northgate
INFORMATION SOLUTIONS

Series X Heartbeat

2.x

Reference Manual

Document No. UM70006394A
May 1996

Northgate Information Solutions UK Limited
Peoplebuilding 2
Peoplebuilding Estate
Maylands Avenue
Hemel Hempstead
Herts
HP2 4NW

Tel +44 (0)1442 232424
Fax +44 (0)1442 256454

www.northgate-is.com

Chapter 1 About this Manual

Chapter 2 System Description

Chapter 3 System Administration

Chapter 4 Administering Gateway Loadsharing

List of Figures

List of Tables

Chapter 1
About this Manual

This chapter describes the purpose and content of this manual, other related manuals and conventions used in the text.

Purpose of this Manual

This manual provides the information necessary to configure and maintain a Series X Heartbeat system, release 2.0 onwards.

Chapter 1, **About this Manual**, describes the purpose and content of this manual, other related manuals and conventions used in the text.

Chapter 2, **System Description**, provides an overview of Heartbeat operation. It describes the overall configuration and the Heartbeat software on the FailSafe hosts and Heartbeat gateways.

Chapter 3, **System Administration**, details administrative procedures for setting up, maintaining and recovering normal Heartbeat operation. They include procedures for: configuring Heartbeat connections to a database or C server program on the hosts, starting up and shutting down Heartbeat, configuring a Heartbeat system, recovering normal Heartbeat operation after a Host/DBMS failure, recovery after a master gateway failure, re-covery after a slave gateway failure and toggling primary/secondary databases.

Chapter 4, **Administering Gateway Loadsharing**, discusses the operation of loadsharing by Heartbeat gateways, for both TCP/IP and OSI networks, and details the parameters which need to be defined in order to configure the software.

Appendix A, **Heartbeat Routing Information**, discusses the use of internal route names in a Heartbeat configuration and the structure and setting up of database configuration information (DCI) for a Heartbeat system.

Appendix B, **hbadm Utility**, describes the purpose, syntax, options and restrictions on the use of the **hbadm** utility.

Appendix C, **hbprocess Configuration File** shows the contents of the configuration file used by **hbprocess**.

Appendix D, **hbmonreal Configuration File,** shows the contents of the configuration file used by **hbmonreal**.

Appendix E, **hbmonsys Configuration File** shows the contents of the configuration file used by **hbmonsys**.

Appendix F, **hbarbiter Configuration File**, shows the contents of the configuration file used by **hbarbiter**.

Appendix G, **hbadm Configuration File**, shows the contents of the configuration file used by **hbadm**.

Appendix H, hbmondisk Configuration File, shows the contents of the configuration file used by **hbpulse**.

Appendix I, **hbuser Script and HBPROC**, contains examples of the **hbuser** script and HBPROC.

Related Documents

Series X Heartbeat can be used in conjunction with RealityX FailSafe software Release 4.0 or later. The following related manuals may therefore be required:

Reality X Resilience Reference Manual

Reality X Reference Manual Volume 1: General

RealityX Reference Manual Volume 2: Operation

RealityX Reference Manual Volume 3: Administration

ENGLISH Reference Manual (Vol. 4)

PROC Reference Manual (Vol. 5)

EDITOR Reference Manual (Vol. 6)

Screen Editor Reference Manual (Vol. 6)

DATA/BASIC Reference Manual (Vol. 7)

General Utilities and Printing (Vol. 8)

UNIX-Connect System User Guide

UNIX-Connect System Administration Guide

Conventions

The following conventions are used in this manual:

`Text`	Bold text shown in this typeface is used to indicate input which must be typed at the terminal.
`Text`	Text shown in this typeface is used to show text that is output to the screen.
Bold text	Bold text in syntax descriptions represents characters typed exactly as shown. For example **hbmenu**
Text	Characters or words in italics indicate parameters which must be supplied by the user. For example in **hbrouter -d** *application* the parameter *application* is italicised to indicate that you must supply the name of the actual application in the configuration file defined on your system. Italic text is also used for titles of documents referred to by this document.
{Braces}	Braces enclose options and optional parameters. For example in **hbmenu {***options***}** one of two options can be included, as defined in the command description (Chap 3) or **hbmenu** can be executed without any options.
Enter	To enter means to type text then press RETURN. For instance, 'Enter 'a'' means type **`a`**, then press RETURN. In general, the RETURN key (shown as ENTER or ↵ on some keyboards) must be used to complete all terminal input unless otherwise specified.
Press	Press single key or key combination, but do not press RETURN afterwards.

X'nn'	This denotes a hexadecimal value.

User Comments

A Comment Sheet is included at the front of this manual. If you find any errors or have any suggestions for improvements in the manual please complete and return the form. If it has already been used then send your comments to the Technical Publications Manager at the address on the title page.

Chapter 2
System Description

This chapter provides an overview of Heartbeat operation. It describes the overall configuration of a Heartbeat system and the Heartbeat software located on the FailSafe hosts and Heartbeat gateways.

Overview

Heartbeat is a software facility which operates in conjunction with a Series X FailSafe system, such as RealityX FailSafe, to enhance the resilience and usability of the system. Heartbeat is for use with applications that require a high level of availability.

A Heartbeat system consists of two Series X host computers operating in FailSafe mode and connected to the user network via Heartbeat switch software located on two or more Heartbeat gateway computers.

Heartbeat enhances FailSafe resilience by monitoring the 'well-being' of the host systems, their resident DBMS applications and the Heartbeat LAN connections. If the live host/primary database fails, Heartbeat automatically switches users to the standby/secondary system automatically, re-configuring it as the primary. Users are logged on automatically to the new primary database and returned to the first screen of their application without any user action being required. Users are therefore able to continue working normally on the system/database with minimal disruption to service.

To benefit from the automatic switch-over facility, users must be connected to the host by a Heartbeat gateway. Router software on each gateway manages the connections to the hosts without users being aware of the gateway, or needing to know the identity of the live host, or the location of the primary database. When a user logs on to a gateway, a router process is executed which establishes the connection to the correct primary. In the event of a primary failure, the router processes execute the switch-over of users to the standby host automatically.

Primary databases can be located on either of the two hosts in the Heartbeat configuration. In the event of a failure, only primary databases on the failed host are switched. Primary databases on the surviving host are unaffected and service to users on that machine is uninterrupted.

Multiple gateways are supported, each one routing a pre-defined number of users. Loadsharing software optimises the user load across the gateways. See Chapter 4.

Non-Heartbeat users, that is, users that do not benefit from Heartbeat automatic switch-over, are routed directly to the live host via the user LAN. See the topic *System Configuration*. Similarly, all out-going connections, such as printers and NET-LOGONS are made via the user LAN. If a failure occurs, non-Heartbeat users will disconnect, hang or time-out (if network time-outs are enabled). They must then disconnect from the failed system, if necessary, identify the surviving system and logon again. This is the same re-connection procedure as for ordinary FailSafe users. Printers etc. must be disconnected and re-connected to the surviving machine All outgoing connections from a host must be made directly on the User LAN.

System Configuration

A Heartbeat system comprises:

- Two Series X host systems operating in a FailSafe environment with one or more primary/secondary pairs.
- Two or more Series X gateways which route Heartbeat-protected users to the primary database.
- Four separate local area networks (LANs) with different sub-net addresses which interconnect the Heartbeat system. Each Series X machine has three Ethernet controllers to support this configuration. See also Appendix A.

A typical Heartbeat configuration is shown in Figure 2-1.

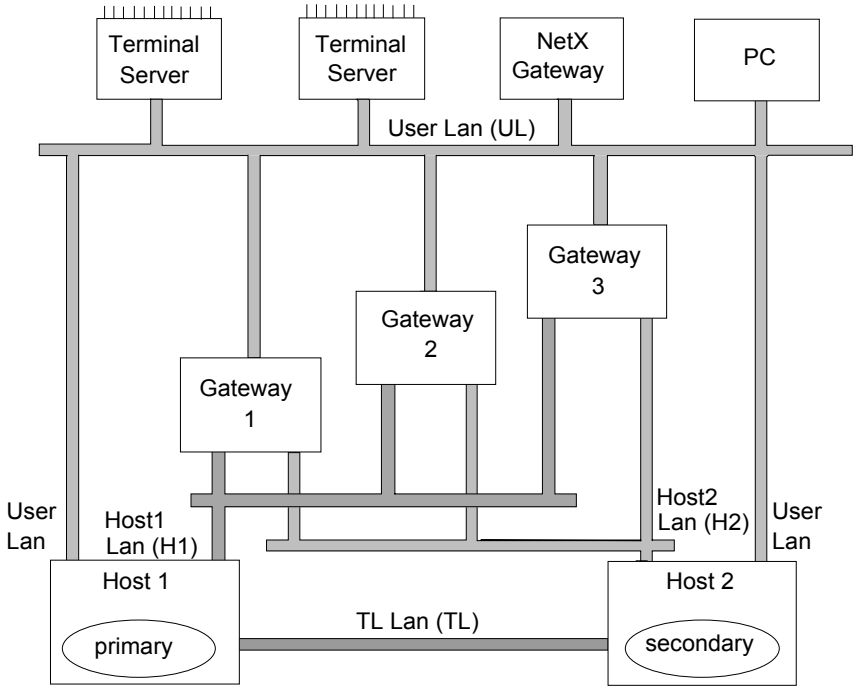


Figure 2-1. Typical Heartbeat Configuration

The four LANs interconnecting the system are:

- **Transaction Logging LAN (TL)**. A dedicated link between the two FailSafe hosts for transferring logged updates to the secondary.

- **User LAN (UL)**. The network connecting Heartbeat users to Heartbeat gateways and non-Heartbeat users directly to the hosts.

- **Host LANs (H1** and **H2)**. One LAN per host connects all Heartbeat gateways to that host. Separating the user LAN and host LANs avoids the duplication of network traffic and improves network resilience.

This Heartbeat configuration ensures that there is no single point of failure.

Only users connected to the FailSafe system via a Heartbeat gateway are protected by Heartbeat from a system or DBMS failure. Multiple gateways are supported, one of which is designated the 'Master'. The master gateway executes the arbiter software to monitor the hosts and initiate switch-over in the event of a failure.

To ensure a level of resilience if a gateway fails, a minimum of two gateways are necessary, so that if the master gateway fails, the surviving gateway can be re-configured as master. Any number of gateways can be used as determined by the user load requirements. Adding extra gateways increases the throughput which is optimised by Loadsharing software.

System Operation

Error Monitoring

Each host in the Heartbeat/FailSafe configuration contains a number of executables and files which co-operate to monitor the local host system, resident DBMS applications, Heartbeat gateways connected to it and Heartbeat LAN connections.

Monitoring a Host

The host system and resident DBMS applications are monitored via the standard UNIX error reporting mechanisms. So long as the system and DBMS application(s) remain healthy, the Heartbeat process on the host sends a regular stream of Heartbeat 'Alive' pulses to one of the gateways called the master gateway which runs the fault arbiter software.

If the Heartbeat process detects a fatal error in the system or a DBMS application, it sends a 'Fault' message to the master gateway which gives the reason for the failure. Alternatively if the host crashes, hangs or suffers a disabling error condition, the stream of Heartbeat 'Alive' pulses is interrupted, indicating that the machine has failed. In this case, a 'Fault' message cannot be sent to the master gateway. Instead arbitration software on the master gateway detects that a specified number of pulses have been missed and generates a 'Fail' message which it sends to both hosts.

Monitoring the Gateways

Heartbeat gateways are monitored by each host sending an Acknowledge flag with each minor tick 'Alive' pulse to the gateways. Refer to the topic *Heartbeat Pulse Generation*. If the gateway software fails to respond to the Acknowledge flag, the Heartbeat process on the host assumes that the gateway has failed. It then marks the database configuration entry for that gateway as 'Failed', and runs a script to display a failure message on the system console.

Monitoring LAN Connections

LAN connections in a Heartbeat system are monitored by two LAN monitor processes running simultaneously one on each host. Both LAN monitors periodically probe the LAN system by attempting to make remote UNIX shell connections across all LAN links in the Heartbeat system. The success or failure of each attempt is recorded and the results of all connection attempts from all points in the system are compiled in a table and compared.

If one or more remote connections are unsuccessful, the type and location of the fault is diagnosed from tabulated results. One of three types of fault is identified, host fault, gateway fault or LAN fault, and the location of the faulty host, gateway or LAN is determined. One of three recovery scripts for a host, gateway or LAN fault is then run on the host. Depending on the type of fault, the recovery script marks the host, gateway or LAN status in the database configuration information (DCI) as 'Failed', and send appropriate fault reports to the system console.

Database Configuration Information

Before a Heartbeat system can operate, the database configuration information (DCI) must be set up on each host and gateway machine and be consistent across all machines. The DCI data is required by Heartbeat processes on the gateways to enable them to establish the primary and standby user connections to the FailSafe hosts.

Initially, the DCI data must be entered by the system administrator using the **hbmenu** utility as part of the configuration procedure described in Chapter 3. This can be carried out on either of the host systems . The DCI is entered in the local host's **dci.file** and is then propagated to all other hosts and gateways in the Heartbeat system. Identical DCI data is therefore held in a **dci.file** on all hosts and gateways.

The DCI includes the name and status of each host, gateway and LAN, and the names of the FailSafe applications running on the hosts, together with the names of associated databases and their host systems.

An application name is an arbitrary name used to identify a FailSafe database pair. The database names and host names are entries in **/etc/ROUTE-FILE,** used by UNIX-Connect to make the user connections to the live and standby hosts.

Making a Heartbeat User Connection

A Heartbeat user connects to the FailSafe hosts by logging in to a Heartbeat gateway using a pre-assigned Heartbeat user-id. Logging in to a gateway using the Heartbeat user-id causes an **hbrouter** process to be run from the user's **.profile**.

The **hbrouter** process creates and maintains the Heartbeat user connection. To do this it requires two parameters, the application name to connect to, and the user-id to logon to the host. These parameters can be supplied in the **hbrouter** command line in **.profile**. If they are not supplied, then **hbrouter** uses the application name $LOGNAME and the UID number as defaults. Refer to Chapter 3 for a description of **hbrouter.**

Having been given the application name, **hbrouter** determines the database to connect to by interrogating the database configuration information (DCI) on the gateway. The DCI maps the application to the databases, indicating which is primary and which is secondary, and specifies their host systems.

Making the Primary Connection

Having interrogated the DCI to determine the name of the current primary for the application, **hbrouter** builds a DDA connection to the live host and logs on using the user-id and password supplied to **hbrouter.** Finally, **hbrouter** replugs the terminal connection to the primary.

hbrouter prompts you for a password, unless supplied previously in **.profile (**This is not recommended as it compromises security). Two logon attempts are allowed. The user-id and password are passed by the **hbrouter** process to the host to execute a login. The host login is configured to automatically logon to the primary database and start up the application.

With the primary connection established and the login successful, the **hbrouter** process transfers incoming and outgoing data between the user terminal and primary database in a transparent manner, so that the user appears to have a direct connection to the host. The primary connection remains active throughout the life of the user logon and is only disconnected when the user logs off.

Making the Secondary Connection

Having established the primary connection, the **hbrouter** process attempts to set up a connection to the standby host containing the secondary database using the same connection mechanism as for the primary connection. It does this as a background activity, without affecting the primary user. It first interrogates the DCI for the status of the standby host and resident database and if they are available it establishes the standby connection.

If the standby system is down or the database is unavailable, **hbrouter** does not attempt to build the connection, but periodically interrogates the DCI until the host/database status changes, at which point it establishes the standby connection.

The standby connection uses the same user-id and password as the primary connection. The secondary host login is configured to start the application automatically from **.profile**. All output, such as, logon banners and message of the day are discarded and are not seen by the user. All CPU-expensive initialisation tasks are carried out. The process is then stalled before the application is started.

The standby connection remains stalled while the user is logged on to the primary and is only disconnected when the user logs off. If the primary fails, the stall is unblocked and the logon is completed as part of the reconfiguration sequence on the surviving host.

This 'stalling capability' can be inserted into the RealityX start-up procedure using the HB-STALL verb, or it can be performed in UNIX using the **hbstall** command. Refer to the topic *Stalling a RealityX Logon* later in this chapter.

Fault Arbitration on Master Gateway

One Heartbeat gateway called the ‘Master gateway’ is chosen to monitor the Heartbeat ‘Alive’/’Fault’ messages from the FailSafe hosts as well as route Heartbeat user connections. Other gateways, designated 'slaves' only route Heartbeat user connections. The monitoring function on the master gateway is carried out by the **hbarbiter** process.

While the two FailSafe hosts remain healthy, the **hbarbiter** process monitors the Heartbeat ‘Alive’ messages, but does not affect either host. However, if **hbarbiter** receives a ‘Fault’ message from either host, or if it detects that the ‘Alive’ pulses have been interrupted, it responds by returning a ‘Fail’ message to both hosts which contains the system name of the failed host and the reason for the failure, as given in the ‘Fault’ message.

Automatic Shutdown of Failed Host

If the failed host is still functioning, the resident **hbprocess** identifies the system name in the 'Fail' message to be the local system and runs a **shutdown** script to shut itself down. If the failure prevents communication with the master gateway, then Heartbeat on the failed host performs a shutdown automatically after a pre-defined time-out period. More than one of the Heartbeat processes on the host can initiate a shutdown. This guards against Heartbeat process failure.

Automatic Reconfiguration of Surviving Host

On receiving a 'Fail' message with a remote system name (that is, with a name that does not match the local host name), the surviving host's **hbprocess** process executes a system reconfiguration script which re-configures the surviving host to be the live one. Typically it reconfigures all resident secondary databases as primaries and runs reconfiguration scripts for each database to reconfigure site specific and application/database specific resources. Primary databases on the surviving host are unaffected. The database reconfiguration scripts may be customised to meet site and application-specific requirements.

Heartbeat on the surviving host also updates its database configuration information (DCI), changing the status of the faulty host and its resident databases to 'Failed'. The changed DCI data is then propagated to all gateways and hosts and the **hbrouters** are informed of the change.

Automatic Switch-over of User Connections

On being informed of a status change in the DCI, each **hbrouter** looks at the **dci.file** to see if the primary database to which it is connected has failed. If it has, then the **hbrouter** sends the message 'Primary system has failed' to the user terminal and closes down the primary connection. It then waits for Heartbeat to reconfigure the secondary database as a primary by executing the **remote** script on the standby host.

