XEROX
BUSINESS SYSTEMS
Systems Development Department

To: Distribution
Date: September 20, 1978

From: Hal Murray
Org: SDD/SD System Architecture

Subject: Alto Gateway Operation
Filed: [Iris] < Murray > Memos > AltoGateway.bravo

This is my first attempt at modifing Ed Taft's memo titled "Nova Gateway Operation" to describe how to run the new Alto Gateway/Router program. Comments and such are desired.

This memo documents the operation of the Alto Gateway program. You are assumed to be reasonably familiar with Altos. Familiarity with Mesa will be very helpful. You might also want to read Ed Taft's memo titled "Nova Gateway Operation". It is stored on [IVY] < Portola > GatewayOperation.bravo.

I have avoided any details of the current network topology since it keeps changing so often these days. There are two documents that are very helpful when trying to track down network troubles. They are [MAXC] < AltoDocs > AltoNetwork.press and [MAXC] < SYSTEM > Pup-Network.txt. I suggest that you keep a reasonably up to date copy handy.

1. Normal Operation

Power Up and such

The only problem with turning a Gateway Alto on and/or off is setting the time if there is not another Gateway connected to the normal Ethernet. If the Alto asks you to set the date and time, the trick is to use text such as in "Jul" rather than a number for the month. If you get really frustrated, SHIFT-SWAT will also get you past this problem.

Starting the Gateway Program

Type the command Gateway followed by carriage return. The program will, read the parameter file, print out a few messages about boot files, print a few informational messages, pause for a while, and finally print the message "Gateway of *date-time* in operation". At this point the Gateway program is running. However, it takes approximately one minute for new routing information to be propagated throughout the inter-network, so it may not be possible for connections to be established through the Gateway during the first minute after the Gateway program is started. About a minute after the program is started, the display will get turned off.

The Alto Display

The display is turned off shortly after the Gateway program is started. This conserves CPU cycles, and saves the core that would be needed by the bitmap. Normally the screen will be all black. If you type anything on the Keyboard, it will reappear. If the disk is active, the screen will switch to all white. This happens while the Gateway is sending a boot file, searching the name lookup data base, or receiving a new copy of a file. If you want to see if the Gateway is alive, watch the cursor, or hit the space bar.

Whenever any character is typed in, the display will be turned on for 30 seconds. If the Gateway program is running in debugging mode, the display is turned on for 5 seconds whenever it prints out an informational message. Normally, it stays off, and the messages can be found in **Mesa.typescript** if you are interested.

The cursor will move one step to the right for each Pup that is forwarded, 50 steps left for each Pup that cannot be forwarded, and one step down for each Pup received by the Echo server or the Miscellaneous Services server.

If the Gateway has an SLA line driven by an EIA board, there are troubles keeping the clock accurate when the display is on. To avoid sending out inacurate information, the Pup Time Server is disabled whenever the display is turned on. It is automatically reset when the display goes off.

Gateway Program Commands

The Gateway program has a simple command interpreter that responds to single-character commands typed on the keyboard. Whenever a character is typed, the display is turned on for 30 seconds.

The commands are as follows (the "?", or any unrecognized character elicits this list):

Gateway Statistics

Prints out a summary of various operating statistics, including the length of time Gateway program has been running (hours:minutes:seconds), the number of times each server has been invoked (explained later in this memo), and a matrix showing number of packets forwarded from one directly-connected network to another. The number of discarded packets is also listed. Discard of packets is not cause for alarm: it is a normal consequence of the great disparity in speed between the Ethernet and the leased lines, and does not give rise to loss of information in file transfers or terminal connections.

SLA Statistics

Prints out operating summaries for the Synchronous Line Adaptor (SLA). For each line, the number of packets successfully sent and received on that line is printed, followed by the number of instances of three types of errors: CRC (Cyclical Redundancy Check) errors, Sync errors (bit synchronization was lost), and Control sequence errors. The total number of errors should be much less than one percent of the packets successfully received. If a high error rate occurs on a single in-use line (whose state is "Up"), the line or modem is suspect, whereas if frequent errors occur on all lines (or particularly on "Looped back" lines), the SLA interface is suspect.

This command also prints out the SLA routing table. Under normal circumstances, the routing table for a particular Gateway should show that it can reach all other Gateways through one or more of the connected lines.

Reset Time

The Gateway program maintains the current date and time, which it gives out to other hosts that request it. The date and time are obtained from another Gateway when the Gateway program is started. The **Reset Time** command causes the local date and time to be invalidated so as to force the Gateway program to reset itself from another Gateway. This is necessary if the local time has become incorrect.

