Ignite-UX Reference
for HP-UX 11i

HP Part Number: 5992-6587
Published: September 2009
Edition: 17

About This Document

This document includes the latest manpages for the Ignite-UX product.

Intended Audience

This document is intended for system and network administrators responsible for installing, configuring, and managing HP-UX servers and workstations. Administrators are assumed to have an in-depth knowledge of HP-UX operating system concepts, commands, and configuration. It assumes familiarity with installing HP computer hardware and software, upgrading software, applying patches, and troubleshooting system problems.

Additionally, administrators are expected to have knowledge of Transmission Control Protocol/Internet Protocol (TCP/IP) networking concepts and network configuration. This reference guide is not an administration guide, a TCP/IP guide, or an Ignite-UX tutorial.

Related Documents

The most current edition of the following documents are found at the HP Technical Documentation website at http://www.docs.hp.com/.

- *Ignite-UX Administration Guide*
- *Ignite-UX Quick Start Guide*
- *Ignite-UX Custom Configuration Files*
- *Successful System Cloning using Ignite-UX* White Paper
- *Successful System Recovery using Ignite-UX* White Paper
- *Installing and Updating Ignite-UX* White Paper
- *Ignite-UX Installation Booting* White Paper
- *Read Before Installing or Updating to HP-UX*
- *HP-UX Installation and Update Guide*
- *HP-UX Reference*
- *HP-UX System Administrator's Guide*
- *Managing Systems and Workgroups: A Guide for HP-UX System Administrators*
- *Software Distributor Administration Guide*
- *HP-UX Patch Management*
- *nPartition Administrator's Guide*
- *Getting Started with Software Package Builder*
- *VERITAS File System 4.1 (HP OnlineJFS/JFS) and VERITAS Volume Manager 4.1 Installation Guide*

Some or all of these documents are available on the Instant Information media and in printed form.

Publishing History

The document printing date and part number indicate the document's current edition. The printing date will change when a new edition is printed. Minor changes may be made at reprint without changing the printing date. The document part number will change when extensive changes are made. Document updates may be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, you should subscribe to the appropriate product support service. See your HP sales representative for details. You can find the latest version of this document online at:

http://www.docs.hp.com/en/oshpux11iv3.html#Ignite-UX.

Manufacturing Part Number	Supported Operating Systems	Edition Number	Publication Date
5992–6587	HP-UX 11i v1, 11i v2, 11i v3	17	September 2009
5992–4740	HP-UX 11i v1, 11i v2, 11i v3	16	September 2008
5992–3346	HP-UX 11i v1, 11i v2, 11i v3	15	March 2008
5992–3345	HP-UX 11i v1, 11i v2, 11i v3	14	December 2007
5992–1958	HP-UX 11i v1, 11i v2, 11i v3	13	September 2007
5992–0611	HP-UX 11.00, 11i v1, 11i v2, 11i v3	12	June 2007
B2355–91066	HP-UX 11.00, 11i v1, 11i v2, 11i v3	11	February 2007
5991–7360	HP-UX 11.00, 11i v1, 11i v2	10	December 2006
5991–5522	HP-UX 11.00, 11i v1, 11i v2	9	September 2006
5187–4555	HP-UX 11.00, 11i v1, 11i v2	8	June 2006

NAME

add_new_client − add a client to an Ignite-UX server

SYNOPSIS

/opt/ignite/bin/add_new_client -s *Ignite-UX_Server* [**-R** *release_name*] [**-f**] [**-d**] [**-?**]

DESCRIPTION

add_new_client is used to construct a client directory on an Ignite-UX server without requiring the client to be booted from the Ignite-UX server first.

The **add_new_client** command must be executed on the client and requires privileged user permission. (See the *privileges*(5) manpage for more information on the privileged user.) The **/var/opt/ignite/clients** directory must be exported from the Ignite-UX server with the appropriate permissions so that the client may mount the directory using NFS. For help exporting the client directory from the server, see the **Exporting the Client Directory** section.

add_new_client mounts **/var/opt/ignite/clients** from the Ignite-UX server. It then performs a series of commands which result in the construction of required information files under the client directory (**/var/opt/ignite/clients/0x***LLA*). These files are primarily **io.info**, **hw.info**, **host.info**, and **config.sys** along with several other files, which are used by the Ignite-UX server to recognize and manage the client. Once **add_new_client** has successfully executed, *ignite*(5) may be invoked on the Ignite-UX server and used to perform additional actions on the client.

If you do not have this command installed on the client, here are two possible ways to obtain it. The first is to simply **rcp** or **ftp** the **add_new_client** command directly to the client. If you elect to use this method be sure to note the **DEPENDENCIES** section of the manpage and pull the additional supporting commands too.