When reconfiguration is complete, a message is sent to the gateway from the surviving host and the DCI data is again updated and propagated to all machines in the Heartbeat system. Again the **hbrouters** are informed that the DCI has changed and each **hbrouter**, after determining the nature of the change, activates its standby connection and sends a 'Switch-over Complete' message to the user. This indicates to the user that the standby connection is now active. It then starts transferring incoming and outgoing data between the user terminal and the standby host. Once Heartbeat has reconfigured the primary database, it releases all stalled logons for that database.

Depending on the application being run, some loss of synchronisation will occur in that the user will be presented with the first screen of the application and not that being used when the failure occurred.

Transaction Handling flags must be used to ensure that the database is in a consistent state after the automatic switch-overs, so that any partially completed transactions are rolled back.

Automatic switch-over in now completed and operation on the standby host is maintained while the failed host and database(s) are recovered. The host is re-booted, the failed database is recovered and resynchronised, and Heartbeat is restarted. The standby connection is then re-established automatically and normal Heartbeat operation re-instated.

Client/Server Connections and Remote Q-Pointers

Incoming client server connections and remote Q-pointers can be routed through a Heartbeat gateway with the same transparent routing capability as user connections. However, unlike user connections, they do not execute an **hbrouter** process. Instead their incoming connection causes the UNIX-Connect software to build an outgoing connection to the appropriate host and 'replug' the two circuits.

UNIX-Connect interrogates the DCI to determine the identity of the host containing the primary database. It then replugs the primary connection to enable bi-directional data transfer. It does not, however build a standby connection to the secondary.

In the event of a failure, the client/server connection or Q-pointer disconnects, hangs or times out. It is then the responsibility of the client or remote system to reconnect to the gateway which causes the UNIX-Connect software to re-build the connection, this time to the standby host containing the new primary. Client/server applications must be capable of recovering from this type of failure.

Outgoing client server connections are built from the host directly onto the LAN. They are not directed via a Heartbeat gateway. In the case of a primary host failure, they must be re-built by the surviving machine after switch-over. This is usually achieved by the **hbuser** script which is called by the re-configuration script on the surviving host as part of the switch-over sequence.

Printer Connections

Printer connections must also be re-built from the surviving machine after the automatic switch-over of users is completed. This may be performed by the Heartbeat re-configuration script, or the DBMS application. Directly connected printers cannot be switched.

The Heartbeat PROC (HBPROC) in the SYSMAN account is run at switch-over on the surviving databases to carry out any initialisation tasks necessary, such as, re-queuing spooler jobs, restarting despoolers etc.

Loadsharing by Gateways

Loadsharing software supported by Heartbeat, enables user connections to be dynamically shared between gateways so as to avoid overloading by optimising throughput. The operation and administration of this facility is described in Chapter 4.

Starting Up and Shutting Down Heartbeat

Heartbeat operation is started and stopped by using the **hbmenu** utility at the UNIX shell on each of the hosts and on each Heartbeat gateway. See Chapter 3 for a description of he **hbmenu** utility.

Entering **hbmenu** on a host system and selecting the Start heartbeat option starts up, the controlling process **hbprocess**.

Entering **hbmenu** on a Heartbeat gateway and selecting the Start heartbeat option starts up the controlling process **hbarbiter**.

To perform a clean shutdown of Heartbeat, **hbmenu** must be executed on each host and the Stop heartbeat option selected.

Normally Heartbeat is left running on the gateways and need only be stopped for gateway maintenance.

Refer to Chapter 3 for a detailed description of **hbmenu**.

Heartbeat on a Series X Host

Heartbeat software on the host monitors the local system and resident DBMS for fatal errors and, in the event of a failure, shuts down the failed system and reconfigures any secondary databases on the standby host as primaries.

Host Software

Heartbeat software on the host resides in the Heartbeat directory defined by the environment variable **HBROOT** and in the Heartbeat directory **/etc/hb**. Typically the directory structure is as shown in Figure 2-2.

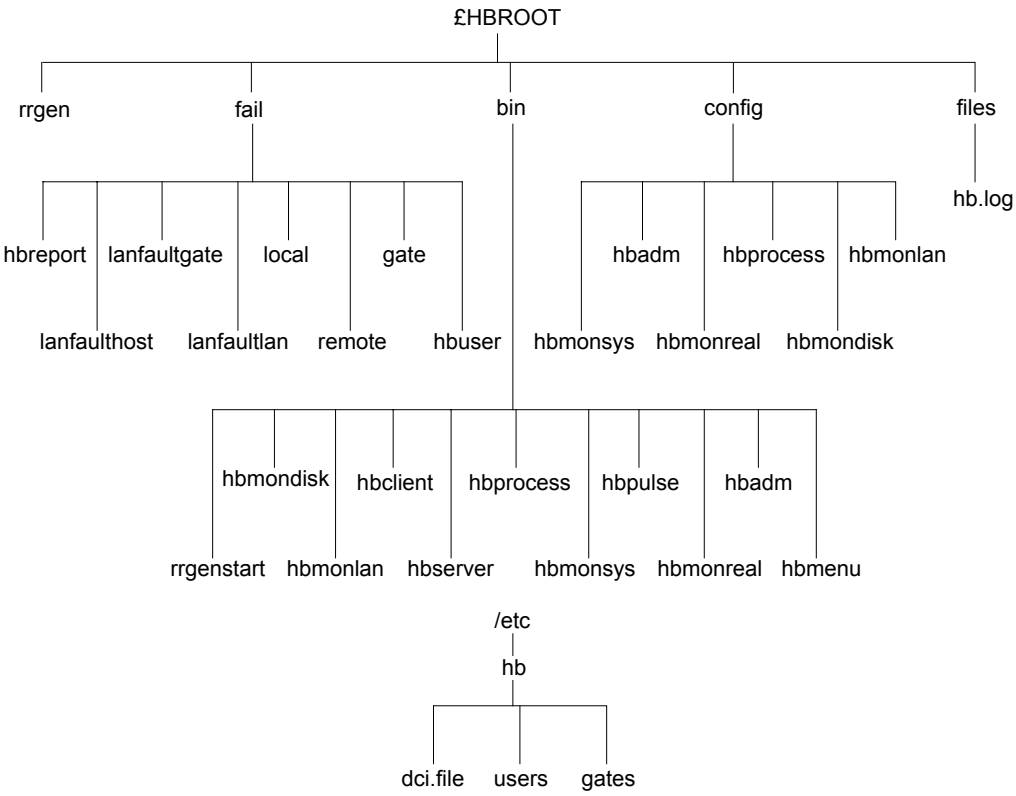


Figure 2-2. Heartbeat Directory Structure on Host

Heartbeat software on a host comprises:

- Executables, in the **bin** directory which perform the Heartbeat monitoring and administration functions.
- Scripts in the **fail** directory which run in the event of a system, application or LAN failure.
- Configuration files in the **config** directory which configure the executable processes of the same name.
- A log file in the **files** directory which is used to log significant Heartbeat events on the host.
- Loadsharing software in the **rrgen** directory, described to Chapter 4.
- Status files in the **/etc/hb** directory used to monitor and display the status of the Heartbeat system.

These software grouping are described below.

Executables

Executables in **£HBROOT/bin** on a host comprise:

hbadm The basic Heartbeat administration tool used to start-up, shutdown and configure Heartbeat. See Appendix B. This tool is provided for MDIS Support and advanced users only. The **hbmenu** utility calls **hbadm**.

hbmenu A menu-driven administration utility which provides a more user-friendly interface to start up, shutdown and configure Heartbeat. It is recommended that all users use this utility for day-to-day administration of Heartbeat. See Chapter 3 for a description of **hbmenu**.

hbprocess The controlling process for the Heartbeat monitoring system, started up by **hbadm**. It runs as a daemon and spawns a number of child processes **hbmon***, **hbpulse, hbclient**, **hbserver** and **rrgen.start.** It reads the **config** directory and starts an **hbmon** process for each file with the prefix **'hbmon'.**

Note: The asterisk '*' is a wild character indicating that a different name is appended to **hbmon** to identify the different monitor processes for the system software and each application.

hbpulse Sets up the communications link to the master gateway and sends the Heartbeat message pulses to **hbarbiter**. Refer to the topic *Heartbeat Gateways* in this chapter.

hbmon A generic monitoring process that monitors operating system or DBMS application errors. Several **hbmon** processes, each with a name beginning with '**hbmon**', may be spawned by **hbprocess** to monitor different error streams. For example, **hbmonsys** monitors system status, **hbmonreal** monitors RealityX status and **hbmonlan** monitors the local area networks which connect a Heartbeat system. All fatal errors are reported to the parent process **hbprocess**.

hbclient Sends the database configuration information (DCI) to the other host and Heartbeat gateways.

hbserver Receives the DCI from the other host.

Fail Scripts

UNIX scripts in **£HBROOT/fail** on a host are run in the event of a failure condition, as follows:

local Shuts down the local system if a failure is detected on it.

remote Reconfigures the surviving standby host as the live one. This includes reconfiguring secondary databases on the surviving system as stand-alone primaries.

hbuser Executed by the **remote** script for each new standalone primary database to re-configure site-specific and database resources. **hbuser** can be tailored to meet the requirements of each site. See Appendix I for a sample script.

Note: Using **hbuser** means that the **remote** and **local** scripts should not need to be modified.

gate Executes commands to be run when a gateway fails. The installed version generates a message to inform users that a particular gateway has failed.

lanfaultlan Executed if a LAN fault is detected by **hbmonlan**. It generates LAN fault messages and marks the status of the LAN in the DCI as 'Failed'.

lanfaulthost Executed if a host fault is detected by **hbmonlan**. It generates host fault messages and marks the status of the host in the DCI as 'Failed'

lanfaultgate Executed if a gate fault is detected by **hbmonlan**. It generates gateway fault messages and marks the status of the gate in the DCI as 'Failed'.

hbreport Called by remote **lanfaultgate**, **lanfaulthost** and **lanfaultlan** to generate fault reports. This script can be customised to report problems in a user-defined way.

Configuration Files

Configuration files in **£HBROOT/config** on a host comprise:

hbprocess Defines the rate at which Heartbeat pulses are processed by **hbprocess**, alternate locations for status files, the Heartbeat message queue key, the level of messages written to **hb.log** and whether monitor mode is set. See Appendix C.

hbmon Files commencing with this name, for example, **hbmonsys**, **hbmonreal,** and **hbmondisk**, contain environment variables defining which messages and fault levels are to be treated as fatal errors by the associated monitor process. See Appendixes D, E and H. **hbmonlan** is not used, but must be present in order for the **hbmonlan** process to start.

hbadm Sets the limit at which **hb.log** is archived during an **hbadm start** and sets root check suppression. See Appendix G.

Heartbeat Log

The **hb.log** file in **£HBROOT/files** records system, application and LAN fault messages on the local system. The level of messages logged in **hb.log** is set in the **hbprocess** config file.

hb.log is automatically copied to **hb.log.old** when it reaches a pre-defined size. The size is defined by the **HBLOGLIMIT** variable in **config/hbadm**. This must be set to a suitable value so that the partition containing **hb.log** is not filled up.

Status Files

Status files in **/etc/hb** on a host comprise:

dci.file Contains the database configuration information (DCI) for the Heartbeat system. The DCI is entered using **hbmenu**. See Chapter 3.

gates Contains status information about the Heartbeat gateways, including names, addresses, participating/non-participating status and loading. This is used by the Loadsharing process **£HBROOT/rrgen/rrgen**. See Chapter 4 for details.

users Contains status information for each Heartbeat user connection. This includes gateway connected to, gateway user-id, host user-id, application connected to, and secondary connection status. This can be listed using the **hbmenu** 'Display users file' option (see Chapter 3), or by generating an ENGLISH listing of the special file view HB.LOG.

Logging into RealityX on the Hosts

When a user logs on to a Heartbeat gateway, the **hbrouter** process builds a DDA connection to the live host and a backup DDA connection to the standby host. After connecting to the live host, the logon to the primary database is completed and any user application is started. The secondary logon to the standby host is also started, but is stalled before being completed. The setting up of the standby connection happens in the background and does not affect user operation. **hbrouter** then maintains the primary connection as active and the standby connection is stalled as long as the user is logged on, or until a failure occurs.

The recommended login configuration runs the **reality** programs on the live and standby hosts when the user logs in. The stall on the secondary connection is implemented within the RealityX environment by a TCL verb HB-STALL, inserted in the logon PROC. This means that in the event of a switch-over, the logon to the newly configured primary is expedited without further delay incurred by running **reality**.

Stalling the RealityX Logon

To set up the stall on the secondary connection, the HB-STALL command is inserted in the LOGON PROC, or in the application called from the PROC. It interrogates the DCI to determine whether it is being executed on a primary or secondary database.

On a primary database, HB-STALL allows RealityX to complete the logon successfully and data is then replugged to and from the user.

On a secondary database, RealityX is stalled and remains stalled until the primary fails and the secondary is reconfigured.

An alternative method of stalling the secondary connection is available by running the **hbreality** script when logging on to the host. This stalls the connection in the UNIX environment. If its the primary connection, then **hbreality** executes the **reality** program to start up the DBMS application and logon to the primary database immediately. If its the secondary connection, then the **hbreality** script stalls. The secondary connection remains stalled, until it is unblocked when the standby host is reconfigured during a Heartbeat switch-over.

Unblocking the Stall

The stall on the secondary connection is unblocked during a Heartbeat switch-over by the **hbstall -u** command run in the system reconfiguration script **remote** on the surviving host.

Host Monitoring

The monitor processes **hbmonsys** and **hbmonreal,** monitor the status of error messages sent to a message log. **hbmonsys** monitors operating system errors in the system log **/dev/log** and **hbmonreal** monitors RealityX errors in the log file **REALROOT/files/daemon.log**.

Fault levels which constitute a fatal error are defined in the associated **hbmonsys** and **hbmonreal** configuration files.

Each **hbmon** process also sends a periodic 'Alive' message pulse to the parent **hbprocess** to indicate that it is operating correctly. If an **hbmon** process misses sending more than a configurable number of 'Alive' pulses to **hbprocess**, the **hbmon** process is considered to be dead. This is a fatal error. The number of 'Alive' pulses that can be missed by **hbprocess** before it is considered a fatal error is defined in the **HB_MISSED_BEATS** environment variable in the **hbprocess** config file. See Appendix C.

An **hbmon** process contains all the device specific information required to read and interpret a particular error log and write information to log files. Each **hbmon** process checks these machine/application-specific error reports against fatal error conditions defined in its configuration file and, if a fatal error is detected, it sends a 'Fault' message to **hbprocess** containing the reason for the error. It then continues monitoring its error stream until requested otherwise by **hbprocess**.

On receiving a 'Fault' message from an **hbmon** process, **hbprocess** passes it on to **hbpulse.** Refer to the topic *Fault Message Handling* later.

hbmondisk periodically reads a disk block from each disk listed in its **config** file. While read operations are successful, **hbmondisk** sends periodic 'Alive' messages to **hbprocess**. However, if a serious disk error is detected, such as a SCSI bus hang, or similar, **hbmondisk** stops sending 'Alive' messages to **hbprocess**. **hbprocess** then deduces that the system has failed and sends a fault message to **hbpulse**.

LAN Monitoring

Two **hbmonlan** processes, one on each host, run simultaneously to monitor the status of all LAN connections in the Heartbeat system, including the Transaction Logging LAN (TL), the two host LANs (H1 and H2) and the user LAN (UL). See Figure 2-3.

hbmonlan is spawned by **hbprocess** along with the other **hbmon** processes and communicates with **hbprocess** by means of periodic 'Alive' pulses to indicate that it is operating correctly.

At startup **hbmonlan** obtains a list of Heartbeat system connections from the **/etc/hb/routes** file. See Appendix A.

Having determined the connections to be monitored, **hbmonlan** periodically probes each link in the Heartbeat system by executing a remote shell (**rush**) command to try to make the connection. If the attempt is successful, then the session is closed and the connection is marked as O.K. in an internal table. If the attempt fails or times out, then the connection is marked as 'Failed' in the internal table. See *UNIX-Connect User Guide* for details on the **rush** command.

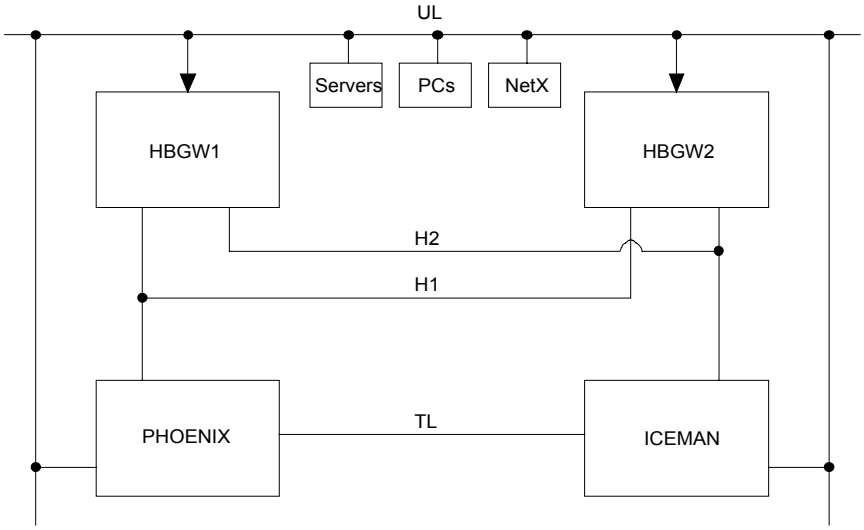


Figure 2-3. Heartbeat LAN Connections Monitoring

When all Heartbeat LAN connections have been probed, **hbmonlan** takes the table of probe results and transmits it via the user LAN to the **hbmonlan** process running on the other host. If the user LAN connection is faulty, then the TL LAN is used as an alternative route.

Once the probe results have been exchanged, one of the **hbmonlan** processes compares the tables and constructs a cross-reference table for each of the four LANs (TL, H1, H2 and UL). The matrix contains an entry for each possible connection via a LAN and is filled with 0's and 1's (0 - success, 1 - failed) according to the result of the latest probe.