Probe for New Directory

A data base of host names and addresses is maintained at Parc and distributed to all

Gateways for use in responding to name lookup requests from Altos. The data base distribution procedure is automatic, with requests for data base update generated once per hour. The **Probe for New Directory** command simply forces such an update to occur immediately.

Local Routing Table

This command prints out the routing table used by the Pup Software to determine where to send a pup to get it to its final destination. 0 hops means that this gateway is directly connected to that network by the net and host number indicated. If the hop count is greater than 0, then the pup will be forwarded to another gateway at the indicated address for further processing.

Echo

There is an Echo user program built into the Gateway. It is just like the one in PupTest. If you are Echoing to your self, the $ won't get printed, but the cursor will move down one step each time a packet gets echoed.

Debug

This command causes control to be given to the Mesa Debugger (if there is one on the disk). From within the Debugger, the Gateway program is resumed by typing **P** (for Proceed) and confirming with a RETURN.

Toggle Display

This command toggles a lock on the display. If The display was off, or on temporatily, it will be turned on, and the 30 second timer will be disabled. If it was on, it will be turned off.

X

This belches forth reams of statistics that are probably only interesting to programmers and/or hardware debuggers.

<SPACE BAR>

This is a dummy command that just turns the display on for 30 seconds.

Boot File Table

This just prints the table used by the Boot Server. It includes counters so you can see which files are really used in case the disk gets full.

Cache for Name Lookup Server

This prints out the current contents of the cache maintained by the Name Lookup server.

Quit

Terminates the Gateway program and causes control to return to the Alto Executive, which responds with an @.

Bugs

If the Gateway program gets into trouble, it will probably print a message on the display and wait for a person to do something. At this point, there are two options. If you have

the Mesa Debugger on your Gateway disk, you can get to it and poke around. Normally there won't be room for it. In that case, you can only get to SWAT. I think I can do a reasonable job of diagnosing bugs if you are careful to save SWATEE an the right time. The trick is to be sure the the right information is there, and to avoid clobbering it. If the Gateway program is waiting for you to hit SWAT, do so. That will write the current core image into SWATEE. The Alto will then go to either SWAT or the Mesa Debugger. In either case, reboot it. ↑K or SHIFT-SWAT will overwrite SWATEE. Then FTP SWATEE away to a safe place. Remember that FTP only knows how to talk to machines that are connected to the normal Ethernet board. If you don't have an IFS handy, you can store it on another Alto until you get the Gateway back up again.

There are three general categories of troubles. The first is a problem during initialization. Hopefully, the text will be sufficient to solve the problem. The most likely cause is a bug in **GateParameter.txt**. If you run into a message that is too cryptic, please let me know, and I will fix it. The second is a uncaught SIGNAL. It will be printed in octal, and a listing of **Gateway.signals** (from [IRIS]<MesaGate>Gateway.signals>) will probably help translate it into English. The third catagory of trouble is an inconsistancy discovered by the Pup Software. Again an octal number will be printed, and you will need an up to date copy of DriverDefs.mesa to translate it into English. Even if you can translate it, it probably won't help you much, but it will help me.

2. Functions Performed by the Alto Gateway

The primary purpose of the Gateway is to forward Pups (Parc Universal Packets) from one network to another. However, the fact that the Alto is somewhat underutilized by this task and that it is in operation all the time makes it a desirable source of other services as well. These services are necessarily limited to those that are relatively easy to implement and whose resource requirements don't significantly impact the primary role as a Gateway, but they make life considerably more pleasant for Alto users than would be the case were these services not available.

The Gateway program provides the following services:

Gateway Information

This service is actually related intimitely with the Alto's role as a Gateway. It provides routing information to all hosts on directly-connected networks. It is by means of this routing information that Pup software such as FTP is able to communicate with hosts on other networks.

Date and Time

The Gateway program maintains the date and time in a form usable by Altos. The SetTime EXEC command obtains the date and time by this means.

Name Lookup

This service translates inter-network name/address expressions into addresses usable for communication. When an Alto subsystem is requested to establish contact with a host addressed symbolically (e.g., "Maxc"), it generates a name lookup request, to which the gateway responds by supplying the corresponding inter-network address (e.g., "3#200#").