The second method is to use **swinstall** to install the required filesets onto the the client. For example:

swinstall -s *source_depot* **Ignite-UX.RECOVERY Ignite-UX.MGMT-TOOLS**

Options

add_new_client recognizes the following options:

-s *Ignite-UX Server*
: Specifies the host name of the Ignite-UX server on which the client directory is constructed.

-R *release_name*
: Specifies the release that you intend to install on the client. By default the value of **uname -r** is used as the target OS release. This option is also useful when the Ignite-UX server does not support the OS release that the client is currently running.

-f
: Instructs **add_new_client** to overwrite existing client information files found in the **/var/opt/ignite/clients/0x***LLA* directory. This option is intended to be used to refresh the contents of the client directory. The previous contents are not preserved.

-d
: Instructs **add_new_client** to use the fully qualified host name of the client for the *client_name*.

-?
: Display the help screen.

Exporting the Client Directory

On the server, the client's directory may be exported as follows:

For 11.31 or later, edit **/etc/dfs/dfstab** to add following entry:

```
share -F nfs -o anon=2 /var/opt/ignite/clients
```

For other releases, edit **/etc/exports** to add the following entry:

```
/var/opt/ignite/clients -anon=2
```

For 11.31 or later, run the **shareall** command to have the edits to the exports file take effect:

/usr/sbin/shareall -F nfs

For other releases, run the **exportfs** command to have the edits to the exports file take effect:

/usr/sbin/exportfs -av

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

add_new_client will exit with a status of 0 (zero) if the client was able to successfully initialize its corresponding client directory on the Ignite-UX server. A non-zero value is returned if any problem occurs.

EXAMPLES

Initialize the client using the specified Ignite-UX server:

add_new_client -s *Ignite-UX_Server*

Initialize the client and overwrite existing contents of the client directory on the Ignite-UX server:

add_new_client -s *Ignite-UX_Server* **-f**

Initialize the client using the specified Ignite-UX server, for target OS release **B.11.23***:*

add_new_client -s *Ignite-UX_Server* **-R** *B.11.23*

FILES

/opt/ignite/boot/Rel_*release***/INSTALLFS**
/opt/ignite/boot/Rel_*release***/VINSTALLFS**
/opt/ignite/boot/Rel_*release***/WINSTALLFS**
/opt/ignite/boot/Rel_*release***/IINSTALLFS**

PA-RISC 32-bit, PA-RISC 64-bit (V class), PA-RISC 64-bit (non V class), and Itanium®-based file systems used by install clients. Configuration information available at client boot-time is stored in the first 8KB of these files. These are the default files **instl_adm** modifies.

/var/opt/ignite/clients/0x*LLA***/host.info**

File that contains the hardware model and system name of the client.

/var/opt/ignite/clients/0x*LLA***/hw.info**

File containing hardware information about the client such as any information about the hard drives, tape drives, LAN devices, etc. This file will be deprecated in a future release and is being replaced by the **io.info** file.

/var/opt/ignite/clients/0x*LLA***/io.info**

File containing hardware information about the client such as any information about the hard drives, tape drives, LAN devices, etc.

/var/opt/ignite/clients/0x*LLA***/config.sys**

File that contains the system information about the clients such as IP address, system name, etc.

/var/opt/ignite/clients/0x*LLA***/client_name**

File that contains the name of the client.

/var/opt/ignite/clients/0x*LLA***/client_status**

File containing information about the status of each of the client addition steps.

DEPENDENCIES

lanscan(1M)

/opt/ignite/lbin/loadfile

/opt/ignite/lbin/rescan_hw_host

SEE ALSO

exportfs(1M), exports(4), ignite(5), nfs(7).

NAME

ansitape − ANSI-standard magtape label program

SYNOPSIS

ansitape txrcpuA[**vqfaei3**] [**mt=***device*] [**vo=***volume-name*] [**wo=***output_file*] [**rs=**[**r**|*recordsize*]] [**bs=***blocksize*] [**rf=**[**v**|**f**]] [**cc=**[**i**|**f**|**e**]] *filename1 filename2...*

DESCRIPTION

Ansitape reads, writes, and creates magtapes conforming to the ANSI standard for magtape labelling. Primarily, this is useful to exchange tapes with VAX/VMS, which makes this kind of tape by default.

Ansitape is controlled by a function key letter (**t**, **x**, **c**, or **r**). Various options modify the format of the output tape.

Writing ANSI Tapes

The list of files on the command line is written to the tape. A full Unix pathname may be specified, however, only the last pathname component (everything after the last **/**) is used as the filename on the tape.

