful. They were written for the sysop of an echo hub, and are part of the REC package. Consider them a bonus. These utility programs are listed below and explained later in this documentation:

AREALIST.EXE - Alphabetical listing of echos
AREARPT.EXE - Echo Distribution Report
ECHOSOUT.EXE - Outbound echo mail tracker
READMSG.EXE - Formatted Display of .MSG file

REC is now on your BBS, and ready to be configured. Coincidentally, Configuration is the next topic to be discussed.

Configuration File

The configuration file for REC is a standard ASCII file that you can edit with any ASCII or Text editor. The example is broken down into 6 sections, but the actual order you use is up to you.

Certain entries are required, other entries must occur at least once, and some entries are optional. The entire configuration will be verified at the start of each run of REC, which only takes a second or two. The entries are case-insensitive, and the keywords must be followed by a single space. If an entry can have more than one field after it, these fields must be separated with a single comma. With the exception of SystemName and SysopName statements, any spaces in the field must be typed as an underscore ("_"). The program will replace them with spaces when the configuration file is processed.

There is a required order of some of the configuration statements. All Address statement must come before any other statement that have addresses, and your primary Address **MUST** be the first Address listed.All EchoHub statements must come before any EchoNode statements. Other than that, the order you specify the statements in is your own personal choice.

Each of the entries are described below in alphabetical order. They appear in several "logical" groups in the sample REC.CFG file. For examples, please refer to the sample configuration file found in the distribution file REC.CFG.

Required Statements

Address (MINIMUM 1)- This is a list of all the addresses that your system is known as. This list should include every address that your system uses to pass echo mail in areas that you want to control with REC. If you have addresses that you do NOT use for echo mail traffic, do NOT list them here. At best they will do nothing, at worst you will get wrong addresses on your outbound mail. Your primary address must come first, but other than that there is no real order required..

EchoControlFile (REQUIRED) - This is the complete path and name of the file that controls how the echos are distributed. Only one file can be specified and must be in the format shown below. All fields are separate by at least one space. All fields are required except the receiving addresses.

{location} {echo tag} {main feed} [receiving addresses]

Location - Can be any value desired, such as a directory name, board number, or Pass Through indicator. No spaces are allowed in this field.

Echo Tag - The name of the echo area.

Main Feed - Your echo hub's address for that echo

Receiving Addresses (optional) - The addresses that your send this echo to. This should be compatible with nearly all echo mail processors. If it is not compatible with your mail processor, get in touch with me so I accommodate your needs.

EchoNode (MINIMUM 1)- This is the definition of each of your echo nodes that you want to use REC with. You can have nodes listed in your Echo Control File that are not listed as an Echo Node, but only nodes listed as Echo Nodes will be allowed to use REC. The only optional fields are the 3 sysop names. The format is described below:

EchoNode {address},{echo hub},{password},{security}[,sysop name 1][,sysop name 2][,sysop name 3]

Address - The net address of the echo node being defined.

Echo Hub (Optional) - The net address of the echo hub that any forwarding echo requests should be sent to. This address must be defined with an Echo Hub entry.

Password - This the password that your echo node must use to get access to REC processing.

Security - This is the numeric security level (0 - 32767) that the echo node has. The echo node will only be able to request echos with an security level equal to or below its own security level.

Sysop Name 1-3 - You have the option of restricting REC access by name. The password checking is still active. If at least 1 name is specified, then the sender of the REC message must be listed here for REC to accept the message. If no names are specified, then the message will be accepted by REC no matter who the sender is. These fields are also used by some other parts of REC, so I strongly encourage to specify at least one name for each echo node.

MailDirectory - This is the directory that you will place net mail messages. REC requires that incoming net-mail and out-going net-mail be placed in the same directory. These messages must be in MSG format per Fidonet Technical Standard #001 (FTS-001).

SystemName - This is the name of your system, and will be placed on generated messages to your echo nodes and your own echo hubs.

SysopName - This is your name, and will be placed on generated messages to your echo nodes and your own echo hubs.

Optional Statements

AccessName - This is the name that you want your echo nodes to use when they sent a message to REC. "Remote Control" is assumed by REC ONLY if this parameter is NOT specified. If this parameter is used, then you will have to specify EVERY name that you want REC to respond to, including "Remote Control". A space in the name is shown by an underscore (_) character, as shown in the example in the sample configuration file.

AccessName {text}

CancelEmptyNotify - This option will tell REC not to send out any notify messages to any system that is not receiving any echos from you.

CrossZone - This command will permit echos to cross the zone boundary as long as the requesting echo node has at least the minimum security stated in this statement. The two flavor parts are optional and is discussed in the CrossZone section. The maximum number of statements is determined by how much RAM you have available. The format is shown below:

CrossZone {from zone},{to zone},{security}[,[node flavor,] [zone flavor]]

EchoHub - This is the definition of your echo hubs, which are your primary echo feeds. An entry is needed for any echo hub which you want to use automatic forwarding or Automatic Cleanup, which will be described in the Major Features section. The format is shown and described below.

EchoHub {address},{send to},{subject}[,type}

Address - The network address of your echo hub.

Send To - The name that you want to address the message to.

Subject - The text that you want to see on the message subject line.

Type (option) - Two possible values can be placed here: "REC" or "Areafix". If the field is omitted or has any other value, it will be ignored. This field is used only with Automatic Forwarding, and is described in the REC Processing Section. The default value will have messages addressed for a human being to read.

Valid Echo List - This is the name of any text file that contains a list of valid echos. The use and format of this file is covered under the section on Echo Tag Validation.

ExtraNotify - This statement allows you to notify nodes other than those listed as EchoNodes. If you specify the optional "To name" on the message, the message will be address to that person. Otherwise REC will put "Sysop" in to "To name" field on the notify message.

ExtraNotify {Address}[,to name]

Lockout - This allows you to specifically deny access to an echo by a particular node. No validity checking is perform on either the echo tag or the address. The maximum number of statements is determined by how much RAM you have available. If desired, you can use the "ALL" phrase for Net, Node, or PrivateNet fields. The format is shown below:

Lockout {echo tag},{address}

NoKillMsg - This will stop REC from deleting the messages is processes.

NotifyHeader - This does the same thing as ReplyHeader, except that it is used for any notification messages created by using the "/N" command line parm.

NotifyHeader {filename}

PassThrough - This allows you to specify the board that will designate a pass-through echo. The default value is "P".