Boot

Altos are capable of boot-loading over the Ethernet if a suitable server is available to provide the boot files. The Alto Gateway provides this service for a limited set of boot files that are useful on an Alto with no disk loaded or with a disk that won't boot (e.g., DMT, Scavenger, FTP, CopyDisk, and a NetExec providing more convenient access to the other

boot files). There is no intention, however, of making available a complete set of Alto subsystems by this means.

Echo

This service consists simply of a process that echoes all received packets back to their source. It is useful for hardware and software diagnosis.

In addition to these services, the Gateway program also implements several internal support protocols, such as the one that automatically updates the network directory data base.

3. Contents of the Gateway Disk

The Alto Gateway disk contains all of the files needed for normal operation of the Alto Gateway. Since space is tight, it probably does not contain much else.

Gateway.image

The Gateway program itself. This file is updated when a new version of the Gateway program is released.

GateParameter.Txt

A parameter file that specifies the configuration for this particular gateway. All Gateways run the same Gateway program, and differences in configuration are dealt with by this parameter file. The specifications in this file include:

The inter-network addresses (network and host numbers) of all network interfaces installed in the machine.

An empirically-determined correction for regulating the clock in software.

The association between boot file numbers and names.

This file is likely to be changed only to add new boot files or adjust the clock correction.

MesaGateEIA.br or **MesaGateCP.br**

This is the microcode loaded into the RAM if the parameter file specifies a second Ethernet board or an SLA network.

PupNetwork.Directory

The local copy of the inter-network name data base. The master copy is [MAXC]<System>PupNetwork.Directory. This file is updated automatically by the Gateway program. There must be one before the Gateway program will start up.

DMT.Boot, Chat.boot, NewOs.Boot, FTP.Boot, Scavenger.Boot, CopyDisk.Boot, NetExec.boot

These are Alto bootstrap files, given out by the Gateway program when an Alto is boot-loaded over the Ethernet. These files are updated occasionally when new versions of the programs are released at Parc. They have been "reformatted" so that they are NOT useable on the Gateway Alto. NB: SYS.boot is needed to get the Alto off the ground. It should NEVER get reformatted. That's why SYS.boot is called NewOs.Boot.

Gate.Scratch$, NewGateway.image

Gate.Scratch$ is a temporaty file used while getting a new version of **PupNetwork.Directory** and during remote updating of files. There must be enough room on the disk for it to hold a copy of the biggest file that will ever be remotely updated. **NewGateway.image** is a temporary copy of the Gateway program used when remotly distributing a new verrion of the Gateway software. It is needed for remote updating and automatic restarting because Mesa uses **Gateway.image** for code swapping, and is automatically deleted by the Restart command from GateControl.

SYS.boot, Swat, Swatee, Executive.run, FTP.run (and whatever)

These are just the normal files/programs needed to get any Alto off the ground. Note that you will probably need FTP to update anything on the Gateway disk if the Gateway program is busted since you can't boot it over the Ethernet.

Mesa.typescript

Mesa.typescript is a typescript file of everything that gets printed on the Alto display. It is reset whenever the Gateway program is restarted. The Gateway program will crash if it stays up long enough to fill up the disk.

RunMesa.run

RunMesa.run is the bootstrap program that loads the Mesa microcode into the RAM if you don't have a ROM1, and then loads **Gateway.image** into core and starts running it.

Xdebug.image, MesaDebugger, Debug.typescript, Windex.bcd, Fetch.bcd

These are the Mesa Debugger. They will be on your disk only if it is setup for debugging -- there isn't room for them normally. NB: Do not move or delete any of them unless you know what you are doing. They contain FPs.

Gateway.symbols, Mesa.symbols, BufferDefs.bcd, DriverDefs.bcd, ...

These are the symbols tables for the Mesa Debugger. Again, they will be on your disk only if you are chasing a nasty bug.

4. Gateway Disk Maintenance

The Gateway software includes the capability for remote updating of files on the Gateway disk. It is also possible to remotely command the Gateway to restart, to reset its date and time, and to perform other operations. We use this remote control capability at Parc to release new versions of the Gateway program and to update the boot files. Therefore the following procedures should not be needed in the ordinary course of events.

Note that you are in deep trouble if your last Gateway disk gets clobbered. It is a very good idea to keep a backup copy. You can easily make one using CopyDisk.run. A Gateway disk can also be configured to run on any Alto. It won't be able to forward packets to any other network, but it will provide luxuries like a a Time server and a Boot server which make working on Altos much more pleasant.