Normally, regular text files are to be exchanged. **ansitape** reads the files one line at a time and transfers them to the tape. The newline character at the end of each line is removed, and the file is written in a variable-length record format. Variable-format files have the length of the longest record specified in a file header. Therefore, **ansitape** will read each input file from disk before it goes on to tape, to determine the maximum record size. The read is skipped if the file is more than 100,000 bytes long. The default carriage control (implied) instructs the other host to restore the newline character before printing the record.

If **ansitape** thinks that the input file is a Unix text file (Fortran or implied carriage control), it will automatically strip the the Unix newline from the end of each record. No strip is done with embedded carriage control files, or with any file using a fixed-length record format.

For binary files, fixed-length records should be used. VAX/VMS normally uses a record length of 512 bytes for things like directories and executable files, but data files may have any record length. Binary files should be flagged for embedded (*rf=e*) carriage control.

Reading ANSI Tapes

When reading, the input file list is presumed to be the names of files to be extracted from the tape. The shell wildcard characters asterisk (*****) **and question-mark (?) may be used. Of course, they must be quoted to prevent the shell from interpreting them before ansitape** sees them.

None of the options for record format or carriage control need be specified when reading files. **Ansitape** will automatically pick up this information from the header records on the tape, and do the right thing. If you can't get just what you want from **ansitape**, the resulting files may be run through *dd*(1).

FUNCTION LETTERS

These function letters describe the overall operation desired. One of them must be specified in the first argument to **ansitape**. For lexically rigorous Unix fans, a minus sign (**-**) **is allowed, but optional, to introduce the first keyword option set.**

r Write the named files on the end of the tape. This requires that the tape have been previously initialized with an ANSI volume header.

c Create a new magtape. The tape is initialized with a new ANSI volume header. All files previously on the tape are destroyed. This option implies **r**.

p This option is used with option c or r to create ANSI labels for Ignite-UX recovery tape only. It requires a non rewind tape device file.

x Extract all files from the tape. Files are placed in the current directory. Protection is r/w to everyone, modified by the current *umask*(2).

t List all of the names on the tape.

u This option is used to write an ANSI label UHL3EFIAUX header for an Ignite-UX recovery tape.

A This option is used to create an Ignite-UX recovery tape archive label.

MODIFIER KEY LETTERS

These key letters are part of the first argument to **ansitape**.

v Normally **ansitape** does its work silently; the **v** (verbose) option displays the name of each file **ansitape** treats, preceded by the function letter. It also displays the volume name of each tape as it is mounted. When used with the **t** option, **ansitape** displays the number of tape blocks used by each file, the record format, and the carriage control option.

q Query before writing anything. On write (**c or r** options), this causes **ansitape** to ask before writing to the tape. On extract operations, **ansitape** displays the Unix pathname, and asks if it should extract the file. Any response starting with a **y** or **Y** means yes, any other response (including an empty line) means no.

f File I/O is done to standard i/o instead. For example, when writing a tape file that is to contain a lint listing, we could specify

```
lint xyz.c | ansitape rf xyz.lint
```

instead of

```
lint xyz.c > /tmp/xyz.lint
ansitape r /tmp/xyz.lint
rm /tmp/xyz.lint
```

When reading, this option causes the extracted files to be sent to stdout instead of a disk file.

a The tape should be read or written with the ASCII character set. This is the default.

e The tape should be written with the EBCDIC character set. The mapping is the same one used by the *dd*(1) program with **conv=ebcdic**. This option is automatically enabled if IBM-format labels are selected.

i Use IBM-format tape labels. The IBM format is very similar, but not identical, to the ANSI standard. The major difference is that the tape will contain no HDR3 or HDR4 records, thus restricting the name of the files on the tape to 17 characters. This option automatically selects the EBCDIC character set for output. To make an IBM-format label on a tape using the ASCII character set (why?), use the option sequence **ia**.

3 Do not write HDR3 or HDR4 labels. The HDR3 label is reserved for the use of the operating system that created the file. HDR4 is for overflow of filenames that are longer than the 17 characters allocated in the HDR1 label. Not all systems process these labels correctly, or even ignore them correctly. This switch suppresses the HDR3 and HDR4 labels when the tape is to be transfered to a system that would choke on them.