PassThrough {character string 1-255 in length}

ReAddress - This option allows you to change this from-address on any inbound messages that REC processes. Please read the section on Re-Addressing before you use this option

ReAddress {inbound address},{new address}

ReplyHeader - This allows you to specify a "canned message" to be put on all replies created by REC. It only needs to be in straight ASCII text format, with NO ANSI characters.

ReplyHeader {filename}

Secure - This allows you to specify a security level on individual echos. The security level of an echo will default to 0 unless it is specified here. Valid security levels range from 0 to 32767. No checking is performed to ensure the echo tag exists on the echo control file. The maximum number of statements is determined by how much RAM you have available. The format is shown below:

Secure {echo tag},{security level}

SortBoard - This will make REC sort the echo control file by echo board number. It has the same sectional sorting capability as SortName. If two echo areas have the same board, then REC will automatically use a secondary sort key of echo tag.

SortName - This will make REC sort the echo control file by echo tag. All sorting is done between comment lines. In other words, if you use comments to break you echo control file in a section for non-echo areas, general echos, and private echos, all sorting will occur within the individual sections. The sections will NOT be merged.

StatusReport - This statement will cause REC to create and send to the Sysop a net-mail message every time REC is run in notify mode. This message will detail out the status of you echo control file and well as your echo nodes and echo hubs. This report is described in detail in a later section.

Address Specification

The format for putting an address in REC is quite simple. If you deal with point systems, then I suggest that you be sure to read the section on Point Systems for details on how REC handles them. On the example below, note that the only required field is the "node". All statements will follow this format:

```
[flavor][zone:][net/]node[.privatenet][/point]
```

In addition to the above, the Lockout statement will also allow you to specify an "ALL" in certain locations of the address. Several examples are show below

```
1:All                  (will lockout all of zone 1)
1:104/all                    (will lockout all of zone 1, net 104)
1:104/435.30537/all (will lockout my PrivateNet 30537)
1:104/435.All              (will lockout my PrivateNet 30537)
```

You should note that once REC finds the word "ALL" in the address, it will not read any more of the address. This means that the following statements will do the same thing, that is prevent anyone in Zone 13 from adding to the echo named Restricted_Echo:

```
Lockout Restricted_Echo,13:All
Lockout Restricted_Echo,13:All/101
Lockout Restricted_Echo,13:All/101.30537/all
```

Address Defaulting

The addresses that you specify will default in 3 different ways. This defaulting allows REC to determine address information if a piece is missing. I strongly urge you to specify the complete zone, net, and node for all addresses in the configuration file, and on all echo feeds in the echo area control file.

The first Address statement you specify will default using the address of 0:0/-1. Validation checks will be performed to make sure that the address does not have a negative number for any field.

Additional Address statements, Echo Hubs, Echo Nodes, Security, and Lockouts will all default using the first Address statement. You can see why it is important to have your primary address listed first. If you do not follow this rule, your mail will end up going to the wrong zones, if it even goes out all.

Addresses in the echo control file default using 2 rules. The Main Feed of the echo defaults to the first Address statement in the Configuration file. The first Receiving Address will default to the Main Feed, and all other Receiving Addresses will default to the previous address listed. Please examine the example below for an illustration, and assume that the first Address statement in the configuration file is for 1:104/435:

```
P  SYSOP  1:104/1 200 303/202 204 205 3:100/1 110/50
```

The main feed will be 1:104/1. The complete receiving address will be 1:104/200, 1:303/202, 1:303/204, 1:303/205, 3:100/1, and 3:110/50. This simplifies the echo control file, and REC will sort the receiving address in ascending order after processing, and write them back out in the "short form" shown above. Please note that since the complete address (including zone) was specified for the main feed, no defaulting was needed from the first Address statement. This is the safest way to proceed, and I highly recommend it.

Batch File Changes

To put REC in your batch files may require a bit of re-working of your inbound mail processing. One rule must be followed exactly: ALWAYS run REC before you import any net-mail into your message base. I highly recommend you process all net-mail before you process any echo mail. This way any changes will take affect as soon as possible.

A typical scenario would be similar to this sequence of events. First, you process all net-mail packets and create .MSG files, which are placed in your mail directory. Second, you run REC to process any messages for REC. Third you run your mail processor to import any remaining .MSG files into your message base. Lastly, if any .MSG files remain in the mail directory, run your net-mail bundler to make packets out of the .MSG files.

This scenario will be to be done whether you are processing non-compressed or compressed mail. This will ensure that all echo mail changes are complete before you processes any new echo mail.

Rec processing

At this point, REC is installed and ready to go on your system. Now is a good time to tell you how REC will be processing your change requests.

User Instructions

Included in the distribution archive is an file called RECUSER.DOC, and it contains the instructions for your echo nodes on how to use REC. I suggest that you read it right now. It will be easier for you to explain how REC works if you now what your echos nodes will be trying to do, and how they will go about doing it.

The user instructions also completely describe the various reports and messages that can be sent to the echo nodes. You will want to send this information to all of you echo nodes that you want to use REC.

Command Line Parameters

There are several command line parameters that REC will recognize. They are described in their own sections in this documentation, but I will list them here for your easy reference.

/B - processes a batch command file
/C - activates CleanUp Mode
/M - display memory usage (see Memory Requirements section)
/N - activates Notify mode
/R - activates Status Report generation
/? - displays syntax screen

Overview

Processing occurs is several steps, and is build on the assumption that only one REC message will be processed per run. This is NOT a limitation, but an observation from my own system. If you process net-mail immediately after receiving it, you are not very likely to process more than 1 message (unless the echo node sent more than one message at a time). If you save up net-mail until a specific time and than process them all at once, you will be more likely to processed several REC messages at one time.

REC will process the configuration file and verify it contents to the best of its ability, which mainly means syntax errors. Any errors found will be listed by statement name and sequence number, and at that point REC will terminate immediately.

If there are batch commands, .MSG files, Notifies, or Cleanup processing to be done, the echo control file will be loaded in to dynamic memory for fast processing.

If there are any batch command files to be processed, they will be loaded and processed at this point.

If there are any .MSG files to process in the Mail directory, they will be loaded and processed one at a time. It doesn't matter how many or how they are numbered, REC will check every single one of them.