Obtaining Updated Files

This section describes procedures for obtaining new Gateway software or other files from Parc and installing them on the Gateway disk. This involves stopping the Gateway program briefly, so one should first make certain that there are no users who might be affected.

The following procedure assumes that the file being updated is Gateway.image, but it applies to any

other file. The procedure requires use of a second Alto connected to the same Ethernet as the Gateway Alto.

1. While the Gateway is still in operation, run FTP on an second Alto, connect to IRIS, and retrieve the file <MesaGate>Gateway.image. After closing the connection to IRIS, leave the Alto running FTP. The boot files come from [IVY]<Portola> and [MAXC]<Alto>.

2. Type the **Quit** command on the Gateway keyboard to return control to the Alto Executive.

3. Run the FTP program on the Gateway Alto, connect to the second Alto, and retrieve Gateway.image. Note that since no name server is running at this point, it is necessary to refer to the second Alto by number rather than name (e.g., if it's Ethernet address is 123, you connect to "123#"). After retrieving the file, quit out of FTP to the Alto Executive.

4. If desired, back up the Gateway disk.

5. Restart the Gateway program by issuing the **Gateway** command.

One might wonder why it is not possible simply to connect directly to Maxc from the Gateway in order to retrieve files. In order to do this, FTP would have to communicate over the SLA line to Parc. However, FTP (and other standard subsystems) does not contain the software for driving the SLA interface, but only contains a driver for the Ethernet. Hence, FTP running on the Gateway can communicate only with machines on the directly connected Ethernet.

5. PupTest Operation

PupTest is a program developed primarily for checkout of the Pup software; however, some of its capabilities are helpful in locating and troubleshooting network problems. **Be sure to keep a few copies of PupTest.run on various disks because you can't boot it from the Gateway when you really need it.**

We describe here only those PupTest commands useful for troubleshooting network problems.

The usual symptom of failure is likely to be that a user at a remote location is unable to access Maxc or some other machine in Palo Alto. There are a multitude of problems that can cause this, and one should not jump to the conclusion that the leased line or the Gateway is down. The following procedure should pinpoint the failing link.

The PupTest command used in this procedure is **Echo**. It requests the name or address of a host running an Echo Server, and it then sends Echo requests to that server. All Gateways have Echo servers, and any machine running PupTest itself has an Echo server. For each reply PupTest receives, it prints "!", and if no reply is received, it times out after 1.5 seconds, prints "?", and tries again. The test terminates when any key (e.g., space) is pressed.

1. Run PupTest on two Altos connected to the same Ethernet, and direct one of them to Echo to the other. One should address the other Alto by number rather than name; e.g., if the other Alto's Ethernet address is 123, one should say "Echo to: 123#". If this doesn't work, then the local Ethernet is broken.

2. Echo to the local Gateway; that is, say "Echo to: *n*#", where *n* is the Gateway's address on the local Ethernet. If this doesn't work, then the Gateway program is down or the Gateway or its Ethernet interface is broken.

3. Echo, in turn, to each of the other Gateways along the path from the local Gateway to the unreachable destination. The topology is geting pretty complicated now, so you will have to be careful doing this. (It should be possible to reference these machines by name since the name lookup is performed by the local Gateway, which is known to be up by step 2.) If this doesn't work, then there are a number of possibilities, some of which can be

checked out by running PupTest on the Gateway (see step 4). If it is possible to echo to Portola, then the most likely reason for inability to reach a host in Palo Alto (e.g., Maxc) is that the host itself is down.

4. Unplug the line to the modem and put in an echo plug. Run PupTest on the Gateway Alto. (You will need a reasonably up to date version of PupTest. It didn't learn about SLA lines until August of 1978.) Be sure to have an accurate GateParameter.txt on your disk since it tells PupTest to startup the SLA Driver. Direct PupTest to Echo to itself through the looped back line; that is say Echo to: 7#n# where n is the assigned SLA network address for your machine. The normal behavior in this case is for "!" and "$" to be printed alternately. ("$" means that the Echo server, running on the same machine, replied to an echo request.) If this doesn't work, the SLA interface is probably broken. Be sure to plug the modem back in.

The wirewrap EIA board needs a patch for this to work: add a wire from B10 pin 11 (SYNCLK) to J3 pin 35. (I haven't seen a PC version yet, so I don't know if it has this change.) The CommProc needs a backplane jumper from J11 pin 108, for 9600 baud, or pin 111 for 57KB, to pin 105 of the slot containing the LIM driving the line. That brings an internal clock out to pin 18, which is normally unused, on the modem connector. An echo plug (its female) should have pin 2 (transmit data) connected to pin 3 (receive data), pin 4 (request to send) connected to pin 5 (clear to send), pin 6 (data set ready) connected to pin 20 (data terminal ready), and pin 18 (internal clock) connected to pins 15 (transmit clock) and 17 (receive clock).