FUNCTION MODIFIERS

Each of these options should be given as a separate argument to **ansitape**. Multiple options may be specified. They must appear as after the key-letter options above, and before any filename arguments.

mt=*device* Select an alternate drive on which the tape is mounted. The default is **/dev/rmt8**.

vo=*volume-name*

Specify the name of the output volume. Normally, this defaults to the first six characters of your login name. The string 'UNIX' is used as the default if **ansitape** cannot determine your login name.

wo=*output_file*

Specifiy the file name which stores the volume label information. With this option, user can get volume label information easily.

rs=*recordsize*

Specify the output recordsize in bytes. This is the maximum size in the case of variable-format files. This option also turns on the fixed-record-format option. Thus, if you want to

have variable record sizes with a smaller maximum, you must specify:

rs=*recordsize* **rf=v**

When the *recordsize* is manually given, **ansitape** does not read disk files to determine the maximum record length.

rs=r This is a variant of the **rs=** option. This causes **ansitape** to read all disk files for recordsize, regardless of their size. Normally, files larger than 100K bytes are not scanned for recordsize. Using this option also implies variable-length records.

bs=*blocksize* Specify the output *blocksize*, in bytes. As many records as will fit are crammed into each physical tape block. ANSI standards limit this to 2048 bytes (the default), but you may specify more or less. Be advised that specifying more may prevent some systems from reading the tape.

rf=v Record format is variable-length. In other words, they are text files. This is the default, and should be left alone unless you really know what you're doing.

rf=f Record format is fixed-length. This is usually a bad choice, and should be reserved for binary files. This also turns off the newline strip usually done for Unix text files.

cc=i Carriage control implied (default). Unlike Unix text files, where records are delimited by a newline character, ANSI files do not normally include the newline as part of the record. Instead, a newline is automatically added to the record whenever it is sent to a printing device.

cc=f Carriage control Fortran. Each line is expected to start with a Fortran carriage-control character. **Ansitape** does not insert these characters automatically, it merely marks the file as having them. This is of limited usefulness. (Good opportunity for another ambitious hacker.)

cc=e Carriage control is embedded. Carriage control characters (if any) are a part of the data records. This is usually used in the case of binary data files.

AUTHOR

David S. Hayes, Site Manager, US Army Artificial Intelligence Center. Originally developed June 1986. Revised August 1986. This software is in the public domain.

FILES

/dev/rmt?	half-inch magnetic tape interface
/dev/rar?	quarter-inch magnetic tape interface
/dev/rst?	SCSI tape interface

SEE ALSO

dd(1), umask(2), mtio(4), tp(5).

BUGS

The **r** (write) option cannot be used with quarter-inch archive tapes, since these tape drives cannot backspace.

There is no way to ask for the *n*-th occurrence of a file.

Tape errors are handled ungracefully.

Files with names longer than 80 characters have the name truncated. This is a limitation of the ANSI labelling standard. If the tape is made without HDR3 and HDR4 labels (**3 or i** switch), the name is limited to 17 characters.

Multi-volume tape sets cannot yet be generated. **ansitape** will read them just fine, but it won't write them. Unix provides no device-independent way to detect a physical end-of-tape. It was decided that a 2400-foot limitation was preferrable to device-dependence.

Note to Systems Programmers

ansitape uses a boolean function (**eot**) **to determine when the tape drive has hit the end of file. It is called every time a block of data is written to the tape. If this function ever returns TRUE (a defined constant), an automatic volume switch occurs. The pertinent device registers are obtained by a MTIOCGET ioctl system call. The registers are described in /sys/sundev/tmreg.h** (Sun system with TapeMaster controller). If you have a VAX, the filename will be slightly different. Sun Microsystems supplies this file even with binary-only distributions.

NAME

archive_impact − calculate file system impacts for tar, cpio, and pax archives

SYNOPSIS

/opt/ignite/lbin/archive_impact -t [**-g**|**-z**] [**-l** *level*] [**-b** *block_size*] [**-f** *fragment_size*] *tar_archive* [**-?**]

/opt/ignite/lbin/archive_impact -c [**-g**|**-z**] [**-l** *level*] [**-b** *block_size*] [**-f** *fragment_size*] *cpio_archive* [**-?**]

/opt/ignite/lbin/archive_impact -p [**-g**|**-z**] [**-l** *level*] [**-b** *block_size*] [**-f** *fragment_size*] *pax_archive* [**-?**]

DESCRIPTION

archive_impact is run to calculate the per file system disk space **impacts** for a given **cpio, pax,** or **tar** archive. **archive_impact** writes its results as **impacts** lines in Ignite-UX configuration file format to **stdout**. Results may be redirected to an Ignite-UX configuration file and then included in a **sw_sel** statement. See *instl_adm*(4) for details on the syntax of the Ignite-UX configuration file.

Note that any archives that are produced using relative path names are extracted relative to the "/" directory. When **archive_impact** encounters a relative path name, it maps it relative to "/".

Options

archive_impact recognizes the following options:

-t The archive being processed is in **tar** format. Either this option, the **-c** option or the **-p** option must be specified. **tar** format is the default for **make_sys_image** archives.

-c The archive being processed is in **cpio** format. Either this option, the **-t** option or the **-p** option must be specified.