The messages will be checked to see if they addressed to a valid address as shown in a Address configuration statement, sent to one of the listed AccessNames, and from a system listed as an EchoNode. If any of these conditions are not met, then REC will ignore the message and continue with the next one.

If the message is accepted, it will be checked for a valid password and authorized sender (if you specified one). Any errors will be noted on the screen and logged. In addition, a message will be sent back to the originating system, addressed to "Sysop" indicating that an invalid message was received and to contact the sysop running REC for details. No other information is given for security reasons.

Once the message has passed all verification, it will be processed. Echo areas will be added or removed, and any new areas will be checked for valid forwarding. Any reply messages and forwarding messages will be created at this time.

After all batch and message files have been processed, any notifies and status reports will be generated. Finally, any requested clean up processing will be performed, generating any necessary messages.

At this point, any file sorting (if any desired) will be done, and lastly the echo control file will be written out to disk.

REC does set error levels in DOS when it exits. If any messages are processed, a value of "1" is added to the error level. If any batch commands are processed, then a value of "2" is added to the error level. A full chart is shown below:

0	= Errors found, or no processing
1	= Messages Processed
2	= Batch Commands processed
3	= Messages and Batch commands processed

Inbound Message Addressing

While .MSG files to have zone and point fields in the message header, not all mail processors actually populate these fields. As such, REC has to do a series of searches (which are extremely fast) to determinate the proper originating destination address of the message.

Before anything else is done, any Re-Addressing indicating is performed on the loaded messages from-address. This feature is explained in the section on Major Features.

The first assumption tested is that the from-net and from-node in the message are actually correct. REC will search the list of EchoNodes you put in the configuration file for a match on Net and Node, and then will assign the first match it finds as the originating system.

If that assumption doesn't come back with a match, then REC assumes that the net and node on the message is really a private net and point. Then the same search is repeated for a match on any echo node that has a point address as it's address. Please be sure to read the section on Point Systems to be sure that you understand how points systems work. There are other assumptions and searches made for point systems, so you must be a little more careful with points.

For the destination address, the same set of basic searches are performed, except that no Re-Addressing is done on the to-address found in the message file.

If after this there is no match found for either the from- or to-addresses, REC will assume the message is not intended for REC processing and simply ignore it.

OutBound Message Addressing

OutBound messages are both simpler and more complicated to determine the proper addressing on the messages. The are three types of messages that REC can create: Replies, Echo Node Notifies, and Sysop Notifies. These messages are explained below.

Replies

This is the easiest of messages to address. REC is replying to a message that is was sent from another system. REC will simply switch the from and to address from the original message when it creates the reply.

Echo Node Notifies

These are the most complicated of messages to address. REC has an address to send the notify message to, but it doesn't know which address to send the message "from". REC will look for a match between the destination address and your Address configuration statements in two combinations: match on Zone and Net, or match on Zone. If no matches are found, then the message will be sent from your Primary Address (the first one listed in your configuration file).

Sysop Notifies

There are two sub-types of this message. The first sub-type is a status report and a Cleanup Processing report that REC can generate. These will be addressed to the Primary Address and sent to the person listed in the SysopName configuration statement.

The other type is more difficult. They are the Automatic Forwarding and Automatic Cleanup messages that can be generated. They actually follow the same logic as the Echo Node notifies since they are going to systems other than your own.

Memory Requirements

This section is not intended for everyone. Only those who run in a Multi-tasking environment (OS/2, Windows, DesqView) will be interested in this section, and then only if they need to determine the minumum amount of RAM needed to run REC.

I would recommend that you let REC run in the same amount of RAM as your mail processor. If you can run your mail processor then you should have any problems running REC. However, if you need to trim down the memory that you leave available to REC, then this section will help you.

REC is so flexible for several reasons, but the most important one is that is uses very little "Static" memory. Instead it uses "Dynamic" memory. This choice produces large flexibility, but it does offer a few disadvantages as well. What this means and how it relates to optimizing REC will be discussed in this section

Memory Definitions

If you already know the difference between "Static" and "Dynamic" memory allocation, then you can skip over this part. Otherwise, here is a short course just for you!

Any program written today is intended to manipulate data. This is true for everything from a database to a bulletin board to a game. The difference is in what they do with the data, and sometimes how the allocate the memory needed to store the data.

Static Memory Allocation

The most common method is to use "Static" memory allocation. With this type of memory, the amount of memory needed is explictly defined to the program. For example, an address book program would have defined room for only so many lines in the address book. This type of memory is accesses very quickly and is easily manipulated any way the program wants.

There are two drawbacks to static memory. First, there is a limit of how many items you can have. In the example above, the address book may have a limit of only 50 lines to allow for the "average person". If you want to add a 51st person, you will have to delete a different person from the book. This limit can be changed only by changing and re-compiling the program code.

The second drawback is that there is usually a waste of memory. If the address book program was designed for the largest user, then it would have a 500 line limit. However, if you only use 100 lines, then you are wasting 80% of the space allocated to the address book. You will have loaded a program that needs a large amount of your computer's limited memory, and that memory will not even be used by the program. If each line were to have room for a the usual 6 line address, then each line would take up 240 chacters (40 characters per line). Multiply that by 400 lines and you get 96,000 characters of your limited memory is just WASTED!

Dynamic Memory Allocation

The alternative to static allocation is to use "Dynamic" allocation. If we continue the example of the address book, the program would define each line of the address book as a certain number of characters, say again the 240 characters. Then each time you want to put an address in to the book, the program would go out the the available memory and get just 240 characters. If you delete a line in the address book, the 240 characters is given back to the computer's memory so some other program can use it. This type of memory allocation means that the program uses only exactly how much memory is needed, and not a single character more.

As with anything, Dynamic allocation does have drawbacks. First, it take more program code to handle the memory safely, properly, quickly, and efficiently. Perhaps the greater of the drawbacks is that if the program tries to grab more memory than is available, the program will abort execution right in the middle of running. This can leave your files open or only paritially updated. This can result in a large loss of data.

Analyzing Memory Usage

REC is designed so that the echo control file and log file are only updated at then end of the program. This way, if REC should abort while running, your echo control file will not be trashed beyond use. You will have lost some updates, but a few updates can be recovered easier then the entire file.