For example, for host LAN H1, suppose that host PHOENIX and gateway HBGW1 both report failed attempts to connect to gateway HBGW2, then the matrix for LAN H1 would be as shown in Table 2-1.

Table 2-1. Example of LAN H1 Monitor Probe Results

	phoenix	hbgw1	hbgw2
phoenix	0	1	0
hbgw1	0	0	0
hbgw2	0	1	0

Summing the 1's in the matrix enables the fault to be diagnosed. The highest total provides a pointer to the faulty component (LAN connector on gateway hbgw1, in the above example). All four LAN matrices are analysed and the results compared. If no single system is diagnosed faulty, then a generic LAN fault is reported.

Once **hbmonlan** has diagnosed the fault, it runs an appropriate **fail** script which reports the fault and initiates recovery. One of three recovery scripts, **lanfaulthost**, **lanfaultgate** or **lanfaultlan**, is run, corresponding to the three types of possible fault. See the topic *Fail Scripts* earlier in this section.

The **fail** script generates the appropriate fault messages for display at the system console and marks the DCI entry for the faulty system or LAN as 'Failed'.

Heartbeat Pulse Generation

The **hbpulse** process sets up the communications link to the master gateway and sends a continuous stream of 'Alive' message pulses to **hbarbiter**. **hbpulse** determines which is the master gateway by interrogating the gate DCI entries.

Once the Heartbeat arbiter connection is established, **hbpulse** starts sending Heartbeat 'Alive' message pulses to the **hbarbiter** process to indicate that the FailSafe host software is alive and working correctly. This Heartbeat pulse rate is set by the HB_MINOR_TICK variable in the **hbprocess** configuration file. The default is one pulse every 200 mS.

At longer intervals, **hbpulse** generates an 'Alive' message pulse with an ACK flag added. This major tick 'Alive' message is sent to **hbprocess** and the **hbarbiter** on the master gateway. The pulse rate of the major tick is set by the HB_MAJOR_TICK variable in the **hbprocess** configuration file. The default is one pulse every 40 mS.

Because of the ACK flag added to the major tick 'Alive' pulse, **hbpulse** expects an ACK message from **hbprocess** and **hbarbiter**. If an ACK is not returned before the next major tick, then **hbpulse** constructs a 'Fault' message, with the name of the faulty system and the reason for the fault.

Fault Message Handling

hbpulse takes the 'Fault' message received from an **hbmon** process and passes it in a Heartbeat pulse to the **hbarbite**r process on the master gateway. The 'Fault' message contains the failed system name and the reason for the failure.

Also, if **hbprocess** fails to return an ACK within the major tick period, this is considered as a fatal error as it is most likely that the system is in difficulty. In this case, **hbpulse** constructs a 'Fault' message containing the name of the failed system and the reason for the failure, and sends it to **hbarbiter** on the master gateway. It then runs the **local** script to shut down the system.

If **hbarbiter** fails to return an ACK, **hbpulse** sends a STOP message to shut down the Heartbeat link and then attempts to re-establish it again.

Refer to the sub-topic *Heartbeat Pulse Monitoring* for a description of how **hbarbiter** on the master gateway handles 'Fault' messages.

Fail Message Handling

After processing the 'Fault' message, **hbarbiter** returns a 'Fail' message to **hbprocess** on each host. **hbprocess** then extracts the name of the failed host from the 'Fail' message and runs the UNIX scripts **local** or **remote** to shutdown the failed host and reconfigure the surviving host as the live system. Refer to the topic *Fail Scripts* earlier in this chapter.

Automatic Shutdown of Failed Host

The **local** script is run when the 'local' system has failed, to shutdown the failed host. It sends messages to the host's system console indicating that the local system has failed and is about to be shutdown. It then executes **shutdown** to shutdown the failed system, when possible.

Automatic Reconfiguration of Surviving Host

The **remote** script is run, when the 'remote' system has failed, to reconfigure the surviving host.

Also, on receiving the 'Fail' message, Heartbeat updates the database configuration information (DCI) held on the surviving host, changing the status of the other host and its resident databases to 'Failed'. The changed DCI data is then propagated to all gateways.

Remote Script Execution

The **remote** script first gets a list of all applications residing on the failed machine from the DCI. It checks which has a secondary database on the surviving host, then for each secondary, it calls **hbreport** with the message:

```
Heartbeat: Remote system failed - Reconfiguring Application-name
```

remote then reconfigures each secondary as a standalone primary. Then, for each newly configured primary, it runs the **hbuser** reconfiguration script.

hbuser Script Execution

hbuser is tailored to reconfigure site-specific resources in the UNIX environment e.g. reconfiguring XUI connections. Reconfiguration procedures required on a particular

RealityX database (e.g. resetting despoolers) are performed by the Heartbeat PROC (HBPROC) which can be run from **hbuser**. Refer to Appendix I for a sample **hbuser** script.

Note: It should not be necessary to modify the **remote** script. Procedures specific to a site and a particular database should be carried out from the **hbuser** script and HBPROC.

Once **hbuser** has finished running, the **remote** script reconfigures the Heartbeat DCI, setting the newly configured database as 'Primary', and unblocks the stall on user logons, enabling normal user activity on the new primaries.

Note that **hbuser** is run for each database before users are installed. It is therefore important that the number of foreground processes run by **hbuser** to carry out reconfiguration procedures are kept to a minimum as this will lengthen the switch-over time. Reconfiguration processes can be run in the background by appending an ampersand (&) to the command line in the **hbuser** script.

Heartbeat Gateway Software

Heartbeat gateway software manages user connections to the Series X FailSafe hosts and monitors the health of the hosts. In the event of a primary failure, it initiates the re-configuration of the FailSafe secondaries as primaries on the standby host and re-routes users to the standby host.

Heartbeat software on the gateway resides in the Heartbeat directory defined by the environment variable **HBROOT** and in the Heartbeat directory **/etc/hb**. Typically the directory structure is as shown in Figure 2-4.

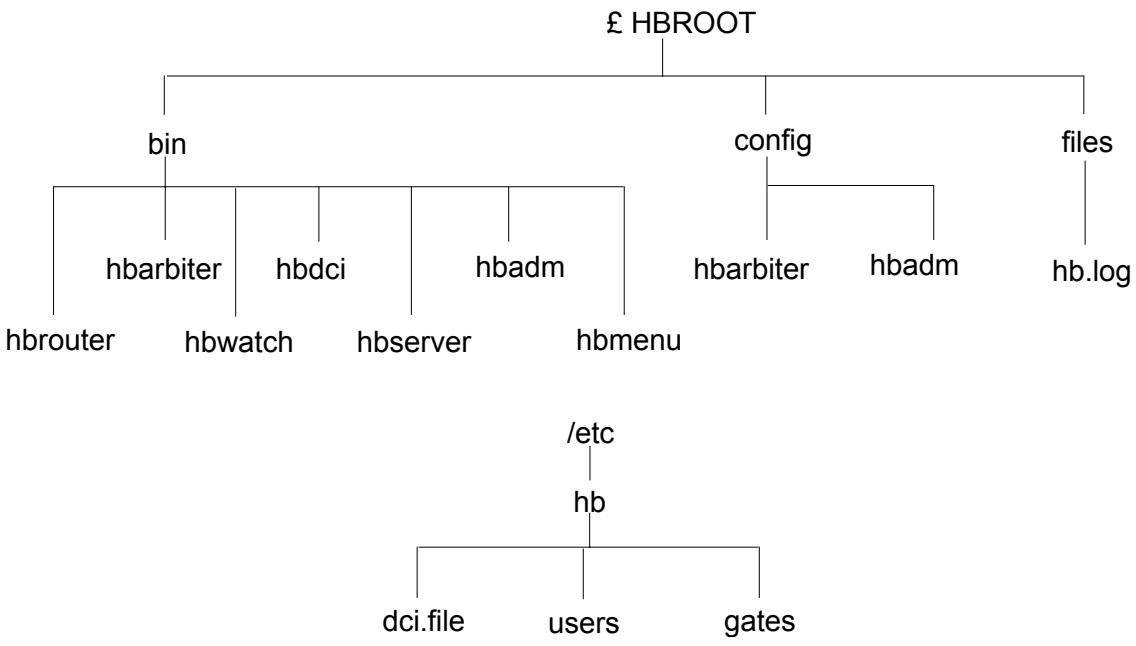


Figure 2-4. Heartbeat Directory Structure on Gateway

The Heartbeat Gateway software comprises:

- Executables in the **bin** directory which perform Heartbeat monitoring, routing and administration functions on a gateway.
- Configuration files in the **config** directory which configure the executable processes of the same name.
- A log file in the **files** directory which is used to log significant Heartbeat events on the gateway.
- Loadsharing software in the **rrgen** directory, described in Chapter 4.
- Status files in the **/etc/hb** directory used to monitor and display the status of the Heartbeat system.

Executables

Executables in **£HBROOT/bin** on a gateway comprise:

hbadm The basic Heartbeat administration tool used to start up, shutdown and configure Heartbeat. See Appendix B. This tool is provided for MDIS Support and advanced users only. The **hbmenu** utility which executes **hbadm** is provided for normal administration.

hbmenu A menu-driven administration utility which provides a more user-friendly interface to start-up, shutdown and configure Heartbeat. It is recommended that all users use this utility for day-to-day administration. See Chapter 3.

hbarbiter This is the controlling process for the Heartbeat gateway software, started by **hbadm.** It monitors Heartbeat messages from the hosts and, on receiving a 'Fault' message, or detecting an interruption in 'Alive' pulses, initiates a switch-over by sending a 'Fail' message to the hosts.

This arbitration function is only carried out on the master gateway. When **hbarbite**r is started up, it interrogates the DCI gate entries in **/etc/dci.file** to determine if it is on a master gateway. If it is, **hbarbiter** spawns an **hbwatch** for each host defined in the **dci.file**. It then monitors the message queue for 'Alive'/'Fault' messages from the **hbwatch** processes.

hbwatch Monitors the 'Alive'/'Fault' messages from the **hbpulse** process on the hosts. One **hbwatch** process for each Heartbeat host is spawned. Each **hbwatch** is given the name of a Series X machine as a parameter e.g. **hbwatch host1** and **hbwatch host2**.

hbrouter Manages the primary and secondary user connections to the hosts, switching users to the secondary connection in the event of a primary failure.

An **hbrouter** process is executed by each user logging in to a Heartbeat gateway. This is enabled by entering an **hbrouter** statement in the user's **.profile**. Refer to the topics *Configuring Heartbeat Connections to a RealityX Database* and *Description of hbrouter Command* in Chapter 3.

hbserver Receives DCI messages from the hosts' **hbclient** and passes them to **hbdci.**

hbdci Manages the DCI on a gateway, updating the **dci.file** and sending a signal to the **hbrouter** processes if the DCI changes.

Configuration Files

Configuration files in **£HBROOT/config** on a gateway comprise:

hbarbiter This file configures the **hbarbiter** process, including parameters to define the number of minor ticks which when missed by **hbwatch** generates warning messages or initiates a failure recovery procedure. It also defines the Heartbeat message queue key, the level of messages written to **hb.log**; the number of hosts sending Heartbeat pulses the time-to-live for gate load; and the number of user connections which constitute 100% loading. See Appendix F.

hbadm This file defines the limit at which **hb.log** is archived during an **hbadm start** and sets root check suppression. See Appendix G**.**

Heartbeat Log

The gateway's **hb.log** file in **£HBROOT/files** logs system and application messages on the local system. The level of messages logged in **hb.log** is set in the **hbprocess** config file.

hb.log is automatically copied to **hb.log.old** when it reaches a pre-defined size. The size is defined by the **HBLOGLIMIT** variable in **config/admin.** This must be set to a suitable value so that the partition containing **hb.log** is not filled up.

Status Files

Status files in the **/etc/hb** directory on a gateway are the same as those on a host. Refer to the sub-topic *Status Files* earlier in this chapter.

Heartbeat Pulse Monitoring

Each Heartbeat link from a host to the master gateway is monitored by an **hbwatch** process. **hbwatch** extracts the messages in the Heartbeat pulse sequence transmitted by the host and

passes them on to the **hbarbiter.** If the Heartbeat 'Alive' message requires an ACK**, hbwatch** sends one, indicating to the host that **hbarbiter** is still functional. If the Heartbeat connection is lost, **hbwatch** constructs a 'Fault' message and sends it to **hbarbiter**.

On receiving a 'Fault' message from **hbwatch**, **hbarbiter** constructs a 'Fail' message, extracting the name of the failed host and the reason for the failure, and putting it in the 'Fail' message. This is then passed to all **hbwatch** processes to send to the hosts. The failed host name is used at the hosts to initiate shutdown of the failed host and reconfiguration of the standby host. Refer to the topic *Fail Message Handling* earlier in this chapter.

Automatic Switch-over of User Connections

On receiving the 'Fail' message from the master gateway, Heartbeat on the surviving host updates the **dci.file** changing the status of the other host and its resident databases to 'Failed'. The changed DCI data is then propagated to all gateways.

On receiving the updated DCI from the surviving host, the Heartbeat DCI process (**hbdci**) writes the updated DCI to the **/etc/dci.file** on the gateway. It also sends a signal to all **hbrouters** on the gateway to tell them that the **dci.file** has changed.

The 'dci.file changed' signal from the **hbdci** process causes each **hbrouter** to interrogate the **/etc/dci.file** to check if the host failure has affected its primary or secondary connections. If its primary connection status has changed to 'Failed', **hbrouter** stops replugging data to the failed machine, closing it down, if possible, and generates a message to inform the connected user that the primary host connection has failed. It then waits for the FailSafe pair to be re-configured, by changing the secondary to a standalone primary.

Once the re-configuration is completed a DCI message is sent to the gateways from the standby machine for each re-configured database, to mark it as 'Primary' in the DCI. The **hbdci** process then marks all databases in the **/etc/dci.file** as primaries**.** It also sends a signal to all **hbrouters** on the gateway to tell them that the **/etc/dci.file** has changed.

On receiving the 'dci.file changed' signal, each **hbrouter** interrogates the **dci.file** again to check if the change has affected its database connections. If its secondary connection status has changed to 'Primary', **hbrouter** replugs data to the new stand-alone machine and activates the stalled process on the standby host. **hbrouter** also sends a message informing the user that the standby connection is now active.

User Interface

Logging On

The user interface to the primary database via a Heartbeat gateway simulates logging on directly to a Series X host. Only a few extra login messages show that you are logging in to a gateway. For example, you enter your user-id at the login prompt:

```
login: 101455
```

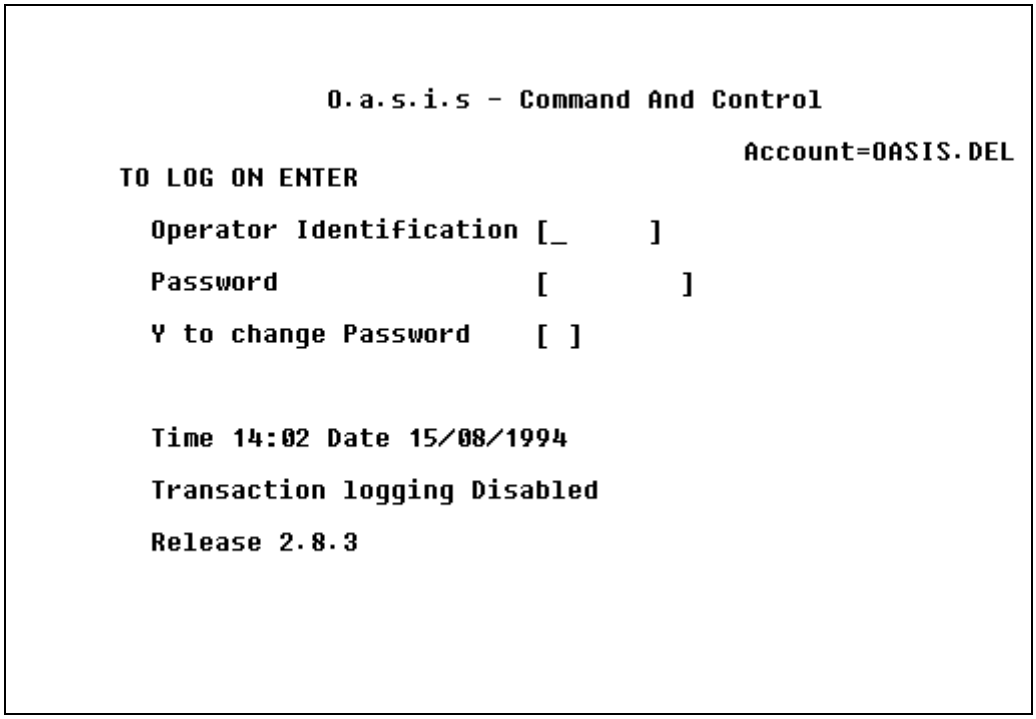
This then displays the login messages on the gateway followed by a UNIX password prompt:

```
UNIX SYSTEM V/88 Release 4
hbgw1
Copyright (C) 1984, 1986, 1987, 1988, 1989, 1990 AT&T
Copyright (C) 1991, 1992 UNIX System Laboratories, Inc.
Copyright (C) 1987, 1988 Microsoft Corp.
Copyright (C) 1991, 1992 Motorola, Inc.
All Rights Reserved
UNIX:login: INFO: Last login: Mon Jul 04 10:46:26 on pts45
              UNIX SYSTEM V/88 Release 4 Version 4.2
Password:
```

On entering the appropriate password you are logged on to the database. The login to the host is transparent to the user. The RealityX logon banner is then displayed, similar to the following:

```
*****************************************************************************
***                MDIS  -  McDonnell  Information  Systems                ***
***                         RealityX 4.0 Rev C ut4.                        ***
***                               O.A.S.I.S                                ***
***                                                                        ***
*----------------------------------------------------------------------------*
*                                                                            *
* Copyright McDonnell Information Systems Ltd 1994. All rights reserved.     *
* No part of this software may be reproduced, adapted or loaded onto any     *
* computer system without the prior written consent of the copyright owner.  *
* For more copyright information see the file /usr/realman/copyrights.       *
*                                                                            *
******************  Logged on at 10:52:21 on 04 Jul 1994   ******************
```

This may be followed by the logon sequence for the application being run. For example:



Automatic Switch-over

If the primary system or application fails, Heartbeat initiates an automatic switch over. Input is suspended for a short time and the following messages are displayed:

```
Primary System has failed

Switching to secondary

Please wait
```

When switch-over is completed, the following is displayed:

```
Switch-over completed
```

You are then returned to the top working environment of your application and normal Heartbeat operation continues.