-p The archive being processed is in **pax** format. Either this option, the **-t** option or the **-c** option must be specified.

-g The archive has been compressed using **gzip**.

-z The archive has been compressed using **compress**.

-l *level* Determines for what *level* in the file system hierarchy to build **impacts** statements. For example, if *level* is **1, impacts** statements would be generated for **/usr** and **/var**. If *level* is **2**, **impacts** statements would be generated for **/usr/local**, **/usr/bin**, etc. It is not recommended to specify a *level* greater than **2** for performance reasons. The default is **1**.

-b *block_size* Set the default file system block size to use for size calculations. The default is 8192.

-f *fragment_size*
Set the default file system fragment size to use for size calculations. The default is 1024.

-? Display the help screen.

Note: The **-b** and **-f** options are left for compatibility only. They are deprecated as of Ignite-UX version C.6.0.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Generate file system **impacts** for the **gzip** compressed **tar** archive named *mysys.tar.gz*:

archive_impact -t -g *mysys.tar.gz*

Generate file system **impacts** for the compressed **cpio** archive named *othersys.cpio.Z*:

archive_impact -c -z othersys.cpio.Z

SEE ALSO

pax(1), make_sys_image(1M), instl_adm(4), ignite(5).

NAME

auto_adm − manage LIF AUTO configuration files

SYNOPSIS

/opt/ignite/bin/auto_adm [**-p** *platform*] [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-n -T** *timeout* **-M** *message* **-l** *label* **-c** *command* **-b** *device* **-i** *image* [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-a -l** *label* **-c** *command* **-b** *device* **-i** *image* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-A** *appendfile* -f *infile* [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-L** *label record_modifier* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-d -L** *label* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-D -L** *label* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-T** *timeout* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

/opt/ignite/bin/auto_adm [**-p** *platform*] **-M** *message* [**-f** *infile*] [**-o** *outfile*] [**-O** *output_format*] [**-?**]

DESCRIPTION

The **auto_adm** command is used to manipulate LIF autoconfiguration files.

auto_adm manages files in two different formats, **auto_conf** (**CONF**) **and Initial System Loader** (**ISL**) **format. See** *auto_adm*(4) for a description of these formats. **auto_adm** may read and write either format, and convert between formats. **auto_adm** defaults to PA architecture unless IA is specified with **-p** *platform*.

record_modifier consists of at least one (and optionally any combination) of **-l** *label*, **-c** *command*, **-b** *device*, and/or **-i** *image*.

Options

auto_adm recognizes the following options:

-p *platform*
: Specifies the hardware platform of either **IA** or **PA**. If the platform is **IA**, no LIF volume manipulations will apply. If omitted, platform is assumed to be **PA**.

-f *infile*
: Specifies the input file name. If file name is a LIF volume, an ISL-formatted file entitled **AUTO** is read from that volume. If **-f** is omitted, data is read from standard input. **auto_adm** uses the following rules to determine the input format:

  - If file name is a LIF volume, or the file begins with the characters **hpux**, the input format is assumed to be in **ISL** format.
  - Otherwise it is assumed to be in **CONF** format.

-o *outfile*
: Specifies the output file name. If *outfile* is a LIF volume, a file entitled **AUTO** is written to that volume. If omitted, data is written to stdout.

-O *output_format*
: Specifies the format of *outfile*, either **CONF** or **ISL**. The default is **CONF**. If the file specified by **-o** is a LIF volume, the output format must not be **CONF**.

-n Create a new **AUTO** or **auto_conf** file, and initialize it with data specified by the **-T**, **-M**, **-l**, **-c**, **-b** and **-i** options.

-T *timeout*
Specifies the timeout in seconds before the default action is taken. Zero (0) means wait indefinitely.

-M *message*
Specifies the kernel boot prompt message.

-l *label*
Specifies the label to write into new or modified records.

-c *command*
Indicates the **ISL** command to run when a record is executed.

-b *device*
Indicates the boot device containing the boot image. Boot devices apply to platform **PA** only and are specified in the form **(;0)**. **auto_adm** expects the parentheses to be present, and care must be used to prevent the shell from removing them. If **auto_adm** issues an error message about missing parentheses characters but you have included them, please see the documentation for your shell to determine its method of specifying arguments containing special characters.

-i *image*
The name of the image on the boot device.

-a Add a record to an existing file using data specified by the **-l**, **-c**, **-b** and **-i**.

-A *appendfile*
Add a record contained in *appendfile* to an existing file using data read from the file specified by **-f** infile.

-L *label*
Specifies the label to search for when modifying records.

-d Delete the record specified by **-L**.

-D In conjunction with -L, sets the default boot action to take upon expiration of the timeout period.