If you REC with the "/M" option, you will get a report indicating how much memory REC has dyanmically allocated. At the end of the program, there will be a display on the screen with these lines:

```
Memory Check:
  Start: xxxxx
 Config: xxxxx
  Echos: xxxxx
    End: xxxxx
   Diff: xxxxx
```

The amount of "Start" memory is the amount of available memory when REC starts running. The "Config" memory is how much memory is used to load the configuration file. The "Echos" memory is how much memory is used to load the Echo Control File. The "end" memory is how much memory is available when REC is finished running. The "diff" memory is the difference between the "start" and the "end" memory.

REC releases every single byte of dyanmically allocated data before it ends. Therefore the difference should always be "0", and the "end" memory should always be the same as the "start" memory.

You must realize that, at this time, REC does not report how much memory has been used to process the messages and/or batch command files. Notification, CleanUp, and Sysop Report production utilize hardly any, if any at all, of dynamic memory. If you add the Config and Echos memory together, you will get a good idea of how much memory REC dynamcially allocates.

Predicting Memory Needed

Determining just how much memory REC needs is not easy. First, you have to add together the program size (REC.EXE), the config memory, and the echos memory. Then you have to add enough additional memory to handle the largest batch and message file that you expect to receive.

To help you do this, there is a program called "RECSIZE" in the distribtuion package. When you run this program, it will tell you how much memory is need for each of the various types of configuration and commands that REC has. You can use these numbers plus a calcluator to determine how much RAM is needed by REC for dynamic allocations.

After you arrive at that number, then you need to add some more memory for the actuall logic processing inside of REC. I would recommend that you allow at least 30K for this.

The other method of doing this is just to keep running "Trial and Error" with different amounts of memory until REC won't run anymore. If you do this method, then MAKE SURE you are working with a test copy your echo control file. Otherwise you may end up damaging your data.

As I said before, the easiest method is just to run REC in the same amount of memory as your mail processor. This is usually what will happen in most cases. However, if you enjoy testing the limits, then by all means go ahead and experiment.

Major Features

There are several major features that REC has. Each of these features has its own advantages, restrictions, and requirements. Most of the time, all you will have to do it either add a configuration statement, or add a command line switch to activate the feature. However, you should know what the particular feature will and won't do for you.

Readdressing Inbound Messages

Unfortunately, not every mail processor in use today is zone intelligent, or even zone aware. You may run in to a situation where a system has echo mail in a zone other than their primary zone, but all mail will come addressed from there primary zone. Let's consider the following example.

You send echomail to 200:5000/211. REC will always use that address to send mail to that system. However, that system is not using a zone-aware mail processor. All message sent to you arrive from 1:104/435. Since REC isn't looking for that address, it won't process any change requests. Also, if you were to define the echo node to REC as 1:104/435, you have to deal with Crosszone issues for it to get the zone 200 echos. This becomes a real mess in big hurry.

REC's job is to automate and simply your life, not make it worse. As such, the ReAddress statement is used. You put the following line in your configuration:

```
ReAddress 104/435,5000/211
```

Notice that I only specified the net and node for the address. That is because only the net and node are actually in the message header itself. If you specify anything else, such as zone or privatenet or point, they will be ignored.

When REC reads in the message from 104/435, it will change the address to 5000/211 and process it that way. Please note that the change is made internally to REC and not written out the .MSG file itself.

If you need to re-address a point, instead of specifying the net and node, you would specify the private net and point. For example, say all mail from 7450/101 is to be re-addressed as one of my points. The following line would be added to my configuration file:

```
Readdress 7450/101,30537/55
```

REC will internally change the from address to 30537/55, then realize it is one my points and process it correctly.

Any re-addressing that is done is noted on the screen and in the log for your reference.

Areafix Compatibility

There are 3 reports that the user can request. The are Active Echo Report (:A command), Available Echo Report (:L) command, and Forwardable Echo Report (:F command).

The Active Echo Report is a listing all echos that the echo node is currently receiving. The Available Echo Report indicates every echo available to the echo node that is ALREADY being received by your system. The Forwardable Echo Report is a simply a copy of the Valid Echo Control file, if one exists for the EchoHub that the EchoNode is associated with.

Areafix allows certain parameters to be sent on the subject line of an echo node's message. These parameters are supported by REC. However, REC produces 3 reports while AreaFix will only produce one report. Below is how the codes will be processed:

-R	Active Echo Report
-Q	Available Echo Report

More information on these reports is available on the User Documentation for REC. Please refer to that documentation if you have any questions.

Automatic Forwarding

I have mentioned Automatic Forwarding a few times, so now I will explain this neat little feature. You don't have to be an echo hub for long before one of your echo nodes asks for an echo that you are not receiving from your own echo feed. This feature will allow you automatically pass on a request to your own echo feed to start the echo. All that you have to do is specify an Echo Hub address on the Echo Node statement for each system that you want to use automatic forwarding. The address on the Echo Node statement much match the address of one of the Echo Hub statements.

When an echo node requests an echo that is not found in the echo control file, the list of Echo Hubs is searched for a match on the echo hub address found on the echo node statement. If a match is found, a message is sent to the echo hub requesting that the echo be activated. The fact that the echo request was forwarded is noted on the reply message to the echo node and the log file. If a matching address is not found or an echo hub address is not specified on the echo node statement, an appropriate message is noted in the reply and on the Log File.

If desired, you can have REC search a list of valid echos before forwarding the echo request. Please see the next section on Echo Tag Validation for more information.

This Automatic message can be sent to either a human or another automatic echo control system, and this is controlled by the Type field entered on the Echo Hub. At this point, REC can send messages to another REC (of course) or Areafix. There is only 1 difference between a message sent to a human or an automated system. A message to a human will have a short blurb at the beginning of the message asking to have the listed echos activated. Both types of messages will have a tear line ("---") followed by a short line informing the recipient that the message was created automatically by REC.

The receiving address, receiving name, and subject line are all populated from the Echo Hub entry. To have a message to an AreaFix program, put "Areafix" (of whatever your echo feed is using) as the Echo Hub's send-to field and the password on the subject field. Remember you have to specify the class as either AreaFix or REC to either of those programs. It can't really be much easier.

Echo Name Validation

People create the messages that are sent to REC, and people do make mistakes. One of these mistakes that I found tiresome was a misspelled echo tag on a request to start an echo feed. If a tag was misspelled then REC would forward the echo request on to your feed, and create a new echo tag in your echo control file. Once you found out the echo tag was invalid, you would have to go back in and drop the echo tag.