Chapter 3
System Administration

This chapter details administrative procedures for setting up, maintaining and recovering normal Heartbeat operation. It includes procedures to:

- Configure Heartbeat connections to a RealityX database or C server program.
- Start up and shutdown Heartbeat.
- Configure a Heartbeat system.
- Recover normal Heartbeat operation after a Host/DBMS failure.
- Recover Heartbeat operation after a master gateway failure.
- Re-configure Heartbeat after a slave gateway failure.
- Toggle primary/secondary databases in a Heartbeat system.

A full description of **hbmenu** is given at the end of the chapter.

Introduction

Most of the procedures described in this chapter are carried out using the Heartbeat administration utility **hbmenu**. Two exceptions are the procedure to configure Heartbeat connections to a RealityX database and the procedure to configure Heartbeat connections to a C server which both involve the setting up of routing information and user-ids on each system.

hbmenu is a UNIX-based utility, invoked by entering the **hbmenu** command at the UNIX shell prompt, which runs a set of interactive menu-driven procedures for starting up, shutting down and configuring a Heartbeat system. **hbmenu** can only be run by super-user (root). A detailed description of **hbmenu** is provided towards the end of this chapter.

This chapter details the menus, prompts and status screens displayed by **hbmenu** when the following Heartbeat administration procedures are carried out:

- Starting up, stopping and configuring Heartbeat
- Recovery after a host or DBMS failure
- Recovery after a master gateway failure
- Re-configuring after a slave gateway failure
- Toggling primary/secondary databases

Note: Although, the overall structure of menus and status screens is now established, as described in this manual, **hbmenu** may still be subject to minor enhancements which may not always be reflected in this manual.

The configuration procedures run by **hbmenu** are used to enter the database configuration information (DCI) into **/etc/hb/dci.file** on the local system. The DCI is thereafter distributed to all other hosts and gateways in the Heartbeat configuration and entered into their local **dci.file**s.

The **hbmenu** utility invokes the UNIX-based utility **hbadm** to perform Heartbeat administrative operations. For a description of **hbadm**, see Appendix B.

Notes:

1. Heartbeat operation can be tuned by changing the environment defined in the configuration files **hbprocess, hbmondisk**, **hbmonsys**, **hbmonreal**, **hbadm** and

hbarbiter which reside in the **$HBROOT/config** sub-directory on all hosts and gateways. The default configuration is shown in Appendixes C to H. Refer to your MDIS Support representative for more details.

2. Re-configuration requirements for a particular system when switching over to standby must be defined in the **hbuser** script in the **£HBROOT/fail** directory. **hbuser** is called by the **remote** re-configuration script, also residing in **£HBROOT/fail**. Again, if you need to modify **hbuser,** see your MDIS Support representative. Refer to Appendix I for a typical example of an **hbuser** script.

Examples

The typical Heartbeat configuration shown below in Figure 3-1 is referenced in the examples given in this chapter. Host system names iceman and phoenix, gateway names hbgw1 and hbgw2, and database names oasis1 and oasis2 are given as examples.

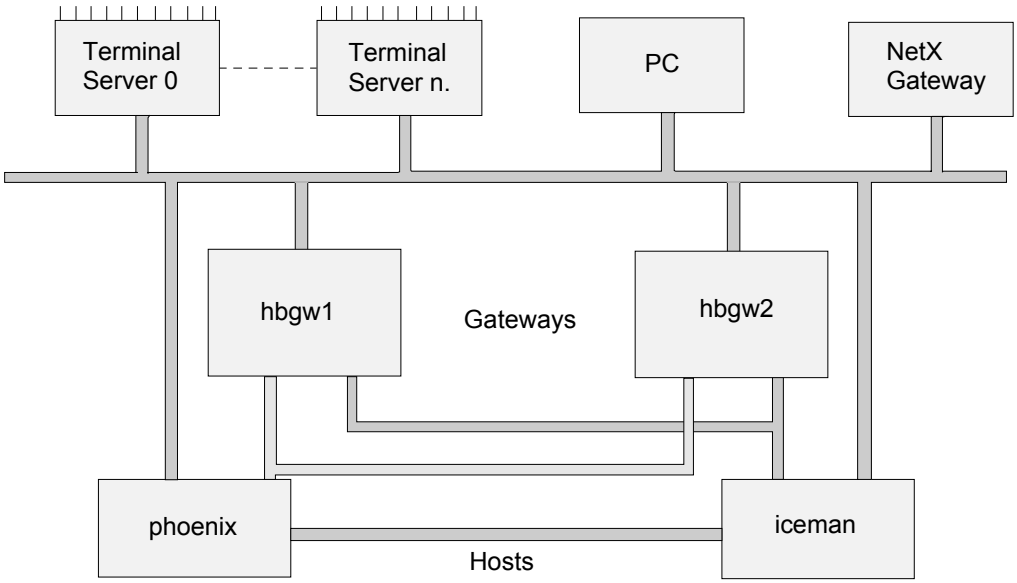


Figure 3-1. Typical Heartbeat Configuration

Configuring Heartbeat Connections to a RealityX Database

To configure Heartbeat connections to a RealityX database, you must:

1. Set up the routing information in **/etc/hb/routes** and **/etc/ROUTE-FILE** on all gateways and hosts to enable inter-communication between systems in the Heartbeat configuration.

2. Set up the routing information in **/etc/hb/routes** to enable connection to the primary and secondary databases for an application.

3. Set up the database configuration information in **/etc/hb/dci.file** to define the Heartbeat configuration.

4. Set up the Heartbeat user-ids on the gateways and hosts to enable users to logon to the primary database via a Heartbeat gateway.

5. Set up routing information on each user system e.g. terminal server or PC to enable users to connections to the Heartbeat system.

Setting Up System Routing

The routing information to be entered in /**etc/ROUTE-FILE** on both Heartbeat hosts and gateways in the Heartbeat system must include:

- Destination entries on all hosts and gateways, defining the private system names of the two hosts in the Heartbeat configuration, e.g. **phoenix** and **iceman**. See Appendix A.

- Listening entries on the hosts.

Refer to the *Unix-Connect System Administration Guide* for details.

The routing information to be entered in **/etc/hb/routes** is described in Appendix A.

Setting Up Application Routing

The following routing information must be set up in **/etc/ROUTE-FILE** on all Heartbeat gateways and hosts to enable connection to the primary and secondary databases on the hosts:

- Q-type entries on the gateways (e.g. **oasis1** and **oasis2)** to reference the primary and secondary host destination entries e.g. **phoenix** and **iceman**.

- Heartbeat entries on the gateways, identified by the Application name e.g. **oasis**. These are used to set up client/server and remote Q-pointer connections to the primary database.

- RealityX entries e.g. **oasis1** and **oasis**, to define the application databases on the hosts.

Setting Up the DCI

Database configuration information (DCI) must also be set up on all hosts and gateways, as described later under the topic *Procedures to Start Up, Stop and Configure a Heartbeat System*.

Setting Up Users

The following must be set up to enable a Heartbeat user to logon to an application database via a Heartbeat gateway.

1. Gateway UNIX user-id
2. **hbrouter** command in **.profile** on gateways
3. Host UNIX user-id
4. RealityX database user-id

The recommended procedure is:

1. Set up the same UNIX user-id on each Heartbeat gateway. It is recommended that you set up a user-id for each user. Gateway user-ids do not need a password.
2. Enter an **exec hbrouter**, specifying the application name, as the last line in the **.profile** for each UNIX user-id. For example:

   exec hbrouter -d oasis

   Note: It is recommended that all user-ids for the same application have the same home directory defined, so that when users logon they all run the same **.profile**, assuming that terminal type is the same.

   hbrouter interrogates the DCI to determine the identities of the primary and secondary hosts and builds a DDA connection to each.
3. Set up UNIX user-ids and allocate passwords on the hosts, to match those on the gateway. This is the password that **hbrouter** prompts for and passes on to the host to execute the login.
4. Set up RealityX user-ids on the database to match the gateway and host UNIX user-ids exactly (including the case of the alpha characters). With or without a RealityX password, the UNIX user will have trusted access to RealityX.

Setting Up User LAN Routing

The following routing information must be entered in **/etc/ROUTE-FILE** on a terminal server or user's PC to enable connection to a Heartbeat system:

- A destination entry for each Gateway
- An alternative host entry that list these destination entries
- A Q-type entry that specifies the name of the Heartbeat entry on each Gateway and references the Alternative Host entry as the destination system

Refer to *UNIX-Connect System Administration Guide* for details.

Example of Operation

The effect of using this configuration is illustrated by the following example.

If a user logs in to a gateway as **hboasis**, no password is requested and the **.profile** of user **hboasis** is executed. This runs the **hbrouter -d oasis** command.

hbrouter interrogates the DCI to ascertain the names of the primary and secondary databases associated with **oasis** to connect to. So for example, the DCI may show that a primary database **oasis1** and a secondary database **oasis2** are the databases for the **oasis** application.

hbrouter prompts for a password, not previously provided in the **hbrouter** command line, then builds DDA connections to both live and standby hosts, **phoenix** and **iceman.**

The user only sees two terminal outputs:

- Login prompt from the gateway
- Password prompt from **hbrouter**

The user login and password prompts look the same as if you were logging on directly to a Series X host, except for a few extra messages displayed by the gateway.

Once the DDA connection to the live host is established, the user-id and password supplied previously to **hbrouter,** are passed on to the host to execute the login without any further user interaction. No password check is made.

Any user application set up for the database is then run.

Refer to the topic *User Interface* in Chapter 2 for an example of a login session.

Configuring Heartbeat Connections to a C Server Program

The procedure to configure a Heartbeat connection to a C server program is as follows:

1. Set up the routing information in **/etc/hb/routes** and **/etc/ROUTE-FILE** on all gateways and hosts to enable inter-communication between systems in the Heartbeat configuration. These must include:

   - Destination entries on all Heartbeat gateways defining the private system names of the two hosts connected to the gateways via the Heartbeat LANs, e.g. **phoenix** and **iceman**. See Appendix A.

   - Listening entries on the hosts

   Refer to the *Unix-Connect System Administration Guide f*or details.

   Refer to Appendix A for a discussion of routing information to be entered in **/etc/hb/routes**.

2. Set up Heartbeat entries in **/etc/ROUTE-FILE** on all gateways. The system name of a Heartbeat entry is the application name. When a client server link is attempted, the application name is passed to the Heartbeat software which then determines the live host name.

3. Set up the DCI required to locate the C server. The procedure is the same as that described under the topic *Configuring Applications* later in this chapter. The only difference from configuring the application for a RealityX database is that you must enter the host name again at the 'Database name' prompt instead of a database name.

4. Ensure that the routing information on the user's PCs and terminal servers is set up correctly. Refer to the previous sub-topic, *Setting Up User LAN Routing*.

Procedures to Start Up, Stop and Configure a Heartbeat System

The following procedures to startup, stop and configure Heartbeat using **hbmenu** are described with reference to a typical Heartbeat configuration, illustrated in Figure 3-1.

Running hbmenu

The **hbmenu** command must be entered at the system console for each FailSafe host and Heartbeat gateway. In our example, two hosts, **phoenix** and **iceman**, and two gateways, **hbgw1** and **hbgw2,** are considered.

On entering **hbmenu**, a 'top menu' screen similar to that illustrated below will be displayed on a terminal connected to **iceman**. Slightly different status information will appear on the terminals connected to the gateways. For details, refer to the description of the status information under the topic *Description of hbmenu* later in this chapter.

Starting Up Heartbeat

```
hbmenu            Heartbeat status on iceman            Wed Feb 28 10:51:56

Host............ Status...  Arbiter status
Gate............ Status...  Type.... Load.. Load status

Application..... Dbase........... .Host........... Status....


Lans  -   h1.. Up      h2.. Up       t1.. Up      u1.. Up

s.  Start heartbeat
r.  Redisplay main menu
q.  Quit hbmenu

Option [r]:
```

Start up Heartbeat on each host and gateway console by entering '**s**' at the 'Option' prompt in the top menu screen, first on the gateways and then on each host.

Note: To avoid a large number of unnecessary error messages on the system console, start up Heartbeat on the gateways first.

The status information changes to show that Heartbeat on that system is now active:

```
hbmenu            Heartbeat status on iceman            Wed Feb 28 10:52:20

Host............ Status...  Arbiter status
iceman           Active     Never Connected

Gate............ Status...  Type.... Load.. Load status......

Application..... Dbase.......... .Host.......... Status....


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

NO MASTER RETURN for menu
```

Notes:

1. The previous screen illustrates the status information that appears on **iceman**'s system console. A DCI entry for the local host (**iceman**) is created when Heartbeat is started on that host. No status information is displayed on the gateways as gateway entries are not created when **hbmenu** is run on the gateways, and client-server links to the hosts have not yet been built.

2. The LAN status is reported automatically by the **hblanmon** process on the hosts.

Press RETURN to display the top 'Stop heartbeat' menu. Now select a system console to use. For the purpose of this description we will use **iceman**'s console.

Configuring Hosts

1. At **hbmenu's** top menu, enter '**h**' to select the Host menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: h
```

2. At the Host menu, enter '**a**' to add a host entry to the DCI:

```
Host............ Status...
iceman           Active

a.  Add host
b.  Delete host
c.  Change host status
r.  Return to upper menu

Option [r]: a
```

3. At the next prompt, enter the name of the remote host to create a DCI host entry for it:

```
Host name  [iceman]: phoenix
```

This writes a **phoenix** entry into the DCI and re-displays the Host menu with the revised DCI displayed, as shown below in step 4:

4. At the Host menu, press RETURN to go back to the top menu:

```
Host............ Status...
iceman           Active
phoenix          Never

a.  Add host
b.  Delete host
c.  Change host status
r.  Return to upper menu

Option [r]:
```

Notes:

1. The gateways pick up the local host information automatically.

2. When you add the remote host DCI entry (**phoenix**) on **iceman**, **iceman** connects to the remote host (**phoenix**) and passes its DCI details. **phoenix** then reciprocates by connecting to **iceman**. A two way client-server link is therefore set up. Each host then has two DCI entries defining their respective local and remote systems.

5. At the top menu, press RETURN again to display the current status of the system:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

The status information displayed on **iceman**'s console will now have changed to the following:

```
hbmenu            Heartbeat status on iceman            Wed Feb 28 10:52:53

Host............ Status...  Arbiter status
iceman           Active     Never Connected
phoenix          Active     Never Connected

Gate............ Status...  Type.... Load.. Load status......

Application..... Dbase..... .Host..... Status....

Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

NO MASTER RETURN for menu
```

Note: Status information on the gateways will differ slightly from that seen on the hosts. Differences between host and gateway status displays are described under the topic *Description of hbmenu* later in this chapter.

Configuring Gateways

Before you starting this procedure, ensure that the status of each host is 'Active', then proceed as described below. If the status does not change after one minute, check the Heartbeat and UNIX-Connect log files.

1. At the top menu, enter '**g**' to select the Gate menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]:  g
```

2. At the Gate menu, enter '**a**' to add a gateway entry to the DCI:

```
Gate............ Status...  Type.... Load.. Load status......

a.  Add gate
d.  Delete gate
c.  Change gate status
r.  Return to upper menu

Option [r]: a
```

3. At the next prompt, enter the name of the first gateway:

```
Gate name  [iceman]: hbgw1
```

   This re-displays the Gate menu with revised gateway DCI status, shown below:

4. Again at the Gate menu, add the next gateway by entering '**a**':

```
Gate............ Status...  Type.... Load.. Load status......
hbgw1           Never      Slave    100%   Participating

a.  Add gate
d.  Delete gate
c.  Change gate status
r.  Return to upper menu

Option [r]: a
```

5. You are again prompted for a gateway name. Enter the name of the second gateway at this prompt:

```
Gate name  [iceman]: hbgw2
```

This re-displays the Gate menu again with revised gateway DCI status:

```
Gate............ Status...  Type.... Load.. Load status......
hbgw1           Connected  Slave      0%   Participating
hbgw2           Never      Slave    100%   Participating
```

Selecting the Master Gateway

6. Now you must select the gateway that you want to be the 'master'. Enter '**c**' to change gate status:

```
a   Add gate
d   Remove gate
c   Change gate status
r.  Return to upper menu

Option [r]: c
```

7. This prompts with the first gateway (hbgw1) in the DCI and asks if you want to make it the 'master'. In this example, **hbgw2** is to be the 'master', so enter '**n**' :

```
hbgw1  Y/N: n
```

8. You are then prompted for the next gateway **hbgw2**. Enter '**y**' to change the status of **hbgw2.**

```
hbgw2  Y/N: y
```

9. Enter '**m**' at the next prompt to make **hbgw2** the 'master':

```
Select new gate status

e   Enabled
f   Failed
m   Master
r   Return to upper menu

Option [r]: m
```

This re-displays the Gate menu again with revised gateway DCI status information.

10. At the Gate menu, press RETURN to return you to the top menu:

```
a   Add gate
d   Delete gate
c   Change gate status
r.  Return to upper menu

Option [r]: RETURN
```

Note: The status change for **hbgw2**, is not yet completed on the host.

11. At the top menu, press RETURN.

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

This displays the current status as shown below.

```
hbmenu            Heartbeat status on iceman            Wed Feb 28 10:53:49

Host............ Status...  Arbiter status
iceman           Active     Never Connected
phoenix          Active     Never Connected

Gate............ Status...  Type.... Load.. Load status.....
hbgw1            Connected  Slave      0%   Participating
hbgw2            Connected  Master     0%   Participating

Application..... Dbase............ .Host........... Status....

Lans  -   h1.. Up      h2.. Up       t1.. Up      u1.. Up
```

Note: Status information on the gateways will differ slightly from that seen on the hosts. Refer to the topic *Description of hbmenu* later in this chapter.