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

For more information about the formats of files used by **auto_adm**, see *auto_adm*(4). In the following examples, the CONF file is *conf1*:

```
timeout = 0                          # wait indefinitely
message = Choose version to install  # prompt to user
default = 1                          # choose 32-bit by default

label   = B.11.11 32bit install
bootcmd = boot
boot    = (;0)
image   = boot/Rel_B.11.11/INSTALL   # path to 32-bit install kernel
```

Read the input file *conf1* and write it to the terminal in ISL format:

auto_adm -f *conf1* **-O ISL**

This produces the following output:

```
hpux KernelPrompt "Choose version to install" 0 1
reset
"B.11.11 32bit install" boot (;0)boot/Rel_B.11.11/INSTALL
"Exit" reboot
```

Append the contents of the CONF file *conf1* to a similar CONF file *conf2*, and write the results to the terminal:

auto_adm -A *conf1* **-f** *conf2*

Convert the conf file *conf1* to ISL format, and write it to file *islout*:

auto_adm -f *conf1* **-O ISL -o** *islout*

Read the file *islout*, change the timeout to *10* seconds, and write the results to the terminal:

auto_adm -f *islout* **-T** *10*

Read the file *islout*, delete the record with the label *Install HP-UX 11.31*, and write the resulting file to the terminal:

auto_adm -f *islout* **-d -L "***Install HP-UX 11.31***"**

To get a list of labels in the file *boot_lif* (note: this is for PA):

auto_adm -f *boot_lif* **-O CONF -p PA | grep label**

To set the default boot action for a PA-RISC client to take upon expiration of the timeout period in the file *boot_lif* with timeout *10* and label "*target OS is B.11.31 PA*":

auto_adm -f *boot_lif* **-T** *10* **-L "***target OS is B.11.31 PA***" -D**

To get a list of labels in the file *AUTO* (note: this is for IA):

auto_adm -f *AUTO* **-O CONF -p IA | grep label**

To set the default boot action for a *IA* client to take upon expiration of the timeout period in the file *AUTO* with timeout *10* and label "*target OS is B.11.31 IA*":

auto_adm -p *IA* **-f** *AUTO* **-T** *10* **-L "***target OS is B.11.31 IA***" -D**

FILES

/opt/ignite/boot/boot_lif
: Used for the booting of PA clients. This file is a LIF image that contains the file "AUTO". To see the contents of this file run the command "lifls -l /opt/ignite/boot/boot_lif".

/opt/ignite/boot/AUTO
: Used for the booting of IA clients. To see the contents of this file run the command "cat /opt/ignite/boot/AUTO".

RETURN VALUES

auto_adm returns the following values:

0 Normal exit.

1 Request to modify LIF volume failed.

2 Requested LIF volume not found.

3 File was not a LIF volume.

4 Requested auto_conf file was not found.

5 Syntax error parsing file.

6 Could not open input file.

7 Could not open output file.

8 Requested record not found.

9 Internal Error.

10 File locked.

AUTHOR

auto_adm was written by Hewlett-Packard Company.

SEE ALSO

auto_adm(4), lif(4).

NAME

bootsys − reboot and install clients using Ignite-UX

SYNOPSIS

/opt/ignite/bin/bootsys [**-a**|**-w**] [**-dvhrfS**] [**-i** *configuration*
|**-R** *target_operating_system_release*] [[**-l** *sw-sel=true|false*]...]
[**-V** *var[=value*]...] [**-s** *grace*] [**-t** *Ignite-UX_server*]
[**-g** *gateway*] [**-m** *netmask*] *[client1[:IP_address*]] [*client2*[*:IP_address*]...] [**-?**]

/opt/ignite/bin/bootsys -c *Ignite-UX_Server* [**-a**|**-w**] [**-dvhrfS**]
[**-i** *configuration* | **-R** *target_operating_system_release*]
[**-l** *sw-sel=true|false*]...] [**-V** *var[=value*]...] [**-s** *grace*]
[**-g** *gateway*] [**-m** *netmask*] [**-?**]

DESCRIPTION

bootsys is used to remotely initiate HP-UX operating system installation on one or more clients without the need to interact with the console of each client. The requirements are that each client must be running HP-UX version B.10.20 or later, be accessible over the network, and have enough space in **/stand** to hold the Ignite-UX kernel and RAM file system for the desired target release of HP-UX. The Ignite-UX kernels (***INSTALL**) and RAM file systems (***INSTALLFS**) are located on the Ignite-UX server in **/opt/ignite/boot/Rel_***release*.

bootsys copies an Ignite-UX kernel and RAM file system to each client and then sets the client's **AUTO** file, so that it will boot from the Ignite-UX kernel at the next system reboot. **bootsys** also modifies stable storage (also known as nonvolatile memory) using the **setboot** command to set the primary boot path and auto-boot flag to the disk from which it is currently booted. If you want to boot back to HP-UX from the disk that bootsys has modified and the **-r** option of **bootsys** was used, then use **/tmp/bootsys_prep -u** on the client. If **-r** was not used, reboot the client, interact with the **ISL**, and manually select **/stand/vmunix** to boot, and the changes will be undone. The command **/tmp/bootsys_prep** is left on the client by the **bootsys** command and is removed when its **-u** option is used.