That problem has been solved! You now can put an optional parameter on the EchoHub statement, a text file name containing a list of valid echos for that hub. If an EchoNode requests an echo that you are not already receiving, this list will be searched for that echo tag. If the tag is found, then the echo will be created locally and forwarded to the EchoHub. If not, then the request for that echo will have an result message indicating that the echo isn't valid.

The format of this Valid Echo List is simple. It must a be straight text file. The first word on each line is considered a valid echo tag, and everything else on the line is ignored. The end of the first word is marked by a comma, space, or end of line. The list must be sorted alphabetically before it can be used, which is no problem with the DOS sort program.

Please remember that if no file name is specified, there will be no validation. If the file is not found or is empty, no validation will be attempted. In either case that no validation is performed, the echo request will be forwarded and created under the assumption that the echo tag is valid.

This validation can be done for any system, either automatic such as Areafix or REC, or a manually controlled system.

Cross-Zone Echos

Allowing echos to cross zones was added for those gateway systems that do this regularly. There are policy rules and guidelines that must be adhered to when echos start crossing zones, and I have no intention of addressing them here. REC will assume that you know what you are doing if you employ this feature.

The Cross Zone statement was described in the configuration section. Here are a few tips for easy use of the Cross Zone feature. First off, don't use it unless you need to use it. Beyond that, set the security for crossing zones above that of what you normally assign to echo nodes that do NOT cross zones. If you have any echos that should not cross zones, set their security higher than any of the Cross Zone statements.

A word about Cross Zone and Automatic Forwarding: An echo node cannot have forwarding for multiple echo hubs. If you have a system in a zone that is getting echos from more than one zone, you will have to either disable automatic forwarding for that echo node (by not specifying an echo hub address on the Echo Node statement), or allowing automatic forwarding for only one of those zones. I realize that this could cause some problems for any gateways that cross several zones. If it does, please let me know and we can discuss a better solution.

When echos cross-zone using this feature you can force Zmail flavors on either the echo node, or the echo feed. This is mostly used when running a gateway system. The first flavor code will be put on the echo node requesting the echo. The second flavor code will be put on the echo area's feed. You can use either, both, or neither. Please look at the examples shown below:

```
CrossZone 8,1,3000
CrossZone 8,1,3000,H
CrossZone 8,1,3000,,*
CrossZone 8,1,3000,H,*
```

The first example will allow an echo node in Zone 8 to access an echo from Zone 1, providing that the echo node has at least a security of 3000, and the echo has a security of less than or equal the echo node security.

The second example does the same as the first, but will force the "H" flavor code (for "Hold") on to the echo node.

The third example does the same as the first, but will force the echo area feed to have "*" flavor code, which will have Zmail leave these messages as *.MSG files. Do notice that there are two commas between the security and the Feed Flavor code. That is not a typo, but a requirement.

The fourth example will do all of the above with just one statement.

For more information on Zmail Flavors, see the next section.

Zmail Flavors

Zmail Mail Processor (Copyrighted by PROZ SOFTWARE) allows special processing of the outbound flavors for your echo mail. You can have the mail marked Crash, Hold, Normal, or allow it to default. You can also have the echo mail not bundled but left as .MSG files in your mail directory for another program to work on. The actual rules and use of these options are left up to the Zmail documentation.

However, they are completely supported by REC. Any character other than a number, color (:), slash (/), or period (.) is consider a special processing flag (or flavor code). You can up to 10 symbols on a single address, and they can be put on any address you need to. They also can be used with the FLAVOR batch command and the CROSSZONE configuration statement.

Some default logic is imposed with these flavor commands. If you put any flavor on an echo for an echo node, it will stay there until you remove or change it. If it has no flavor when the echo node is added to the echo (via a message or batch), the flavor codes will be taken from the following places in this order:

Batch Command Line

Cross Zone Statement

Echo Node or Echo Hub Statement

If you wish to have all the echo mail for a certain echo node to have a certain flavor, just add that flavor to the address part of the Echo Node statement. Every time that echo node adds an echo, the flavor on the Echo Node statement will be used. This works for Echo Hubs as well as Echo Nodes.

Batch Mode

Overview

There will be times that is will be easier and faster for you to submit an update to REC than to wait for the echo node to send mail. Also there will be certain things that an echo node cannot do, such as change your echo feeds around. Batch mode allows you to do this with a simple text file and a command line parameter.

Batch mode is optional for the Sysop. You may feel it faster and/or easier to make the change directly to the echo control file yourself.

However, batch mode is reported in the log file which provides an audit trail. It can also be used to perform certain updates at a particular time such as nightly maintenance.

Perhaps it single greatest advantage is in transferring an echo node from echo hub to another. You could just setup an event to run REC with your special batch command file at a particular time, and let REC make the necessary changes while you are sleeping comfortably.

Commands

To use batch mode, first you create a simple text file with any text editor. In this text file you just list the commands you want to execute with any needed parameters. The complete list of available commands is shown below:

Add - This command allows you to add an echo node to an echo tag.

ADD {tag},{echo node}

Remove - This command allows you to remove and echo node from the distribution of an echo. The special tag for all tags can be used (see below).

REMOVE {tag},{echo node}

Create - This command allows you to create a new echo in the echo control file. To add echo nodes to this echo, you would need to use an "Add" command, which would have to be coded after this statement. If you use the literal "*_Local_Only_*" instead of the feed, REC will create the area as a local or non-echo message area. This is needed for off-line message readers such as XRS to operate properly.

CREATE {tag},{board,{feed|"*_Local_Only_*"}

Drop - This command allows you to remove an entire echo from your echo control file. It is completely deleted from the file.

DROP {tag}

Flavor - This command allows you to set flavor to an echo for a particular echo node. Any flavor specified on the address directly will be ignored. The flavor must be placed after the echo node. To remove all flavor from the tag and echo node, use the special flavor of "*_NONE_*". The special for all tags can be used (see below).

FLAVOR {tag},{echo node},{flavor}

Board - This command allows you to change the board of an echo. You must specify the tag and the new board value.

BOARD {tag},{board}

Tag - This command allows you to change an echo tag. You must specify the current tag value, and the new value you wish to use.

TAG {current tag},{new tag}

Feed - This command allows you to change the feed for an echo.

FEED {tag},{new feed}

There is a special tag name that will cause the command to work on any echo tag. This can only be used on Remove and Flavor, for obvious reasons. The special tag value is "*_ALL_TAGS_*" and is not case sensitive.