Configuring Applications

Finally you need to add DCI entries for the databases to be configured with Heartbeat resilience. Each pair of FailSafe databases is identified by an Application name. This is an arbitrary name which identifies a FailSafe pair configured for Heartbeat operation. In the typical Heartbeat configuration described in this chapter (Figure 3-1), the application **oasis** comprises databases **oasis1** on **phoenix** and **oasis2** on **iceman**.

To configure the application, proceed as follows:

1. At the top menu, enter '**a**' to select the 'Application menu' option:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: a
```

2. At the Application menu, enter '**a**' to add an application entry to the DCI:

```
Application..... Dbase............ Host........... Status....

a.  Add application
d   Delete application
c   Change application
c   Toggle application status
r.  Return to upper menu

 Option [r]: a
```

3. At the next series of prompts, enter parameters using step 3a to configure a database connection, or using step 3b to configure a C server connection.

3a. To connect to database **oasis2** on your local host, in this example **iceman**, enter parameters similar to:

```
Machine name [iceman]: RETURN
Application name:      oasis
Database name:         oasis2
```

3b. To connect to a C server program (e.g. **cserver)** on the live system (e.g. **iceman**), enter the following.

```
Machine name [iceman]: RETURN
Application name:      cserver
Database name:         iceman
```

4. You are then prompted to specify the FailSafe status of the database or C server you have just entered. For this example, enter '**p**' to make **oasis2** on **iceman** the 'Primary' or to specify the **cserver** on the live host.

```
Select new application status

p    Primary
s    Secondary
u    Unknown
f    Failed
r    Return to upper menu.

 Option [r]: p
```

This returns you to the 'Application menu shown below in step 5.

Note: The application change just completed is not yet recorded on the display.

5. Now add the other database (**oasis1** on **phoenix**) or the **cserver** to the application DCI entry and make it the 'secondary'. Enter '**a**'

```
Application..... Dbase.......... .Host.......... Status....

a.  Add application
d   Delete application
c   Change application status
t   Toggle Application status
r.  Return to upper menu

Option [r]: a
```

6a. At the next series of prompts, enter parameters similar to:

```
Machine name [iceman]: phoenix
Application name:      oasis
Database name:         oasis1
```

6b. Alternatively, to connect to a C server program (e.g. **cserver**) on the standby system (e.g. **phoenix**), enter parameters similar to:

```
Machine name [iceman]: phoenix
Application name:      cserver
Database name:         phoenix
```

7. You are again prompted to specify the status of the database: Enter '**s**' to make **oasis1** the 'Secondary' or to specify the **cserver** on the standby host.

```
Select new application status

p    Primary
s    Secondary
u    Unknown
f    Failed
r    Return to upper menu.

 Option [r]: s
```

This returns you to the Application menu. You can continue this Application menu loop (steps 1 to 7) until all FailSafe database pairs requiring Heartbeat protection have been registered in the DCI.

8. Once all databases or C servers have been added to the DCI, press RETURN at the Application menu to return to the top menu:

```
Application..... Dbase............ Host.......... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary

a.  Add application
d   Delete application
c   Change application status
t   Toggle Application status
r.  Return to upper menu

Option [r]: RETURN
```

9. At the top menu, press RETURN to display the current status the Heartbeat system.

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

The revised status screen as seen on **iceman** will be similar to the following:

```
hbmenu            Heartbeat status on iceman          Wed Feb 28 10:56:57

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave      40%  Participating
hbgw2            Connected  Master     60%  Participating

Application..... Dbase..... .Host..... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary


Lans  -   h1.. Up      h2.. Up       t1.. Up      u1.. Up

RETURN for menu
```

Note: The status information on the gateways will be slightly different. For a description of the status information seen on the hosts and gateways, refer to the topic *Description of hbmenu* later in this chapter.

The configuration of the Heartbeat system is now complete. Ensure that the status information displayed on the two host consoles and all gateway consoles agrees.

Stopping Heartbeat

Heartbeat is stopped by executing **hbmenu** on each gateway and host. This is carried out by entering '**s**' at the top menu screen (shown below for a host).

```
hbmenu            Heartbeat status on iceman          Wed Feb 28 10:57:55

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected             40%  Participating
hbgw2            Connected             60%  Participating

Application..... Dbase.......... .Host.......... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary

Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: s
```

You can if you wish stop Heartbeat on the hosts only, without stopping the gateways. Heartbeat on the hosts sends a message to the gateway indicating that it is shutting down. This executes a clean shutdown without causing a fail condition.

Restarting Heartbeat

When you stop Heartbeat and restart it after it has been fully configured, you must wait until all connections have been built between the two hosts, and between the hosts and the Heartbeat gateways. This is indicated by 'Connect'/'Connected' status messages being displayed in the **hbmenu** status information on all systems. For details, refer to the description of the status information under the topic *Description of hmenu.*

CAUTION

Before restarting Heartbeat, ensure that the underlying FailSafe status of the databases is configured correctly, one database defined as the primary and the other as the secondary. The Heartbeat application status you define in this procedure must match the underlying FailSafe status. If it does not Heartbeat will not operate correctly.

Recovery After A Host or DBMS Failure

Effect of a Primary Failure

If the live host/primary database fails (e.g. **iceman/oasis2**), the live host will be shutdown, if possible, and all secondary databases on the surviving host will be re-configured as primaries. At the same time, users are switched to the surviving host and logged on to the newly configured primaries. The failed system must then be recovered and restarted as the standby The procedure to do this is detailed below under the topic *Recovery Procedure*.

Effect of a Secondary Failure

If the standby host/secondary database fails (e.g. **phoenix/oasis1**), the live host and its resident primaries are unaffected. Service to primary users is maintained, but FailSafe resilience is no longer available. It is therefore imperative to recover the failed standby system as quickly as possible. The procedure to do this is detailed below.

Recovery procedure

The procedure to recover full Heartbeat operation after a failure is as follows:

1. Repair the failed system.

2. Recover FailSafe operation using the **tlmenu** procedures described in the *RealityX Resilience Reference Manual* Chapter 17.

3. Having recovered FailSafe, use **hbmenu** to restart Heartbeat on the recovered host.

   Note: Heartbeat will still be running on the gateways, unless it has been stopped using the **hbmenu** 'Stop heartbeat' option.

   Initially, the **hbmenu** status screen will show the status of the databases on the previously failed host as being 'Failed'. For example:

```
Application..... Dbase..... .Host..... Status....
oasis            oasis2     phoenix     Failed
oasis            oasis1     iceman      Primary
```

CAUTION

Before starting Heartbeat, ensure that the Heartbeat application status you define matches the underlying FailSafe status. If it does not, Heartbeat will not operate correctly.

4. Change the status of the Application **oasis** on either host, as described in steps 5 to 8.

5. At the top menu, enter '**a**' to select the 'Application menu' option:

```
s.  Stop heartbeat
h   Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

 Option [r]: a
```

6. At the Application menu, enter '**c**' to change application status.

```
Application..... Dbase............ Host........... Status....
oasis            oasis2           iceman          Failed
oasis            oasis1           phoenix         Primary

a.  Add application
d   Delete application
c   Change application status
t   Toggle Application status
r.  Return to upper menu

Option [r]: c
```

7. You are then prompted to specify the new status.

CAUTION

Once you change the status to 's', the **router** will start to build the secondary connection. So before you do this, ensure that the underlying FailSafe and Heartbeat status agree.

Enter '**s**', as below, to make **oasis2** a 'Secondary'.

```
Select new application status

p    Primary
s    Secondary
u    Unknown
f    Failed
r    Return to upper menu.

 Option [r]: s
```

8. You are then presented with a prompt for each application database, asking if you want to change its status. Enter '**y**' to change the status of oasis2 to secondary. Enter RETURN to maintain the current Primary status of oasis1. For example:

```
oasis:oasis2:phoenix:F   y/n [n]: y
oasis:oasis1:iceman:P   y/n [n]: RETURN
```

This returns you to the 'Application menu' with the revised status displayed.

9. At the Application menu, press RETURN to return to the top menu:

```
Application..... Dbase.....  Host..... Status....
oasis            oasis2           iceman          Secondary
oasis            oasis1           phoenix         Primary

a.  Add application
d   Delete application
c   Change application status
t   Toggle Application status
r.  Return to upper menu

 Option [r]: RETURN
```

10. At the top menu, press RETURN to display the current status of the Heartbeat system:

```
s.  Stop heartbeat
h.  Host menu
l   Lan menu
g   Gate menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

The status information displayed will be similar to the following:

```
hbmenu            Heartbeat status on iceman          Wed Feb 28 11:15:09

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave     40%   Participating
hbgw2            Connected  Master    60%   Participating

Application..... Dbase............. Host........... Status....
oasis            oasis2            iceman          Secondary
oasis            oasis1            phoenix         Primary


Lans  -   h1.. Up      h2.. Up       t1.. Up      u1.. Up

RETURN for menu
```

The status of the Heartbeat application should now be consistent with the re-configured FailSafe status, and normal Heartbeat operation is re-established. Heartbeat is now running in reverse configuration; **phoenix/oasis1** as the primary connection and **iceman/oasis2** as the secondary standby connection.

Toggling the Application

Once Heartbeat operation is restored, it may be desirable to reverse the primary/secondary roles of databases again to what they were before the failure. For example, the systems in the Heartbeat configuration may be asymmetrical, where one machine is larger and more powerful than the other. To do this, refer to the procedure *Toggling Primary/Secondary Databases in a Heartbeat System* later in this chapter.

Recovery After a Master Gateway Failure

CAUTION

If the master gateway has failed, Heartbeat switch-over will not be executed if a fatal error occurs on the primary host. It is therefore vital to re-configure Heartbeat quickly in order to restore Heartbeat protection.

Failure of the master gateway (e.g. **hbgw2)** is detected by the host which responds by marking the gateway DCI entry as failed and passing the name of the failed gateway to the **gate** fail script. The **gate** script then passes a message to the host system console informing you that the gateway has failed. For example:

```
Heartbeat Gate hbgw2 failed
```

If you run **hbmenu** on the host iceman the status screen will show:

```
hbmenu            Heartbeat status on iceman          Wed Feb 29 10:51:56

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave     90%   Participating
hbgw2            Failed     Master   100%   Participating

Application..... Dbase.......... .Host.......... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up



RETURN for menu
```

The aim of this procedure is to re-configure Heartbeat so that one of the slave gateways becomes the master and the failed gateway is marked as a slave. Heartbeat protection is then restored as quickly as possible. The failed master can then be repaired and restarted later as a slave.

Current users of the failed gateway are hung or disconnected, depending on network time-outs. The user load is then redistributed among the surviving gateways by the loadsharing software. Users must then retry logging on when they will be allocated a different gateway by the Loadsharing software.

To re-configure Heartbeat gateways, use **hbmenu** on the host console to change the gate DCI entries, as follows:

1. At the top menu enter '**g**' to select the Gate menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: g
```

2. This displays the Gate menu. Enter '**c**' to change gate status:

```
Gate............ Status...  Type.... Load.. Load status......
hbgw1           Connected  Slave    60%    Participating
hbgw2           Failed     Master   100%   Participating

a   Add gate
d   Delete gate
c   Change gate status
r.  Return to upper menu

 Option [r]:
```

3. You are then presented with a prompt for each gateway, asking if you want to change its status. At one of the slave gateway prompts, enter '**y**'. For example:

```
hbgw1   y/n [n]: y
```

4. At the next menu, enter '**m**' to make gateway **hbgw1** the master.

```
e   Enabled
f   Failed
m   Master
n   Non participating
r   Return to upper menu

Option [r]: m
```

5. Now press RETURN to re-display the **hbmenu** status screen. This should now show:

```
hbmenu            Heartbeat status on iceman            Wed Feb 29 10:52:30

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Master    90%   Participating
hbgw2            Failed     Slave    100%   Participating

Application..... Dbase............ Host........... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary

Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up


RETURN for menu
```

6. Run **hbmenu** on each host and gateway to ensure that changes are propagated to all other hosts and gateways.

7. Now repair and re-start the failed gateway **hbgw2**.

8. After restarting the repaired gateway, change its status to 'Enabled'. For example, enter:

```
hbgw2   y/n [n]: y
```

9. Finally at the next menu, enter 'e' to enable **hbgw2**.

```
e   Enabled
f   Failed
m   Master
n   Non participating
r   Return to upper menu

Option [r]: e
```

hbgw2 will then operate as a slave again and be allocated users by the loadsharing software.

Re-configuring Heartbeat After a Slave Gateway Failure

If a slave gateway, (e.g. **hbgw1**) fails, the failure is detected by the host which responds by marking the gateway DCI entry as 'Failed' and passing the name of the failed gateway to the **gate** fail script. The **gate** script then passes a message to the system console informing you that the gateway has failed. For example:

```
Heartbeat Gate hbgw1 failed
```

If you run **hbmenu** the status screen will show

```
hbmenu            Heartbeat status on iceman            Wed Feb 29 10:51:56

Host............ Status...  Arbiter status
iceman           Local      Connected
phoenix          Connect    Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Failed     Slave     100%  Participating
hbgw2            Connected  Master     90%  Participating

Application..... Dbase............ Host........... Status....
oasis            oasis2            iceman          Primary
oasis            oasis1            phoenix         Secondary


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up


RETURN for menu
```

Overall Heartbeat operation and users connected to other gateways are unaffected by a slave gateway failure. Current users of the failed gateway are hung or disconnected, depending on network time-outs. The user load is then redistributed among the surviving gateways by the load sharing software. Users must then retry logging on when they will be allocated a different gateway by the load sharing software.

To restore the gateway to service you must repair and re-start it, then use **hbmenu** to remove the 'Failed' status from the gate DCI entry, as follows:

1. Enter '**g**' at the top menu to select the Gate menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: g
```

2. At the Gate menu, enter '**c**' to change gate status:

```
Gate............ Status...  Type.... Load.. Load status
hbgw1            Failed     Slave    100%   Participating
hbgw2            Connected  Master    90%   Participating

a   Add gate
d   Delete gate
c   Change gate status
r.  Return to upper menu

 Option [r]: c
```

   You are then presented with a prompt for each gateway, asking if you want to change its status.

3. At one of the slave gateway prompts, enter **'y'**. For example:

```
hbgw1   y/n [n]: y
```

4. Enter 'e' at the next menu:

```
e   Enabled
f   Failed
m   Master
n   Non participating
r   Return to upper menu

Option [r]: e
```

This returns you to the Gate menu with revised status information.

5. At the Gate menu, press RETURN to return to the top menu:

```
Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave      0%   Participating
hbgw2            Connected  Master    90%   Participating

a   Add gate
d   Delete gate
c   Change gate status
r.  Return to upper menu

 Option [r]: RETURN
```

6. At the top menu, press RETURN to display the current status of the Heartbeat system:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

7. Now redisplay the Heartbeat status screen. This should show:

```
hbmenu            Heartbeat status on iceman            Thu Feb 29 10:52:29

Host............ Status...  Arbiter status
iceman           Local      Connected
phoenix          Connect    Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave      0%   Participating
hbgw2            Connected  Master    90%   Participating

Application..... Dbase............. Host........... Status....
oasis            oasis2             iceman          Primary
oasis            oasis1             phoenix         Secondary


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up


RETURN for menu
```

This will take time to propagate across to the **dci.file**s on the other host and gateways. Once the DCI gate entries are changed, the gateway becomes available to Heartbeat users.

Toggling Primary/Secondary Databases in a Heartbeat System

This procedure is used to reverse the primary/secondary roles of databases in a Heartbeat configuration. It is particularly useful for use in an asymmetric FailSafe configuration where the standby machine is smaller and less powerful than the live machine and is only intended to be used as a backup while the failed machine is being repaired.

> **CAUTION**
>
> Ensure that the Heartbeat application status always reflects the underlying FailSafe status. If it does not Heartbeat will not operate correctly.

1. Login as super user to the primary host and run **tlmenu** Miscellaneous option 5 'Swap Primary and Secondary Databases'. Refer to the *RealityX Resilience Reference Manual* Chapter 18 for details. If **oasis1** is the primary and **oasis2** the secondary, after completing this step, **oasis1** becomes the secondary and **oasis2** the primary.

2. Login as super-user on the new primary host and run **tlmenu** Miscellaneous option 3 'Log Off Users and Lock Failsafe Database'.

3. Enter **hbmenu** on the new primary host. This will display a status screen similar to that opposite. Note that the Failsafe status and Heartbeat status of the databases is now out of synchronisation.

   The Heartbeat application status will now be the reverse of the new underlying FailSafe status. It is therefore necessary to toggle the Heartbeat application status to match the underlying FailSafe status. To do this, perform steps 4 to 6.

4. Enter '**a**' at the top menu to select the Application menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
r   Return to display loop
q.  Quit hbmenu

Option [r]: a
```

```
hbmenu            Heartbeat status on iceman            Wed Feb 28 12:48:56

Host............ Status...  Arbiter status
iceman           Local      Connected
phoenix          Connect    Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave     40%   Participating
hbgw2            Connected  Master    80%   Participating

Application..... Dbase............ Host........... Status....
oasis            oasis2            iceman          Secondary
oasis            oasis1            phoenix         Primary


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

RETURN for menu
```

5. Enter '**t**' at the Application menu to toggle the Heartbeat application status:

```
Application..... Dbase............ .Host........... Status....
oasis            oasis2            iceman          Secondary
oasis            oasis1            phoenix         Primary

a   Add application
d   Delete application
c   Change application status
t   Toggle application
r.  Return to upper menu

Option [r]: t
```

6. To toggle the status for a specified database, enter '**y**' at the next prompt: The message with this prompt indicates the status that will result from toggling.

   For example:

```
oasis y/n [n]: y
```

   displays the message:

```
toggling oasis will result in:

oasis oasis1 host1 s
oasis oasis2 host2 p
```

On entering '**y**' the Application menu is displayed again with the latest status information, showing the reversal of primary and secondary roles.

7. At the Application menu, press RETURN to return to the top menu:

```
Application..... Dbase..... .Host..... Status....
oasis            oasis1      iceman      Primary
oasis            oasis2      phoenix     Secondary

a   Add application
d   Delete application
c   Change application status
t   Toggle application
r.  Return to upper menu

Option [r]: RETURN
```

8. At the top menu, press RETURN again to return to the status display.