The **bootsys** command may be executed from an Ignite-UX server, or by using the **-c** option on the client to be installed. The user running **bootsys** must have privileged user permission on each of the targeted clients. (See the *privileges*(5) manpage for more information on the privileged user.) If the user does not have **remsh** access to a client, **bootsys** prompts for the privileged user password (unless the **-d** option is used, which causes the client to be skipped).

The target operating system release is required data for the **bootsys** command to transfer the correct Ignite-UX kernel and RAM file system. The target operating system release can be explicitly indicated with the **-R** option. If the **-i** option is used, **bootsys** will determine the operating system release that is supported by that configuration and use it. If neither the **-R** option nor the **-i** option are given, bootsys will set the target operating system to match the operating system release currently on the client. If the client has historical configuration information on the Ignite-UX server and the current target operating system release differs from the historical configuration, the **-f** option will be required for **bootsys** to successfully complete.

The **-a**, **-w**, and **-i** options cause **bootsys** to add information to any existing parameters specified in the **/opt/ignite/boot/Rel_***release***/*INSTALLFS** file. These parameters are added during the transfer of this file to the client. The original file on the server is not modified.

Client systems may block the use of the **bootsys** command with the existence of the **/.bootsys_block** file in the client's root directory. See the section, **Blocking bootsys Attempts**.

Options

bootsys recognizes the following options:

-a Automatically start the install process with no interaction. This means that the client will not wait for the user to specify any configuration parameters and it uses all the defaults during installation.

-w Causes the client system to boot Ignite-UX and then wait for the **ignite** user interface running on the Ignite-UX server to instruct the client on how to proceed with the installation.

If neither the **-a** nor the **-w** option is given, the client will boot using the default mode stored in the **/opt/ignite/boot/Rel_***release***/*INSTALLFS** file. The keywords that control executing the user interface are **control_from_server** and **run_ui**. See *instl_adm*(4) for more information on configuration keywords.

-d Causes **bootsys** to ignore any clients listed that are down, cannot be accessed with **remsh**, or are incompatible with the indicated target operating system. **bootsys** will continue operating on systems that can be accessed and are compatible with the target operating system. Without the **-d** option, **bootsys** will prompt for the root password of clients that do not allow **remsh** access, and will exit if any client appears to be down.

The **-d** option should be used when executing **bootsys** from a **crontab** entry so that it will prevent **bootsys** from trying to prompt for passwords.

The user must take appropriate actions, such as configuring a **.rhosts** file or setting up SSH keys before running **bootsys** to ensure automatic execution can successfully occur using this option.

-v Causes **bootsys** to give verbose messages.

-h Prevents **bootsys** from inserting the client's host name and IP address into the configuration information copied to the client. This may be useful if you would rather have the client obtain this information using the **DHCP** or **BOOTP** protocol when it boots. You must have a **DHCP** server, or a server with an **/etc/bootptab** entry for the client, if you use this option or an interactive menu appears on the client on the console to obtain networking information.

-r Causes **bootsys** to prepare each client for install but suppresses rebooting the system. This may be useful if you want to use a different method to reboot the system, or wish to check the files copied to the client before it reboots.

-f Causes **bootsys** to remove all the client's prior configuration history before rebooting the client. This ensures that it uses all the default parameters, and that either the **-i** or **-R** option takes effect preventing past configuration information from being used. This option causes the files **/var/opt/ignite/clients/0x***LLA***/config** and **/var/opt/ignite/clients/history/0x***LLA***/config** to be removed or emptied.

-i *configuration*

Specifies a system configuration to be used for clients. If any clients have a file in the directory **/var/opt/ignite/clients/0x***LLA***/** that conflicts with the configuration given with the **-i** option, **bootsys** returns an error. This option cannot be used with the **-R** option. The **-f option** may be used with this option to have bootsys remove the client's configuration file and force it to the given *configuration*.