If you create the batch command file with the name BATCH.REC in the same directory as REC, it will automatically be run with the next run of REC. Otherwise, you will need to execute REC with a command-line parm of /B followed immediately by the filename of the text file you just created. An example would be nice, and it follows below:

```
REC /Bmyfile.txt
```

In the above example, the file named "myfile.txt" will be read and processed as a batch file. If the filename cannot be found, an error message is displayed and logged. The batch file is processed first, and the commands are processed in the exact same way as if they came in from a net-mail message.

You should remember one thing about Batch commands. It is assumed that the sysop know what they are doing. The Lockout, Security, and Cross-Zone statements are not considered at all when you use these batch commands. This give you both maximum power and control, but also means you must be sure that you are doing what you mean to do.

Notify Mode

This is a quick and simple way to let your echo nodes what echos they are receiving. This helps to make sure that your echo node is getting only the echos desired, and all the echos desired. During this Notify Mode, REC will still process any batch command files and inbound messages addressed to REC. Most often, you would use this mode once a week in a maintenance mode. An example is shown below:

```
REC /N
```

This above example show the command line parm of "/N". This forces batch notify mode, and will send an Active Echo Report to every Echo Node listed in the configuration file, and only to those addresses.

If desired, you can have REC create a sysop's status report during notify mode. This report is described next.

Status Report

The Status Report is can be generated by two different means. If the StatusReport configuration option is used, the report will generated when REC is run in Notify Mode (/N parameter, see previous section). It can also be generated by running REC is Report Mode. While running in report mode, REC will still process any batch command files or inbound messages. An example is shown below:

```
REC /R
```

The status report is sent as net-mail to the address shown in the first Address configuration statement, and address to the name listed in the Sysop configuration statement. This report is broken down into 4 major sections consisting of 3 summaries and a dead-end echo listing. The first summary is a summary of how many echos you have listed in your echo control file. A "Local Area" is not really an echo area, but is listed in the echo control file as required for an off-line message base reader such as QMX or RaQSeX. An "Imported Area" is an echo area that is imported into your local message base, essentially not a pass-through echo. A "Pass-Through Area" is an echo area that you get from one of your echo hubs and pass on, but do not import into your own message base. A "Dead-End Area" is an echo that area that your get from one of your echo hubs, don't import, and don't pass on. Usually you would want to stop these since they take up time and disk space for no gain.

The second summary is an Echo Hub summary. It lists each of your echo hubs and how many echos are feed from that source. It ONLY reports on the echo hubs you have listed in your configuration file.

The third summary is an Echo Node summary. It lists each of your echo nodes and how many echos they are receiving. It ONLY reports on the echo nodes you have listed in your configuration file.

The last piece of the report does not always show up. It is the listing of each dead-end echo and its feed. This would allow you to issue cancel requests to the appropriate feed for the appropriate echos. Any echos listed with a "**" next to the tag are controlled by manually controlled hubs, or hubs not running either REC or AreaFix. Any echos listed without this marking could be automatically cancelled by using the Automatic Cleanup option, which is discussed in the next section.

You should realize that while the Echo Area Summary reports on every echo in your echo control file, the Echo Hub and Echo Node summaries do not. The total number of echos reported in the Echo Hub summary will equal the total number of echo areas ONLY if every one of your echo mail feeds are listed as an echo hub in your configuration file.

Automatic Cleanup

This was a major function that I felt was needed in order to completely automate the control of your echo mail traffic. This feature will automatically find, cancel, and drop any echo that being sent to your system but is not be sent to any other systems and is not being imported. To activate this mode, you use the /C command line option, as shown below:

```
REC /C
```

The way this works is simple, but you may need to read this section twice to understand it. It sounds far more complicated then it really is (I wish I could say the same thing for the program code!!). There are two parts to the CleanUp processing. The first part will drop any echos already cancelled, and restart any cancelled echos that are no longer dead-ends. The second part will cancel all new dead-end echos found. The link between the two parts is a control file. The second part writes a control file listed all dead-end echos found and cancelled from the automatic hubs. The next time REC is run in cleanup mode, the first part will read in this control file, and use it either drop or restart the cancelled echos.

I STRONGLY RECOMMEND THAT YOU RUN CLEANUP MODE ONLY ONCE A DAY, IDEALLY DURING NIGHTLY MAINTENANCE. This means it becomes a permanent part of your nightly maintenance routines. Running cleanup mode on a weekly basis would result in a week delay in either dropping or RESTARTING a cancelled echo.

If this option is implemented incorrectly, you can drop several of your echos from your control file. Please read this section carefully so this will not happen. I will give both an explanation and examples in the following paragraphs. Since the process really starts with the second part, I will start there.

First, REC search's the echo control file looking for any echo that is 1) a dead-end, and 2) controlled by automatic hub. A message is sent to each hub necessary, instructing the hub to stop sending you those echos. A status message is sent to you indicating that the echos were cancelled.

Example: You run REC in cleanup mode during nightly maintenance. During cleanup mode, REC finds that you are getting two dead-end echos from hub #1, SPORTS and FICTION. A message is sent to Hub #1 with the necessary commands to cancel these echos. The echo tags and the appropriate hub address are written to a control file. You are sent a status message listing the echos as dropped. During the following day you should receive a reply from the echo hub that the echos were stopped.

The next time REC is run in cleanup mode, it reads in this control file. If the hub is still automatic, the echo is still a dead-end, then the echo is dropped from your echo control file.

Example: The next night you run REC in cleanup mode again. It finds and loads in the control file showing SPORTS and FICTION were dropped from hub #1 the previous night. However, during the 24 hours between the two cleanup runs, one of your echo nodes requested and was connected to the SPORTS echo. A message is sent to Hub #1 re-activating the SPORTS echo, and the FICTION echo is dropped from your echo control file. You are sent a status message indicating that the SPORTS echo was restarted, and the FICTION echo was dropped.

There are some important things to remember. First, if you can a hub from automatic to manual between the two cleanup runs, and an echo needs to be restarted, you will get a message on the status report that an echo needs to be manually restarted. Any echos that should be dropped will be dropped as normal.

Second, if a echo node should be added to one of the dead-end echos cancelled the previous night, the echo will not be restarted from the echo hub until the second cleanup run. In other words, during normal processing, echos are not checked to see if they need to be restarted if a echo node should request an echo.