```
s   Stop heartbeat
h   Host menu
g   Gate menu
l   Lan menu
a   Application menu
u   Display users file
r   Return to display loop
q.  Quit hbmenu

Option [r]: RETURN
```

This should show the following:

```
hbmenu            Heartbeat status on iceman           Wed Feb 28 12:49:19

Host............ Status...  Arbiter status
iceman           Local      Connected
phoenix          Connect    Connected

Gate............ Status...  Type.... Load.. Load status
hbgw1            Connected            40%   Participating
hbgw2            Connected            60%   Participating

Application..... Dbase........... .Host........... Status....
oasis            oasis2            iceman          Primary
oasis            oasis1            phoenix         Secondary


Lans  -     h1.. Up       h2.. Up        tl.. Up       ul.. Up

RETURN for menu
```

If steps 1 and 2 were completed correctly, then the Heartbeat application status should now match the underlying FailSafe status and normal Heartbeat operation is re-established.

9. Quit **hbmenu** on the new primary host.

10. Login as super-user on the new secondary host and run **hbmenu**. Check that the application status shows the changed DCI. This will take a short time to propagate across the network.

11. Finally run **tlmenu** Miscellaneous option 4 'Unlock Failsafe Database' on the new primary host to allow users to logon again.

Description of hbmenu

Purpose

hbmenu displays menus and executes interactive prompt-driven procedures, providing a user-friendly interface for administering Series X Heartbeat.

Note: Although, the overall structure of menus and status screens is now established, as described in this manual, **hbmenu** is still be subject to minor enhancements which may not always be reflected in the manual.

The utility is executed by entering the **hbmenu** command at the UNIX shell.

Syntax

hbmenu {*options*}

Options

start Starts Heartbeat before-displaying the first status screen.

quick Only displays status and then exits. Does not display menu.

Restrictions

Super-user permissions required.

Location of Utility

The **hbmenu** script resides in the **£HBROOT/bin** directory.

Menu-Driven Procedures

hbmenu executes Heartbeat administrative procedures utilising three menu levels and a fourth prompt driven lowest level.

Top Level Menu at Host Console

The top level menu together with the current database configuration information, is displayed when you enter **hbmenu**, as shown on the next page.

Start Heartbeat Menu

Selecting one of the options in the top menu executes the following:

s Starts Heartbeat and goes to the main display loop:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan Menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]:
```

```
hbmenu              Heartbeat status on iceman          Wed Feb 28 10:51:56

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave     40%   Participating
hbgw2            Connected  Master    60%   Participating

Application..... Dbase.......... Host............ Status....
 oasis            oasis2         iceman           Primary
 oasis            oasis1         phoenix          Secondary

Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

s.  Start heartbeat
r.  Redisplay main menu
q.  Quit hbmenu

Option [r]:
```

r Re-displays the top level menu and status display.

q Exits to the UNIX shell.

Note: The status information displayed on the master and slave gateways is a reduced set of that provided on the hosts. Refer to the description of the status information in this section for details.

Stop Heartbeat Menu While Heartbeat is running, the 'Stop heartbeat' menu becomes the main menu:

```
s.  Stop heartbeat
h.  Host menu
g   Gate menu
l   Lan Menu
a   Application menu
u   Display users file
f   Fail machine
r   Return to display loop
q.  Quit hbmenu

Option [r]:
```

Selecting one of the above options in the 'Stop heartbeat' menu executes the following:

s Stops Heartbeat and returns you to the second-level menu screen described above.

h Displays the **Host menu**, shown below, with options to change the DCI host entries.

```
a   Add host
d   Delete host
c   Change host status
r.  Return to upper menu

Option [r]:
```

g Displays the **Gate menu**, shown below, with options to change the DCI gateway entries.

```
a   Add gate
d   Delete gate
c   Change gate status
l   Change load sharing status
r.  Return to upper menu

Option [r]:
```

l Displays the **Lan menu**, shown below, with options to change the DCI LAN entries.

```
a   Add lan
d   Delete lan
c   Change lan status
r.  Return to upper menu

Option [r]:
```

a Displays the **Application menu**, shown below, with options to change the DCI application entries.

```
a   Add application
d   Delete application
c   Change application status
t   Toggle Application status
r.  Return to upper menu

option [r]:
```

u Displays the contents of the users file **/etc/hb/users**. This contains status information for each Heartbeat user connection. For example:

```
Plid ................Gate .......Guid......Huid......Appl.......SecCon

UNIX-200521-pts0      hbgw1       hbs       hbs,hbs12 oasis      Conn
UNIX-200521-pts3      hbgw3       hbs       hbs,hbs12 oasis      Conn
UNIX-200521-pts2      hbgw1       hbs       hbs,hbs12 oasis      Conn
UNIX-200521-pts4      hbgw2       hbs       hbs,hbs12 oasis      Conn
```

The information displayed is, as follows:

Plid	The Physical Location Identifier of the user
Gate	Gateway connected to
Guid	User-id on gateway
Huid	User-id on host
Appl	Application connected to
SecCon	Secondary connection status

f Shuts down a specified host. The default is the local host. Entering '**f**' displays a prompt similar to:

```
Host/Gate name [phoenix]    :
```

followed by:

```
Fail phoenix y/n [n]
```

then,

```
Reason                      : (on host only)
```

If you enter '**y**', the specified machine (in this case, phoenix) will be shutdown. If its the live host, then Heartbeat switch-over will be executed.

Host, Gate Lan, and Application Menu Options

Selecting one of the options in the Host, Gate or Application menus, described above, executes the following:

a Enables you to add a Host, Gateway, Application or LAN entry to the database configuration information. On selecting this option you are prompted for the Host, Gate Application, or LAN name.

d Enables you to remove a Host, Gateway, Application or LAN entry to the database configuration information. You are prompted for each Host, Gateway or Application in turn, asking if you wish to remove it (y/n).

c Enables you to change the status of a Host, Gateway, Application or LAN entry to the database configuration information. You are prompted for each Host, Gateway or Application entry in turn asking if you wish to change its status (y/n).

> **CAUTION**
>
> Do not remove an entry from the database configuration information until you are sure that it is no longer being used by a current user. If you do, Heartbeat protection will no longer be available to users of that entry.

If you select to change the status of an application, then a menu is displayed, giving you the status options you can change to. The menu displayed is:

```
Select new application status

p    Primary
s    Secondary
u    Unknown
f    Failed
r    Return to upper menu.

 Option [r]:
```

t This only appears in the Application menu and enables you to toggle the primary/secondary status of the two databases associated with an application. You are prompted for the application names and asked if you wish to toggle their databases (y/n).

Status Information

To display the current status of the Heartbeat system, enter '**r**' or RETURN at the top menu.

On a host, for example **iceman,** the status display shows:

```
hbmenu            Heartbeat status on iceman          Wed Feb 28 10:51:56

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...  Type.... Load.. Load status......
hbgw1            Connected  Slave      40%  Participating
hbgw2            Connected  Master     60%  Participating

Application..... Dbase.......... .Host.......... Status....
oasis            oasis2           iceman          Primary
oasis            oasis1           phoenix         Secondary


Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up

RETURN for menu
```

The display divides into three tables:

Host table, comprising the following fields:

Host — Lists names of hosts configured in the Heartbeat system.

Status — Indicates the status of the client-server link between the local host and another host in the Heartbeat system. This provides the path for the transfer of database configuration information between hosts.

Status can be one of the following:

- Active — Indicates that the client-server link with the specified host has been set up.
- Lost — Indicates that the client-server link with the specified host has been lost
- Never — Indicates that the specified host has never been connected.
- Stopped — Indicates that Heartbeat has been stopped on the remote host.

Arbiter Connection

Indicates the status of the connection between a host and the **hbarbite**r process on the master gateway.

The status can be one of the following:

Connected	Indicates that the arbiter connection to the specified remote host has been set up.
Lost	Indicates that the arbiter connection to the specified remote host has been lost.
Stopped	Indicates that Heartbeat has been stopped on the master gateway.

Gate table, comprising the following fields:

Gate	lists names of gateways configured in the Heartbeat system.
Status	indicates the status of the client-server link between the current host and the Heartbeat gateways.

Connected	Indicates that the client-server link with the specified gateway has been set up.
Failed or Lost	Indicates that the client-server link with the specified gateway has been disconnected.
Stopped	Indicates that Heartbeat has been stopped on the specified gateway.

Type	specifies ‘Master’ or ‘Slave’.
Load	Indicates the number of user connections active on a gateway as a percentage of the maximum number permitted. The maximum number is configurable in the **hbarbiter** configuration file.
Load Status	indicates whether the gateway is 'Participating' or 'Non-participating' in Heartbeat Load sharing. See Chapter 4. It may also indicate 'Failed' if the gateway is down.

Application table, comprising the following fields:

Application	lists the names of applications configured in the Heartbeat system.
Dbase	lists the names of the FailSafe databases associated with the application names.
Host	lists the names of the hosts on which the databases reside.
Status	indicates the FailSafe status of the databases marked in the DCI, as follows: Primary, Secondary, Failed, or Unknown.

Some of the information appearing in the status display on the hosts does not appear on the gateways.

On the gateways, the status display does not show: Host Status, Gate Status, Load or Load status. Also slave gateways do not show Arbiter Connection status. Only the master gateway executes the arbiter process.

For example, on the master gateway **hbgw2**, the status display shows the following information:

```
hbmenu              Heartbeat status on hbgw2          Wed Feb 28 10:51:56

Host............ Status...  Arbiter status
iceman           Active     Connected
phoenix          Active     Connected

Gate............ Status...   Type.... Load.. Load status......
hbgw1            Connected
hbgw2            Connected

Application..... Dbase........... .Host........... Status....
oasis            oasis2           iceman           Primary
oasis            oasis1           phoenix          Secondary

Lans  -   h1.. Up      h2.. Up       t1.. Up      u1.. Up


Return for menu
```

whereas, on a slave gateway, e.g. **hbgw1** the status display shows:

```
hbmenu              Heartbeat status on hbgw2          Wed Feb 28 10:51:56

Host............ Status...  Arbiter status
iceman
phoenix

Gate............ Status...   Type.... Load.. Load status......
hbgw1            Connected
hbgw2            Connected

Application..... Dbase........... .Host........... Status....
oasis            oasis2           iceman           Primary
oasis            oasis1           phoenix          Secondary

Lans  -   h1.. Up      h2.. Up       tl.. Up      ul.. Up


Return for menu
```

Description of hbrouter Command

Purpose

Creates and maintains Heartbeat user connections to FailSafe host systems configured in a Heartbeat system. It is executed from a user's **.profile** script when the user logs in to a Heartbeat gateway.

Syntax

hbrouter { -d *application*} {**-u** *user*, *password*} {**-n**}

Options

-d Specifies the application to connect to.

-u Specifies the user-id and optionally the password to be used to connect to the host.

-n Specifies no password required and no password prompt when connecting to the host.

Parameters

application The name of the application to be connected to. If the **-d** option is omitted the default is **$LOGNAME**.

user The user-id to be used to connect to the host. If the **-u** option is omitted, the default is the UID number.

password The password to be used to connect to the host. When provided, the user is logged on to the host using the password given and no password is prompted for at the host.

> **CAUTION**
>
> Use of *password* in the **hbrouter** command line **.profile** is primarily a debug facility. It is not recommended that you use this in operational systems, as it constitutes a security risk.

If you omit the *password* parameter, you are prompted for it on the host.

If you enter **CR** for *password,* the password on the host is simply a RETURN and there is no password prompt.

If you enter **NULL** for **password**, no password is required and no password is prompted for on the host. This is the same as using the **-n** *user* option.

Comments

Refer to Chapter 2 for a description of how **hbrouter** establishes a Heartbeat user connection and executes automatic switch over of users to the standby host. This section discusses the configuring of the **hbrouter** command line in **.profile**.

Example

```
exec hbrouter -d oasis -u oasis
```

This line in the **.profile** will execute **hbrouter** to make a connection to the host containing the primary database for the oasis application. It prompts for the password to login as the **oasis** UNIX user-id on the host.

Chapter 4
Administering Gateway Loadsharing

This chapter discusses the operation of loadsharing by Heartbeat gateways, for both TCP/IP and OSI networks, and details the parameters which need to be defined in order to configure the software.

Introduction

Loadsharing software installed in a Heartbeat system manages the assignment of new user connections to Heartbeat gateways, so as to distribute the user population evenly across the gateways and prevent any gateway from becoming overloaded.

If the overall user load on a Heartbeat system becomes abnormally high, causing the load on one or more of the gateways to exceed a defined limit, the loadsharing software directs any new logon attempts away from the overloaded gateway(s) and toward the more lightly loaded ones.

If a gateway fails, the hosts mark the gateway as 'Unavailable' for new connections and the software then re-routes any Connect Requests bound for the failed gateway to an alternative gateway. Ideally the loadsharing should be configured to allow the failed gateway's user load to be distributed evenly across the remaining gateways.

The re-distribution of user load only applies to new connections. It does not change existing user connections. The change, therefore, is not instantaneous, but is effected over a period of time as users log off and new users log on.

To reduce the impact of a gateway failure it is recommended that users on a given terminal server are distributed evenly across the available Heartbeat gateways.

The administrator can also specify a gateway as no longer participating in loadsharing. See the description of the **hbmenu** Gate menu in Chapter 3. The loadsharing process then no longer routes new connections to that gateway, although existing connections are unaffected. This enables the orderly and gradual shutdown of a gateway as the number of connections through the gateway decreases over a period of time, while no new connections are allocated to it.

Currently, loadsharing is supported for communications networks using the following protocols:

- Transmission Control Protocol/Internet Protocol (TCP/IP)
- Open System Interconnection (OSI)

Configuring Loadsharing

The loadsharing software is designed to operate without any special user actions or reconfiguration of network devices after initial site set-up. There are, however, a number of parameters which need to be defined at initial set-up and which can be re-configured to optimise loadsharing.

The following must be defined in **$HBROOT/rrgen/rrgen.cfg**:

- Lower gateway load threshold (low_threshold)
- Upper gateway load threshold (high_threshold)
- Loadsharing granularity
- Default gateways assigned to connect names

Also for the TCP/IP protocol, the following must be defined**:**

- Internet addresses for the gateways

Refer to the next topic *Description of Loadsharing Software* for a detailed description of the parameters listed above. Specifically, they are described under the sub-topic *Loadsharing Configuration File*.

The gateways which are to participate in the scheme must also be defined. This is done using the **hbmenu** utility at the Gate menu, described in Chapter 3.

Description of Loadsharing Software

The Loadsharing process on the live host (Figure 4-1) monitors each Heartbeat gateway by looking at the status information logged in the gateways file /**etc/hb/gates**. While the loading on each gateway stays below specified thresholds, user connections to the gateways are set up using the default gateways assigned to each connect name, as defined in the **rrgen** configuration file **/$HBROOT/rrgen/rrgen.cfg.**

Note: A connect name is that which is supplied by the user to the communications software in order to request a connection through a gateway.

Both the gateway load thresholds and default gateway assignments are specified in the configuration file **/$HBROOT/rrgen/rrgen.cfg**. The default gateway assignments should be set up so that the total user population is shared evenly across all gateways.

If the loading on a particular gateway exceeds the thresholds specified in the **rrgen.cfg** file, then any new logon attempts which would normally default to the overloaded gateway are diverted to an alternative gateway. Refer to the topic *Loadsharing Algorithm* in this chapter for a description on how the load threshold parameters are used. The method of re-assigning gateways to connect names is different for TCP/IP and OSI protocols.

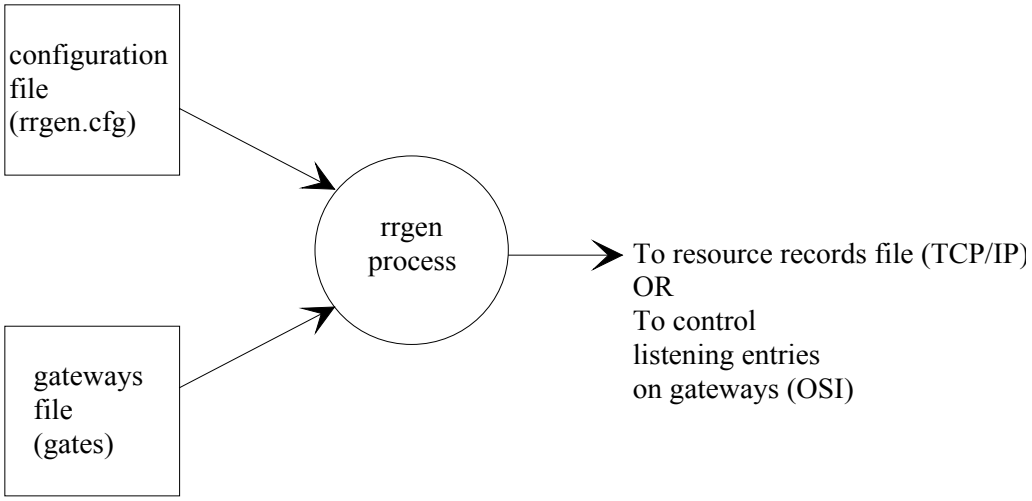


Figure 4-1. Loadsharing Process

Gateways Status File

Gateway status information, including level of user load and whether or not the gateway is participating in the system, is sent to the Heartbeat hosts at regular intervals by each of the gateways and saved in the gateways file **/etc/hb/gates**.

The gateways file **/etc/hb/gates** consists of plain text, similar to the following:

Note: This file should not be edited.

```
Heartbeat gateways file
30
gway1:152.114.200.1:P:55
gway2:152.114.200.1:P:78
gway3:152.114.200.1:P:61
```

The first line is a comment.

The second line is the time to live in seconds for the gateways file information.

Third and subsequent lines comprise the following fields:

1 Gateway name

2. Gateway Internet address

3. Gateway status (P - Participating in ,
   N - Non-participating, F - Failed)

4. Percentage load on gateway

Configuration File

The configuration file **/$HBROOT/rrgen/rrgen.cfg** is a text file that can be edited to define the default assignments for connecting to Heartbeat gateways and the loading limits on a gateway. A typical example of text given in **rrgen.cfg** is given below.

```
# Heartbeat DNS configuration file
#
low_threshold                 60
high_threshold                80
loading-granularity            5
fixed_name     hosta          152.114.200.1 (TCP/IP only)
fixed_name     hostb          152.114.200.2 (TCP/IP only)
connect_name   cname1         gway2
connect_name   cname2         gway1
```

Lines beginning with # are comments.