5. After plugging the modem back in, press the AL (Analog Loopback) button on the modem so that it stays in its "in" position. Again, direct PupTest to echo to itself through the SLA driver. If this doesn't work, the modem is probably broken. When you are finished with this test, be sure the modem's AL button is in its "out" position.

You can use the Echo command within the Gateway program rather than PupTest. If you do, it is probably a good idea to lock the display on with the T command first. To force Echoing to go through a particular line, unplug all of the others.

6. If the previous test worked, but it was not possible to echo to Portola (step 3), then the failure is in the Gateway or modem at the far end of the first failing link or in the leased line itself. Determining which component has gone down requires cooperation with personnel at the far end.

6. Notes to Gateway Maintainers

When updating Gateway.image via GateControl, the Gateway program automatically changes the name of the incomming file on the Gateway disk to be NewGateway.image. The GateControl Restart command looks for this filename. The running program is swapping code out of Gateway.image, and it will probably die horribly if the code were changed out from underneath it. If a file already exists when an update starts, the incomming data is stored in a temporary file until it is all collected to protect against clobbering things if the EFTP transfer fails. If a file does not exist when the update starts, the incomming data is written directly into the target file to avoid filling up the disk with two copies. If the transfer fails, the partial file is deleted.

Be sure to keep at least free 300 pages on your Gateway disk after Gateway.scratch$ has been deleted. If you don't have enough there won't be any way to remotely update big files. Actually, if you really need a few more pages, you can borrow some from a big boot file by storing DMT.boot into it.

The Restart command appends a few commands into Rem.cm, so you can do almost anything you want to a Gateway disk remotely by storing your own commands in there ahead of its restart sequence. If NewGateway.image exists when the Gateway is being restarted, Gateway.image will be deleted and NewGateway.image will be renamed into Gateway.image. At the end is a command to

run Gateway.image.

There is a way to run other programs along with the Gateway. If the file GateRun.txt is on your disk, the Gateway program will process it after things get started. It should contain lines of the form *xxx*.bcd followed by a carriage return. For each line, the Gateway program will load the bcd, and FORK to its control module. The Gateway program EXPORTs GateDefs, and all the Pup interfaces, so you can easily link to its innards. There is a sample module that uses this feature. It is called TimeChecker, and it will ask Portola for the initial time, wait about 24 hours, ask Portola for the time, and print out the difference. If the time has not been reset in the meantime, this should provide a correction factor for your clock.

We use this feature to run the PacketRadio interface. If you need special microcode, you will have to cooridinate things carefully.

If you are using this feature, you should be careful when updating your BCDs remotely. One correct recipe is to first store away a garbage GateRun.txt. An empty file doesn't work yet. I use Com.cm which will hold "GateControl.run". The Gateway program will recover from missing files, and that is a filename that is not likely to be found on a Gateway disk. Then, restart the Gateway, store your updated BCDs, and then restore the correct contents to GateRun.txt, and again restart the Gateway.

7. Hardware Requirements

To run the Alto Gateway software requires appropiate hardware. Here is a quick checklist:

First, you need a 7th build or newer Alto. These are the wide-bodied ones. You don't actually need any extra memory, but it is much easier to make the necessary modifications on the newer Altos. MRT must be made Ram-Related. Connect MRTACTIVE (slot 11, pin 109) to either TASKA' (slot 10, pin 13) or TASKB' (pin 14) whichever isn't being used by other devices. The CommProc uses TASKA'.

If your Gateway is going to talk to a phone line, you will need either an EIA board(s) or a CommProc. An EIA board costs about $1200 per line. It does not seem to run faster than about 20KB, but I don't know why it doesn't. A CommProc costs about $4500 plus $500 per line. It works ok at 50KB.

If you are going to connect to more than one Ethernet, you will have to aquire and install extra Ethernet boards. See [IVY]<Portola>ExtraEther.bravo. An Alto can handle up to three Ethernets.

An Alto disk is quite full if you have the normal collection of boot files. If you want a large collection, a second disk drive will be necessary.

Distribution:
Ed Taft, David Boggs, Larry Stewart, Ted Strollo (Parc)
Steve Butterfield (ASD)