Lastly, all of this is noted in REC's activity log. You are also sent a message indicating what was cancelled, dropped, or restarted. Please note that you will only receive this message if there actually was something cleaned up.

I have tested the <beep> out of this routine because it can be dangerous if implemented incorrectly. It delayed the beta test release for version 1.20 by nearly an entire month. I have put more logic and user error traps in this piece of code than any other feature of REC. It works perfectly if you implement it correctly. Just put "REC /C" in your nightly maintenance routines and you will have no problems. Do not mess with the control file. And, if you have any echo hub that are not automatic, send them a copy of Remote Echo Control with my compliments and you recommendation <grin>.

It has saved my own system plenty of disk space, and myself plenty of time in the month I have been using it in Alpha test. I highly recommend it use.

Point systems

Point Systems are also called Private Nets or Point Nets. In its simplest form, a point system is not much more than a private network of BBS's. It is a fact, however, that humans have become very good at making simple things complicated.

The major difference between PrivateNets and "Public" Nets is that point systems don't appear in the nationally distributed nodelist. The Boss Node of a point system is basically the door by which its dependent points talk to the rest of the world. All other BBS's just send the mail to the Boss Node, and the Boss node passes on the mail to the individual point systems. The point systems send all their mail to the Boss Node, which then passes that mail on the to the rest of BBS world. As far as echo mail is concerned, you are just passing it along like echo mail to any other BBS. Net mail takes a little more work, but there are a couple of different ways to handle that, all of which are beyond the scope of this documentation.

The addressing of a point system of perhaps the most confusing part. There are 3 main ways to address a point system: private net, point reference, or the complete full address. Take my own system, for example. My FidoNet address is 1:104/435, and my private net (or point net) is 30527. I could address my fifth point (or private BBS) is any of these ways:

1:30527/5 (called private net)
1:104/435.5 (called a point reference)

1:104/435.30527/5 (the complete or "long hand" style)

The second style, or point reference, is all that the rest of the BBS world would need to know to get a net-mail message to one of my points. Echo mail is passed down via the echo control file, just like it would be to any other BBS. Now for the specifics of how REC handles point systems.

REC will (of course) accept any of these 3 formats of addressing a point system. The important point to remember is this: However you list the point system's address in the Echo Node definition, is how it will be handled in the echo control file. Where ever possible, Point systems will appear using the short-hand notation that I will describe next.

Assume that I receive the echo tag ABC from my local NEC, and I pass it off to two other systems (210 & 550) and two of own point systems (5 & 15). Here's what you would see in the echo control file if I used point referencing or complete addressing for my points when I setup the EchoNode statements:

```
P  ABC  1:104/1 210 435.5 435.15 550
```

If the second point listed (435.15) was only shown as "15", it would actually be sent to 104/15 instead of my point.

The same line would like this if I used private net addressing:

```
P  ABC  1:104/1 210 550 30527/5 15
```

As you may have noticed, the private net addressing looks just like address for net 104, except that the net number is 5 digits long instead of 3.

It is your preference as to which method you want to use. What you should remember is this: If you use the PrivateNet addressing style, your mail processor most likely will NOT strip the Private Net address from the seen-by and path lines. This may be against the policy of either your Net or Network's policy.

If you would like a recommendation from myself, then I have three of them. First, be consistent in how you define all of your points. Second, use the Point Reference method instead of the Private Net or Complete Address methods. Third, put your private net in you address statement, such as 1:104/435.30527/0

If you put your private net number in your Address statement, then REC will be able to tell that an EchoNode defined as 435.10 is really 30527/10. The searches for this are done automatically, and is one of the reasons why you need to list all your Address configuration statements before any other addresses. This way REC will sent any net-mail to 30527/10 instead of forcing you to use a program such as ReMapper to do the dirty work. See the example below:

```
Address 1:104/435.30527/0
EchoNode 104/210,,Test1,100,First_Joker
EchoNode 435.5,,Test2,100,Second_Joker
```

All notifies and replies will be sent to First Joker on 104/210, or to Second Joker on 30527/5. This type of setup gives you maximum flexibility to do this in which ever manner you want. Just be consistent so you don't start to confuse yourself.

Security

I wish there was a world in which no form of security or protection was needed. Unfortunately, there will always be those type of problems which will required a form of security.

However, REC's security is designed to prevent accidental errors instead of intentional mischief. Even with this design in mind, the mischief can be easily prevented using the security features designed in REC. These feature will be described and explained below.

Security Levels

The very basic means of security is via the security level placed on an echo node and the echo itself. You give all your echo nodes a basic security level, and any echos that require special permission, you put a higher security level on the echo.

For example, say there was a group of echos that were not for general distribution. These would include restrictions to Admin Hubs, Net Coordinators, Region Coordinators, and Zone Coordinators. You could assign a series of security levels as shown below:

```
Secure Admin_Hub,2000
Secure NC_Talk,3000
Secure RC_Talk,4000
Secure ZC_Talk,5000
```

With this setup, you would then give a standard security level of 1000 to all of your echo nodes, with exception of the ones that are allowed to access the Admin_Hub echo you would give a security level of 2000.

The advantage is that once you set this up, it becomes automatic. All you have to do is change an Echo Nodes security level and everything else is automatic.

Of course, there is a single drawback to this scheme. Any system that you high enough security level to access the NC_Talk echo will also be able to access the Admin_Hub echo. The same goes for anyone given security to access the ZC_Talk echo. That system would be able to access all of the other secured echos with lesser security level. It is up to you to decide if this is a problem or not with the echos you wish to Secure.

Echo Lockouts

This is more absolute security measure than security, but it does tend to require more manual effort to maintain. With this method you specify that a specific echo is not allowed to either a specific system or groups of systems.

Let's use an example similar to the one above. Assume I am the NC of my particular net. I need to access the Admin_Hub echo and the NC_Talk echo, but I have read-only access to the RC_Talk echo. I would use the following statements:

```
Secure Admin_Hub,2000
Lockout NC_Talk,1:104/all
Lockout RC_Talk,1:104/all
Lockout ZC_Talk,1:104/all
```

In this example, only other admin hubs that I have defined with at least a security level of 2000 would be able to access the Admin_Hub echo. However, none of my other echo hub, regardless of their security level, would be able to request the NC, RC, or ZC echos.

If you are wondering why I included the ZC echo even though I should not even see it, please read on to the part on Echo Forwarding in this section. You will find out why I did that, and it was very intentional.