The configuration data comprises a number of lines beginning with an identifier, followed by one or more fields of data separated by white space. The identifiers and their associated data are as follows:

fixed_name Specifies a domain name in a TCP/IP network which maps directly to an internet address. For example gway1 is at internet address 152.114.200.1. These mappings are not changed by the **rrgen** process.

connect_name Specifies the default gateway connection for the specified connect_name. So for example, under normal loading conditions, a connect request to 'cname1' would connect you to 'gway2'. However if gway2 is overloaded, then the input from **/etc/hb/gates** will cause the **rrgen** process to generate an output which re-assigns 'cname1' to a different gateway.

loading-granularity
Specifies the width of the bands used in the Loadsharing algorithms. All gateways with a load within a band are considered equivalently loaded. For example, with a granularity of 5, a load of 50 and a load of 54 would be considered equivalent.

low_threshold Specifies the lower loading threshold for the gateways.

high_threshold Specifies the overload threshold for the gateways.

low_threshold and *high_threshold* are read by the process **rrgen** and used by the algorithm to control the assignment of gateways to connect names to implement the mechanism. Refer to the topic *Loadsharing Algorithm* in this chapter.

Configuration Utility

The Heartbeat Daemon Configuration Utility provides easy to follow procedures for adding, deleting and modifying configuration information in the Loadsharing Configuration File **$HBROOT/rrgen/rrgen.cfg.**

To run the configuration utility proceed as follows:

1. Login as super-user.
2. Change directory to **$HBROOT/rrgen**.
3. Enter **`config`** at the shell prompt.

This displays the top configuration menu which contains options to specify the required:

- Loading thresholds (TCP/IP only)
- Fixed names of gateways (TCP/IP only)
- Service names (TCP/IP) or connect names (OSI) of gateways

You can move up and down the menu using the arrow keys and select one by pressing RETURN. You can exit the program at this point by selecting the appropriate option or pressing ESC.

On selecting a top menu option you are presented with a set-up menu screen for the appropriate configuration items. If you are unsure on how to manipulate the screen, use the Left arrow to highlight the **`Help`** option at the top of the screen and press RETURN. This will display a Help Window giving information on how to use the utility.

rrgen Process

The resource records generator (**rrgen)** process runs continuously in the background on the hosts, taking input from the gateways file **/etc/hb/gates** and configuration file **/$HBROOT/rrgen/rrgen.cfg** and periodically generating output to control the connect name-gateway assignments for optimum . The **rrgen** output period is specified in the environment variable **HB_GATE_LOAD_TTL**.

rrgen takes the configuration information and gateways status information and, using the algorithm, assigns connect names to gateways participating in the loadsharing system. Any gateways not participating in, or which have failed, are not assigned any connect names.

For the TCP/IP protocol, the **rrgen** output is used to re-generate the Heartbeat DNS zone information in the host's Resource Record File. The main DNS server on the host is then signalled to reload its zone information into memory Refer to the topic *Loadsharing in a TCP/IP Network*.

For the OSI protocol, the **rrgen** output is used to switch the listening entries on and off in each of the Heartbeat gateways, so that heavily loaded gateways have less listening entries switched on than lightly loaded gateways. Refer to the topic *Loadsharing in an OSI Network*.

The **rrgen** executable and all associated software, including configuration and statistics files are held in the **/$HBROOT/rrgen** directory.

The transfer utility **syncXferd** operates on the two hosts to copy data changed in the **rrgen.cfg** file on one host to the other host, so that the two **rrgen.cfg** files are kept in step and changes only need to be made once.

Algorithm

The **rrgen** process reads two parameters from **rrgen.cfg** which are used by the algorithm, *low_threshold* and *high_threshol*d. Using these threshold values, **rrgen** recognises the following states:

State 1	All gateways are loaded below the low threshold.
State 2	At least one gateway is loaded above the low threshold, but all gateways are below the high threshold.
State 3	At least one gateway is loaded above the high threshold.
State 4	All gateways are loaded above the high threshold.

On recognising one of the above states, the following actions are taken:

If state 1	**rrgen** assigns each connect name to its default gateway defined in **rrgen.cfg**, provided the gateway is participating in the loadsharing. If a gateway is not participating then the connect names defaulted to that gateway are shared among the rest of the gateways.
If state 2	No change to the current connect name assignments, provided the participating gateways are unchanged, since the last time connect names were assigned. If the participating gateways have changed, then **rrgen** assigns the connect names to the participating gateways in inverse proportion to their user load, i.e. the least loaded gateways are assigned the most names.
If state 3	**rrgen** assigns the connect names to the participating gateways in inverse proportion to their user load, i.e. the least loaded gateways are assigned the most names, but excludes all gateways over the high threshold.
If state 4,	**rrgen** assigns the connect names to the participating gateways with a sensitivity factor so that similar loads are treated as equivalent.

The algorithm attempts to assign all connect names to their default gateway whenever possible, so that, on a daily basis, the connect name assignments remain similar.

Statistics

A statistics file is generated for each day **rrgen** is executed. The name of the file is the date with the **stats** extension. For example:

31Mar1994.stats

A new file is created in /**usr/rrgen**, the first time after midnight that **rrgen** is executed. The last seven day's files are maintained in **/usr/rrgen**. Earlier dated files are removed.

A line of statistical data is generated every time the DNS information changes and every ten executions of **rrgen,** while the DNS information remains unchanged.

A typical line of statistical data is as follows:

```
06:02:12 PM gway1:005:002:-002:P  gway2: 20:004:+001:P
```

First the time of day is shown, then there is an entry for each gateway with fields separated by colons. The fields are:

1. gateway name

2 gateway load

3. Number of service names currently assigned to gateway

4. Change in number of service names assigned to gateway since the last line of statistics

5. Status of gateway, (P)articipating, (N)on-participating or (F)ailed

Loadsharing in a TCP/IP Network

Domain Name System

TCP/IP network connections to Heartbeat gateways are set up using the Domain Name System (DNS) mechanism. Using the DNS mechanism, a user specifies a gateway name or connect name at the user device, which maps to an internet address in the Heartbeat DNS zone information held centrally in the Resource Records File on one of the hosts. A DNS server process in the host obtains the appropriate internet address from the Heartbeat DNS zone information and supplies it to the user. This enables efficient gateway address maintenance as, in the event of a gateway's address changing, the address entry has to be changed only once in the DNS zone information, instead of locally in each user device.

Loadsharing

The **rrgen** process on each host takes the information from the gateways file and **rrgen** configuration file and generates the resource records required for the Heartbeat DNS zone which it stores in a Resource Records File **etc/named.data/hbeat.hosts** on the host. It then signals the DNS server to read it into memory.

A DNS server with associated Heartbeat zone information is supported on each host. See Figure 4-2.

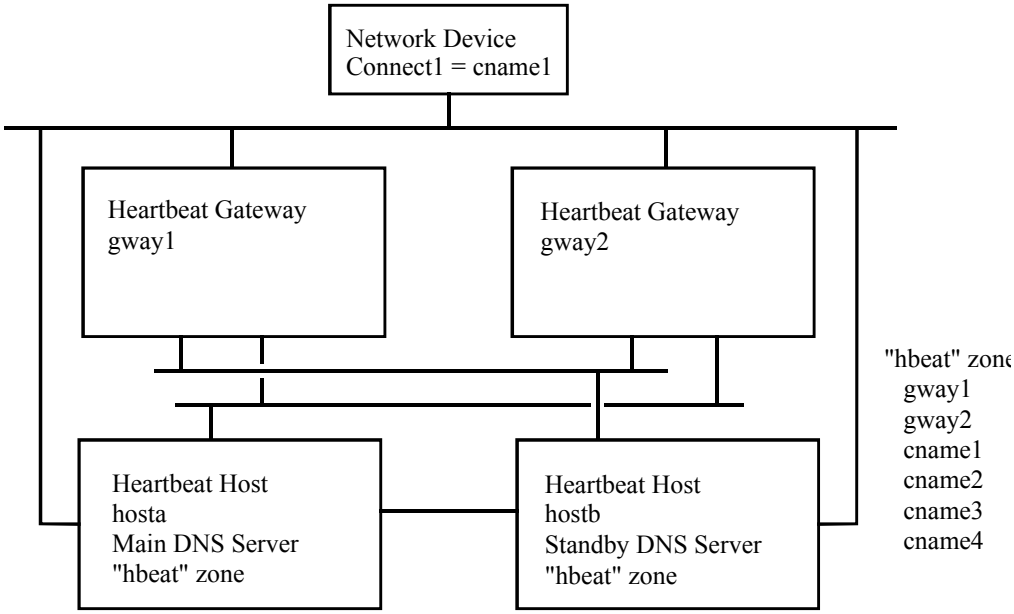


Figure 4-2. Example of TCP/IP Heartbeat Configuration

The DNS server on either host can deal with a connect request containing a specified connect name by looking at the DNS information and ascertaining the gateway currently assigned to that name and the internet address assigned to that gateway. It then returns the address to the network device to enable the user to establish the connection.

If the DNS server on one host cannot be accessed, then the server on the other host can be used.

Note: Both **rrgen** and the DNS server are stopped and started with Heartbeat on the hosts, in order to stop **rrgen** and the DNS server from 'looking' at old data. Hence, a users can only resolve a DNS name while Heartbeat is running on at least one host.

Loadsharing in an OSI Network

Alternate Host Addressing

OSI network connections to Heartbeat gateways are set up using connect names that are defined locally in each network device. In order to participate in the Loadsharing scheme and be capable of re-routing to another gateway, in the event of a gateway failure or overload condition, each network device must support 'alternate host' addressing (sometimes called 'host hunting), whereby if the first connection attempt fails, a reconnect is attempted using an alternative gateway destination.

Loadsharing

If a gateway becomes overloaded, the condition is detected by the process **rrgen** which outputs a signal to switch off one or more listening entries in the gateway's route-file **/etc/ROUTE-FILE.** This inhibits the gateways ability to accept new user connections. New connections to a disabled listening entry on a gateway will be rejected immediately causing the network device to try an alternate gateway destination.

If a gateway fails, the existing connections will hang, disconnect or time-out. New connections will time-out waiting for a response from the failed gateway and will then try to connect to the alternate host destination (another specified gateway) which should succeed. A connection made in this way will take a little longer to build.

Each gateway may have multiple listening entries, not all of which will be included in the scheme. Those listening entries not included in the scheme will remain on all the time and can be used by network devices not supporting 'alternate hosts'

Using the Listening Entries

The listening entries located in **/etc/ROUTE-FILE** are identified by the connect names used to make the user connections. For details on listening entries, refer to the *UNIX-Connect System Administration Guide.* The process **rrgen** has the ability to switch these entries on and off, and therefore to enable and disable connect names for a particular gateway.

In the example shown opposite (Figure 4-3), the process is configured in **rrgen.cfg** with four connect names **listen1** to **listen4.** Also four listening entries **listen1** to **listen4** are set-up in each gateway in the **ROUTE-FILE**.

The process has set the following default assignments:

connect names **listen1** and **listen2** **gway1**

connect names **listen3** and **listen4** **gway2**

The corresponding listening entries for these connect names are switched on the gateways.

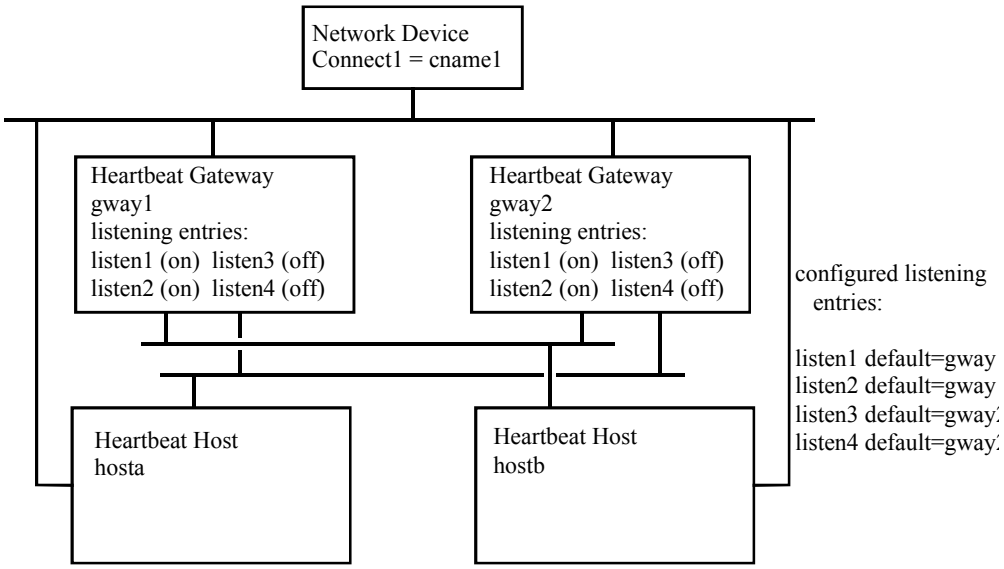


Figure 4-3. Example of OSI Heartbeat Configuration

The network device has two host entries which map to the **listen1** entries on **gways 1** and **2.** If **gway1** becomes overloaded, **rrgen** outputs a signal which switches **listen1** 'off' on **gway1** and 'on' on **gway2.** The attempt to establish a connection to **listen1** on **gway1** will fail. The alternate host addressing mechanism will try again, this time to connect to **listen1** in **gway2**. Hence, the load is re-distributed away from the overloaded gateway.

Appendix A
Heartbeat Routing Information

This appendix discusses the use of internal route names in a Heartbeat configuration, and the structure and setting up of database configuration information (DCI) for a Heartbeat system.

System and Internal Route Names

Heartbeat hosts and gateways are identified to users by their system names. However, the system name of a machine is not suitable for specifying the route to that machine as the internal LAN structure of a Heartbeat configuration ensures that there are always at least two different routes between machines. Refer to Figure A-1.

Each machine is connected to the Heartbeat LANs by three LAN controllers, each is identified by a different name. For example, in the Heartbeat configuration shown below in Figure A-1, **PHOENIX** is connected to a host LAN, user LAN and FailSafe LAN via the LAN controllers **hosta**, **phoenix** and **tla**, respectively The three controller names are used to specify the three routes to **PHOENIX**. So for example, the route from **HBGW1** or **HBGW2** to **PHOENIX** is specified by **hosta**.

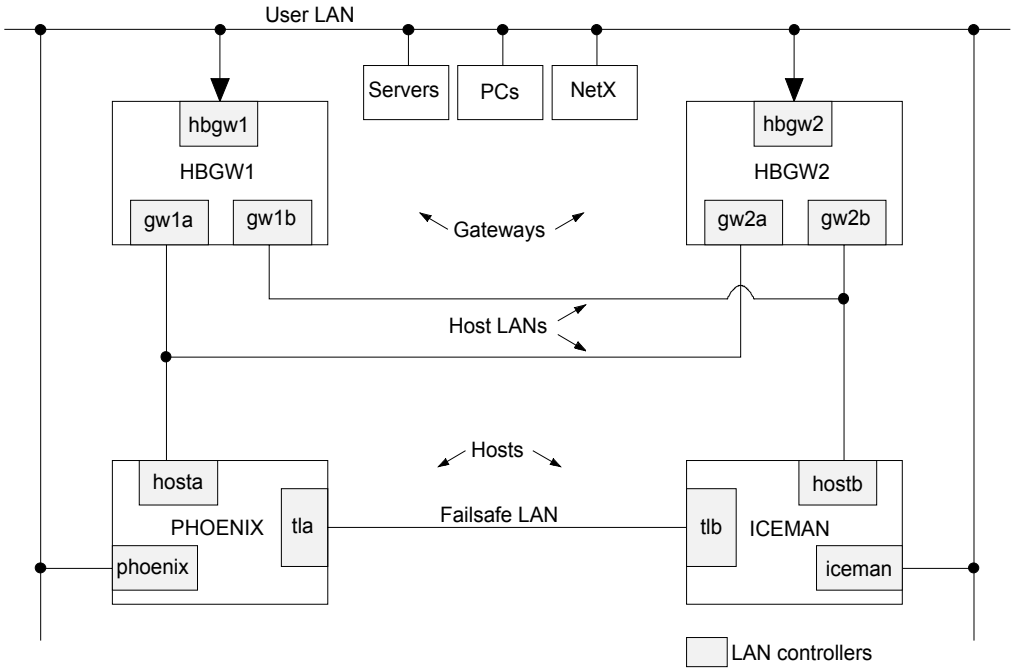


Figure A-1. Heartbeat System Showing Internal Routing

System names are used by the loadsharing software to connect users to the gateways and are entered in the Database Configuration Information.

LAN controller names are used by UNIX-Connect to build connections between Heartbeat gateways and hosts. ROUTE-FILE destination entries identified by the controller names on the remote system must be provided on the local system for each route it wishes to connect over.

Use of the Routes File

When given a system name, Heartbeat determines the appropriate route to connect over by performing a lookup in the **/etc/hb/routes** file. This file contains matrices which specify all the possible routes from the local system to the remote system. See Figure A-2. The **/etc/hb/routes** file must be configured as part of the Heartbeat set-up. See Chapter 3.

```
#   This file specifies the ROUTE-FILE names corresponding to each connection

# User LAN

HBLAN_UL        =           phoenix, iceman, hbgw1, hbgw2

HBCON_UL_phoenix=           phoenix, iceman, hbgw1, hbgw2
HBCON_UL_iceman =           phoenix, iceman, hbgw1, hbgw2
HBCON_UL_hbgw1  =           phoenix, iceman, hbgw1, hbgw2
HBCON_UL_hbgw2  =           phoenix, iceman, hbgw1, hbgw2

# TL LAN

HBLAN_TL        =           phoenix, iceman

HBCON_TL_phoenix=           tla,     tlb
HBCON_TL_iceman =           tla,     tlb

# Heartbeat LANs

HBLAN_H1        =           phoenix, hbgw1, hbgw2

HBCON_H1_phoenix=           hosta,   gw1a,  gw2a
HBCON_H1_hbgw1  =           hosta,   gw1a,  gw2a
HBCON_H1_hbgw2  =           hosta,   gw1a,  gw2a

HBLAN_H2        =           phoenix, hbgw1, hbgw2

HBCON_H2_iceman =           hostb,   gw1b,  gw2b
HBCON_H2_hbgw1  =           hostb,   gw1b,  gw2b
HBCON_H2_hbgw2  =           hostb,   gw1b,  gw2b
```

Figure A-2. Structure of /etc/hb/routes

DCI Structure

Database Configuration Information (DCI) held in **/etc/hb/dci.file** on each host and gateway contains a complete set of DCI data for the Heartbeat system. Gateways and hosts are identified by their system names.