Zone-Crossing Restrictions

Some echos are allowed to cross zones (or networks) with no problem. Others are restricted to crossing only through approved gateways. Others are restricted to not crossing zones at all.

You can use the security level on the CrossZone statement to control this. Any echo node with a security level lower than the CrossZone security for the zones involved will be denied any access to the "foreign" echos. If there are only certain echos that are not allowed to cross zones, then you can either give that echo a higher security level than the CrossZone statement, or use a Lockout statement.

Echo Forwarding

Any echos not found in your current echo control file will be checked for automatic fowarding. This is explained in the section on Major Features.

However, you may be aware of certain echos that are restricted from your echo nodes that are not in your echo control file. In this case, if any requests were forwarded for those echos, those requests would be honored and some un-authorized system would have full access to an echo that they were not supposed to get.

If you put these echos in a Lockout or Secure statement, the automatic forwarding would catch these echos and reject them. This will work even if you don't have the echo listed in your own Echo Control File.

Error Messages

REC can return many different messages, and not all of them are errors. In this section, I will list all of the error conditions, and where they are reported. Everyone of these errors/messages will be sent to the log file and the display screen. I will not be covering the configuration errors, since they are well detailed by the program error message if any should be encountered.

Message Processing

First I will cover those messages that can be produced by a message sent to REC from an echo node:

'Not processed': The command was not processed by REC due to an error condition. You should not see this message, but it was added to cover all possible logic flaws in the program where a message command might be skipped.

'Added': The echo was activated for the echo node.

'Already active': 'The echo was already active for that echo node.

'Insufficient security': The echo node lacked sufficient security level to access that echo.

'Locked out': The echo node has been locked out of that echo via the Lockout statement in the configuration.

'Zone-crossing refused': That echo node is not allowed this echo since the echo would be crossing a zone. If a CrossZone statement was involved, it was because the security of the echo node was less than the minimum specified on the CrossZone statement.

'Not found, was forwarded': The echo was activated for the echo node and forwarded to the echo hub.

'Not active': The echo node was not receiving the specified echo so it could not be removed.

'Removed': The echo node was removed from the echo.

'Invalid report code': A report was requested, but the report code was not one recognized by REC.

'Active echo list': This report was created.

'Available echo list': This report was created.

'Not found, not forwarded': The echo did not exist and could not be forwarded.

'Not found, invalid echo': The echo did not exist on the Valid Echo File associated with the echo hub.

'Forwardable echo list': The list of Valid Echos for the echo node's hub was sent in a separate message.

Batch Processing

Listed next are the error messages that can be generated by a batch command:

'Not Processed': This is the same as the one above.

'Processed': The command was processed correctly.

'Tag not found': The echo requested was not found.

'Node not found': The node request was not found.

'Tag already exists': A tag could not be created because that tag was already being used by an existing echo.

'Node already active': The node was already active for the echo.

'Invalid Command': Either the command was unknown or one or more of the parameters were missing.

CleanUp Processing

Finally, I will list and define the messages that can be produced by CleanUp Mode processing. Each of these messages is preceded by an echo tag and echo hub address.

' Cancelled': The echo was cancelled from an automatic hub.

' Dropped': The echo was dropped from the echo control file.

' Not found': An echo tag & echo hub combination found in the cleanup control file was NOT found in the echo control file. This can be caused by manually dropping the echo in between cleanup runs, or changing the feed of an echo in between the cleanup runs.

' Restarted': The echo was restarted from the indicated echo hub.

' Manual': The echo needs to be manually restarted for a manually control echo hub. The hub was reclassified as manual in between cleanup runs.

Bonus Utilities

As a free bonus, I have included a few simple little utility programs that I find useful on my system. The command syntax can by found by simply executing the program without any command line parameters. I will give the syntax here as well, along with a short description of what the program does.

AreaList

This program takes an echo control file and produces a 3 column, alphabetically sorted, listing of all echo tags found. No consideration is given for separating by zones. An optional second report can be generated, a list of echos tags in the format required by for the Valid Echo List file.

```
AREALIST {echo control file} {list filename} [Valid Echo List filename]
```

AreaRpt

This program will produce a report from your echo control file. The report is a listing of every system found, and which echos that system is receiving from you. Is is sorted alphabetically in a 3-column format.

```
AREARPT {echo control file} {report name}
```

ReadMsg

This is a quick and dirty public domain utility that basically displays the formatted contents of a .MSG file.

```
READMSG {msg filename}
```

EchosOut

This program is designed to be put right in the REC directory. It keeps track of how much echo mail you send out. You should run it immediately after you process any received echo mail, and after you process any echo mail generated on your system. It reads the outbound directory given, updates a small tracking file called ECHOSOUT.TRK, and writes out a sorted report. The tracking file will be created/update in the current directory. This report lists for each system found the system address (with out zone) and the total daily volume in Kybtes for the last 7 days. A Zone total and Run Total are also generated.

You should note a few things. Always run the program from the same directory, so it can always use the same tracking file. To reset the report, just delete the tracking file. Make sure the address on the command line parm is your PRIMARY address but do NOT give the zone. Only specify the outbound directory for your primary zone, meaning without an extension. The program will automatically check for the other directories.

The program only takes a few seconds to run, and it real useful watching how much echo mail your system in handling.

```
ECHOSOUT {outbound directory} {net/node} {report file}
ECHOSOUT m:\out 104/435 volume.txt
```

Size

This is a simple utility that is essential for anyone attempting to run REC is the smallest amount of memory possible. This is often the case in Multi-tasking environments. A full description of the utility is in the section on Memory Requirements. To execute the program, just type "Size" and you will get a one screen display.

Conclusion

Well, that's about all I have to say about REC and the bonus utilities. If you have any questions, feel free to contact me via net-mail to any of the addresses listed below. Send the message to Dan Fitch, or to Sysop.

I have several plans in mind for Version 2.0 of REC. Some of these include a full-screen on-line door program to allow you to directly examine, search, and update the echo control file while on-line while maintaining an accurate log of what you did.

Best of luck in your BBSing, and my thanks for using this software.

Later!

Dan Fitch
Author of Remote Echo Control (REC)
Sysop of The Rec Room
1:104/435@FidoNet
200:5000/211@MetroNet
13:7450/101@YouthNet
Denver, Colorado
revised 1/18/23.