For example, the DCI for the Heartbeat system configuration shown in Figure A-1 would be similar to the following:

```
GATE:0:hbgw1:M:0x0
GATE:0:hbgw2:G:0x0
HOST:0:phoenix:H:0x0
HOST:0:iceman:H:0x0
oasis:oasis1:phoenix:S:0x75500001
oasis:oasis2:iceman:P:0x75500002
cserver:phoenix:phoenix:S:0x0
cserver:iceman:iceman:P:0x0
```

This includes:

- The public system name and status of each host in the system (including the local host)
- The public system name and status of each gate in the system (and whether or not its the master)
- The ROUTE-FILE name and status of each application (including the name of the database that the application is on)
- The ROUTE-FILE name and status of hosts containing C Server program

The application and C server entries which are application-specific will change as applications are added or removed from being heartbeat protected.

Registering DCI

Entering the information for a particular host, gateway or application is referred to as 'registering' that item. This procedure is carried out using **hbmenu** at one of the host machines as described in Chapter 3.

Note: Heartbeat must be started on each of the Heartbeat hosts and gateways before you start the registration procedure.

The procedure is as follows:

Note: The order is important.

1. Configure the Hosts

   Register the other host machine in the Heartbeat system. This puts an entry in the local hosts **dci.file**, sets up an **hbclient-hbserver** link to the remote host and copies all DCI data to the remote hosts **dci.file**. This causes **hbclient** on the remote host to set up an client-server link back to the local host system.

2. Configure the Gateways

   Register each of the gateways. When you register a gateway, an entry for that gateway is placed in the local **dci.file** and copied across the client server link to the remote host. The host's **hbclient** also sets up a client server link to the gateway and passes all DCI data to the gateway's **dci.file.** This process is repeated for each gateway until all **dci.files** on each host and gateway contain a complete set of DCI entries for machines in the Heartbeat system.

3. Configure the applications

   Register the application names on your system. The application name is the user id that you enter on the gateways to connect to a database. You also enter the names of the databases associated with the application, their host machines and whether they have Primary, Secondary, Failed or Unknown status. Again this information is copied into the local **dci.file** and passed to all other host and gateway **dci.files.** (see Chapter 3).

Once this is complete the Heartbeat configuration is complete and ready to operate.

Appendix B
hbadm Utility

This appendix describes the purpose, syntax, options and restrictions on use of the **hbadm** utility.

hbadm

Purpose

Used to administer a Heartbeat configuration, including startup and shutdown.

Syntax

hbadm {*options*}

Options

-c (**close** or **stop)** Close the Heartbeat system tidily on the host machine.

con (config) *application* {**P**|**S**|**F**|**U**|**D**} *machine***]**
Change database configuration, where:

P = Primary
S = Secondary
F = Failed
U = Unknown
D = Delete

-f (**fault**) Fabricate a fault on the host (default is local host).

-h *host* Used with the **-f** option to generate a fault on a remote *host*.

-k (**kill)** Force close the Heartbeat system on the host machine.

> **CAUTION**
>
> Ensure that you understand the effect of the **monitor** option before using it. If you execute this option and failure occurs, the fault will be flagged, but Heartbeat users will remain connected to the failed system.

monitor [ON|OFF) Switches monitor mode on and off. Monitor mode enables Heartbeat software to monitor the FailSafe hosts while inhibiting the re-configuration scripts.

-q, (query or **status)** Displays the current state of the Heartbeat system on the host machine. Note that **query** and **status** are each keywords that can be used in place of the **-q** switch.

reg (**register**) *application database machine*
Register database.

res (**resolve**) *application machine*
Resolve database name.

	-s (**start**)	Starts the heartbeat system on the host machine.
	status	Show status of Heartbeat system on local host.
Parameter	*application*	Application to be registered or configured in **/etc/hb/dci.file**.
	database	Name of database to be registered in **/etc/hb/dci.file**.
	machine	Name of Heartbeat host containing database to be registered, configured or resolved.

Restrictions

Super-user permissions required.

Comments

hbadm start must be run on both FailSafe hosts and gateways to startup the complete Heartbeat configuration. This runs the parent **hbprocess** on the hosts and the **hbarbiter** process on the gateways.

hbadm with appropriate options is used in the **hbmenu** utility to start and stop Heartbeat, configure the database configuration information and generate a status display.

Appendix C
hbprocess Configuration File

This appendix shows the contents of the configuration file used by **hbprocess**, located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

```
#########################################################################
# Config file for hbprocess
#
#
# Set the rate at which ALIVE packets are sent to the arbiter (in milliseconds)
# NOTE all HB systems must have the same minor tick rate.
#
HB_MINOR_TICK=200
#
# Set the rate at which ALIVE messages are sent to hbprocess (in minor ticks)
#
HB_MAJOR_TICK=40
#
# Heartbeat monitor processes, (hbmonsys, hbmonreal, etc. ), must
# send regular alive beats to hbprocess, if too many are missed then # the
monitor is considered to have failed

# The following defines how many must be missed to generate a failure

HB_MISSED_BEATS=4

#
# Define how many beats an hb monitor should miss before warning
# messages are generated
#
HB_WARN_LEVEL=1

# The rate at which the individual monitors tick, (and thus send alive beats or
# report faults), is also configurable, tick values are in seconds
# Monitors can not be configured to tick slower than the expected
# beat rate used by hbprocess, (HB_MON_TICK = (HB_MINOR_TICK *
HB_MAJOR_TICK)/1000)

HBMONREAL_TICK=3
HBMONSYS_TICK=8

# If the following is non-zero then no messages are routed to /dev/console

HB_NO_CONSOLE=0

# Alternate locations for status files
```

```
# HB_DCI_FILE=files/dci.file
# HB_GATE_FILE=files/gate.file
# HB_USER_FILE=files/user.file

# Define the heartbeat message queue key (This is normally defaulted)
#
#HB_KEY=
#
# Set the level of message that are written to hblog
# Level 4 enables NOTICE messages (default)
# Level 7 enables DEBUG messages
#
HB_LOG_LEVEL=4
#
# Determine if heartbeat should shut down after remote system failed
#
HB_CLOSE_ON_REMOTE_FAILURE=0
#
# Define if heartbeat should start in monitor mode
# Set to 1 if it should (inhibits running of fail scripts)
#
HB_MONITOR=0
#
# Set the acknowledge timeout from the arbiter (in minor ticks)
#
HB_ACK_TIMEOUT=150
#
# Time out of gate load packets in seconds
#
HB_GATE_TO=90

# Set the priorities of various heartbeat daemons
#
# By default hbprocess and hbpulse run as realtime processes and the # monitors
run as normal time sharing processes.
#
# Setting HB_REALTIME_MON will also set the monitors to run as real
# time processes.
#
# If HB_REALTIME_PULSE is not set to 1 then hbprocess and hbpulse
# run with
# a nice level of -HB_NICE_VALUE  (This parameter is ignored if they # are run
as realtime processes)
```

```
#
HB_REALTIME_PULSE=1
HB_REALTIME_MON=0
#
HB_NICE_VALUE=20

# Set the following flag to 1 to lock hbpulse in memory to
# (hopefully)guarantee faster pulsing.
#
HB_LOCK_IN_MEMORY=1

# Rotate hb.log file to hb.log.old if it gets bigger than this size
#
HBLOG_LIMIT=1000000

# Define the maximum number of processes to be unstalled per second
#
HB_MAX_PER_SECOND=20

# Define the grace period (in seconds) on startup during which
# faults detected only cause heartbeat to shutdown - the system is
# NOT shutdown
#
HB_GRACE_PERIOD=60

# Define the user name allocated to heartbeat.  This user name must # exist
# and have the same password on all hosts.  /etc/USERS-FILE
# entries must be set up on the hosts so that root can connect
# to the other hosts using this user name.
#
# If this user name does not exist then the heartbeat daemons will
#  try to
# connect to the other hosts as root.
#
HB_USER_NAME=hbeat
                                          #
# Define the UNIX-CONNECT name for this machine
# if other than `uname -n`
#
#HB_LOCAL_NAME=
```

Appendix D
hbmonreal Configuration File

This appendix shows the contents of the configuration file used by **hbmonreal**, located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

```
# Configuration file for hbmonreal

# This is a sample file and may need modification on your system

# The sample settings given below have been found sufficient
# for the purposes of installation and integration testing of a
# heartbeat system, however the fault levels must be reviewed
# prior to going live on a particular system.

# Note that an incorrect level may cause an unexpected system
# shutdown


# RealityX system to be monitored

# To explicitly monitor a specific installation of RealityX enter
# the REALROOT path as follows :

# REALROOT              =       /usr/realman/4.0C

# If REALROOT is not specified in this file then the REALROOT
# environment variable will be used

# If REALROOT is not set in the environment then the LIVE entry in
# ~realman/installed will be used.


# Level at which all daemons are deemed to have faulted

# Messages are prioritized thus DEBUG INFO NOTICE WARNING ALERT PANIC

HB_FAULT_DEFAULT       =        ALERT

# Level at which specific daemons are deemed to have faulted

# Daemon names may be in upper or lower case

HB_FAULT_realcd         =       ALERT
HB_FAULT_realdd         =       ALERT
HB_FAULT_reality        =       ALERT
HB_FAULT_realcip        =       ALERT
HB_FAULT_rip            =       ALERT
```

```
HB_FAULT_rir            =       ALERT
HB_FAULT_fst            =       ALERT
HB_FAULT_tlrestore      =       ALERT
HB_FAULT_mkdbase        =       ALERT
HB_FAULT_rmdbase        =       ALERT
HB_FAULT_realdbck       =       ALERT
HB_FAULT_realformat     =       ALERT
```

```
# Specific messages to be ignored
```

Note: Messages to be ignored are defined as environment variables and specified as regular expressions. The environment variable name is made up of the following:

HB_IGNORE_*daemon_char*

where, *daemon* is the daemon process that generates the message and *char* is a unique identifier. Alpha characters are used to uniquely identify each variable name. When the alphabet becomes exhausted, other characters, such as numbers can be used.

```
HB_IGNORE_realcd     =   ".*ddaemon.*missed.*beats.*"
HB_IGNORE_realcd_A   =   ".*Abnormal exit from ddaemon.*"

HB_IGNORE_realdd     =   ".*DrsAttach: shmget: No such file or directory.*"
HB_IGNORE_realdd_A   =   ".*DflsClearLocks: Unlocking.*"

HB_IGNORE_fst        =   ".*Reject reason 8043.*"
HB_IGNORE_fst_A      =   ".*Reject reason 53023.*"
HB_IGNORE_fst_B      =   ".*Redual failed 54026.*"

HB_IGNORE_fsr        =   ".*Result 53023.*"
HB_IGNORE_fsr_A      =   ".*Result 53030.*"
HB_IGNORE_fsr_B      =   ".*Result 53029.*"

HB_IGNORE_rir        =   ".*child failed 8043.*"
HB_IGNORE_rir_A      =   ".*child failed 53023.*"
HB_IGNORE_rir_B      =   ".*child failed 53030.*"
HB_IGNORE_rir_C      =   ".*child failed 53029.*"

HB_IGNORE_rip        =   ".*Secondary connection lost.*"

HB_IGNORE_reality    =   ".*No buffer for received message.*"
```

Appendix E
hbmonsys Configuration File

This appendix shows the contents of the configuration file used by **hbmonsys**, located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

CAUTION

hbmonsys error messages may also contain user application error messages within their text string. Although the priority of the system error may be below HB_FAULT_DEFAULT, the user error message may be above, causing an unnecessary Heartbeat fault condition. To ensure that the application does not trigger an unwanted fault condition, ensure that the default fault level for a user error (HB_FAULT_USER) is set correctly.

```
# Configuration file for hbmonsys

# This is a sample file and may need modification on your system

# The sample settings given below have been found sufficient
# for the purposes of installation and integration testing of a
# heartbeat system, however the fault levels must be reviewed
# prior to going live on a particular system.

# Note that an incorrect level may cause an unexpected system
# shutdown


# Level at which all daemons are deemed to have faulted

# Messages are prioritized thus
#         DEBUG INFO NOTICE WARNING ERR CRIT ALERT EMERG

HB_FAULT_DEFAULT        =       ERR

# Level at which specific daemons are deemed to have faulted

# Daemon names can be in upper or lower case

HB_FAULT_AUTH           =       ALERT
HB_FAULT_KERN           =       ERR
HB_FAULT_daemon         =       CRIT
HB_FAULT_MAIL           =       NOTICE
HB_FAULT_USER            =      CRIT
```

```
# Specific messages to be ignored

HB_IGNORE_KERN          =       ".*Device write protected.*"
# HB_IGNORE_KERN        =       "/user2: file system full.*"
HB_IGNORE_DAEMON        =       "bootpd.*address not found.*"
```

Appendix F
hbarbiter Configuration File

This appendix shows the contents of the configuration file used by **hbarbiter,** located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

```
#########################################################################
#
# Config file for hbarbiter
#
#
# Define how many beats an hbwatch should miss before warning
# messages are generated
#
HB_WARN_LEVEL=1
#
# Define how many beats an hbwatch should miss before
# the HB system is considered dead.
#
HB_MISS_BEATS=20
#
# Set arbiter into fixed time interval mode
# (When set above variables are in units of 1 second)
#
HB_FIXED_TIMEOUT=1
#
#Define the heartbeat message queue key
#
#HB_KEY=
#
# Set the level of message that are written to hblog
# Level 4 enables NOTICE messages (default)
# Level 7 enables DEBUG messages
#
#HB_LOG_LEVEL=4
#
# Define the number of host systems
#
#HB_HOSTS=2
#
# Define the default time to live for gate load
#
#HB_GATE_LOAD_TTL=30
#
# Define the number of routers that equate to a gate load of 100 %
#
HB_MAX_ROUTERS=100
```

```
#
# Set the priorities of various heartbeat daemons
#
# By default hbarbiter and hbwatch run as realtime processes

# If HB_REALTIME_WATCH is set to 0 then hbarbiter and hbwatch run
# with a nice level of -HB_NICE_VALUE (This parameter  is ignored if #
they are run as realtime processes)
#
HB_RUNTIME_WATCH=1

HB_NICE_VALUE=20
#
# Rotate hb.log file to hb.log.old if it gets bigger than this size
#
HBLOG_LIMIT=1000000
#
# Define the user name allocated to heartbeat.  This user name must #
exist and have the same password on all gates.  /etc/USERS-FILE
# entries must be set up on the hosts so that root can connect to
# the gates using this user name
#
# If this user name does not exist then the heartbeat daemons will
# try to connect to the gates as root.
#
HB_USER_NAME=hbeat
#
# Define the Heartbeat name for this machine
# if other than 'uname -n'
#
# HB_LOCAL_NAME
```

Appendix G
hbadm Configuration File

This appendix shows the contents of the configuration file used by **hbadm**, located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

```
# Configuration file for hbadm

# This is a sample file and may need modification on your system
# Archive hb.log file during hbadm start if size exceeds limit

# HBLOG_LIMIT           =       5000000

# Suppress root check during hbadm start

# HBADM_NOROOTCHECK     =       1

# Log level cannot be set here

# HB_LOG_LEVEL =                Not used
```

Appendix H
hbpulse Configuration File

This appendix shows the contents of the configuration file used by **hbmondisk**, located in **$HBROOT/config**. Comments defining the purpose of each environment variable precedes the defined variable. The settings shown here are the defaults.

```
# This is a sample file and may need modification on your system

# List of devices to be monitored.  This may include wild cards eg
# HBMONDISK_LIST = /dev/rdsk/c[123]d*s3 /dev/rdsk/c0d?s1

HBMONDISK_LIST =

# Set the sample rate in seconds
# This cannot be slower than the expected beat rate used by
  hbprocess

HBMONDISK_TICK = 4



# Set the level of logging messages

HB_LOG_LEVEL = 4
```

Appendix I
hbuser Script and HBPROC

This appendix contains examples of the **hbuser** script and HBPROC.

hbuser Script

```
banner hbuser

#    #    #####    #    #     ####    ######    #####
#    #    #    #   #    #    #        #         #    #
######    #####    #    #     ####    ######    #    #
#    #    #    #   #    #         #   #         #####
#    #    #    #   #    #    #    #   #         #   #
#    #    #####     ####      ####    ######    #    #

# cat hbuser
#########################################################################
#
# Database reconfiguration script (hbuser)
#
# Called by: hbuser <application> <databasename> <databasepath>
#
# This script is run when a database is made the primary after the
# other system has failed.  The script is run before stalled users
# are allowed on.  The script is only run once per database and
# hence should handle all applications in one go.
#
# The user should modify this script as necessary
#

DbaseOwner()
{
    ls -ld £1 | sed 's/ */     /g' | cut -f3
}

Owner='DbaseOnwer £3'

# Switch XUI circuit PNC
£HBROOT/bin/prim_ann BASE &

# Run database HBPROC
reality -d $2 -c HBPROC -u HBPROC -a SYSPROG &
#
```

HBPROC

```
      HBPROC
001   PQN
002   C
003   C This PROC gets run on a Heartbeat Switchover, to initialise
004   C any user or application specific stuff.
005   C
006     HSYS echo "HBPROC: HBPROC started" > /dev/console
007     P
008   C
009   C Reset the Spooler Queues
010     HSYS echo "HBPROC: Restarting despoolers" > /dev/console
011     P
012     OResetting Spooler Queues
013     HRESET.DESPOOLER (D, S
014     P
015     HSYS echo "HBPROC: Despoolers reset" > /dev/console
016     P
017   C
018   C Done !!!!
019   C
020     HSYS echo "HBPROC: HBPROC completed" > /dev/console
021     P
022     RTN
```

/

A

C

D

E

F

G

H

L

M

O

P

R

S

T

U