The *configuration* parameter must be a quoted string matching a configuration found in the **/var/opt/ignite/INDEX** file. The configurations available may be listed with the command:

```
/opt/ignite/bin/manage_index -l
```

-R *target_operating_system_release*

Specifies the version of the HP-UX operating system that will be installed on the client. For example, **-R** *B.11.23*. A release directory matching the selected target operating system containing the required Ignite-UX kernel and RAM file system must exist under **/opt/ignite/boot**, for example **/opt/ignite/boot/Rel_B.11.23**. This option cannot be used with the **-i** option. The **-f** option may be used with this option in order to have bootsys remove the client's configuration file and force it to the given target operating system.

-l *sw-sel=true|false*

One or more **-l** options may be supplied to modify the default software that is selected for loading by the set of clients being installed. This is useful when automating installations where using just the **-i** option to select the configuration does not give enough control. An example command line usage of this option is:

bootsys -l *golden image1=true* \
-l *golden image2=false myclient*

-V *var=value*

One or more **-V** options may be supplied to set a value to a variable. The possible values may be a number, a number followed by a **size suffix**, or a string. A size suffix can be K or KB for kilobytes, or M or MB for megabytes. An example command line usage of this option is:

bootsys -V *foo1=fooval* **-V** *foo2=200MB myclient*

-s *grace*

Causes **bootsys** to use **shutdown** with the specified *grace* period (in seconds). Normally **bootsys** uses the **reboot -q** command and reboots the system immediately with no warning to users.

-S

Directs **bootsys** to use **ssh** based protocols instead of **remsh** based protocols when communicating with remote systems. For more information, see the section **Using SSH with bootsys**.

-t *install_tftp_server*

Specifies the default system to be used as the Ignite-UX install server. The *install_tftp_server* is normally the same system that is being used as the boot server, which is normally the same as the system on which **bootsys** is run. *install_tftp_server* is a system that has the Ignite-UX product loaded and is configured with *tftp*(1) access to the files listed under **FILES**.

-g *gateway*

Specifies the default IP address for the *gateway* system through which the client system may reach the install_tftp_server and the SD_server system. (See *route*(1M)).

If no default *gateway* is specified, the default is the same as the client host's IP address and wild-card routing is performed (on networks that support it).

-m *netmask*

Specifies the default *netmask* for the client system to use in reaching the install and SD server systems. This is necessary if your network uses subnetworks. The *netmask* is the same as what is supplied to the ifconfig command (see *ifconfig*(1M)). *netmask* may be supplied either in dot notation (for example, 255.255.248.0) or as a hexadecimal number with a leading 0x (such as, 0xfffff80).

client
client:*IP_address*

Specifies the host name(s) of the clients to be rebooted using Ignite-UX. If the client should use a different IP address when rebooted, or if the client's IP address cannot be obtained using **nslookup**, then it can be specified using the second syntax shown.

-c *Ignite-UX_Server*

Directs **bootsys** to execute in "client" or "local" mode. Thus, instead of operating remotely on one or more clients, **bootsys** contacts the specified *Ignite-UX_Server*, and then performs a local reboot. All other options are processed normally.

-?

Display the help screen.

Blocking bootsys Attempts

For computing environments in which the server has **.rhosts** permission to all systems, it is sometimes desirable to block the **bootsys** command from being able to execute on critical systems. This helps to prevent users from accidentally targeting the wrong system or a system that is actively in use and not yet ready to be installed.

bootsys may be blocked by creating the file **/.bootsys_block** on the clients that you do not want to be reinstalled. This file may either be empty, contain the word **confirm**, and/or it may contain a message that explains why the client is blocking **bootsys**. If the file is empty, **bootsys** refuses to execute on the target. If the first line of the file contains the word **confirm**, the user running **bootsys** on the Ignite-UX server is asked if client installation should continue. If the file contains any other text, that text is displayed to the console when the **bootsys** command was executed. Typically this text is used to explain why the client is blocking any **bootsys** attempts.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

Using SSH with bootsys

The **bootsys** command remotely executes a series of commands on the client system from the Ignite-UX server, some of which are protected from waiting for an indefinite period of time. For these reasons, **bootsys** must use an access method that does not require the user to repeatedly enter the client system's root password. In **remsh** mode, this is accomplished using the **.rhosts** file. In **SSH** mode, **bootsys** must use public/private key authentication in conjunction with an **ssh-agent** process. This allows **bootsys** to automatically perform its tasks without intervention.

When **bootsys** is invoked with the **-S** flag, it verifies that the user has created a public RSA or DSA key, and checks the **SSH_AGENT_PID** environment variable to determine whether an **ssh-agent** process has been started. If this check shows no **ssh-agent** process has been started, **bootsys** starts the **ssh-agent** and uses **ssh-add** to register the user's private key with the agent. If the private key is protected with a passphrase, the user is prompted for the passphrase.

The **bootsys** command then attempts to execute commands on the client via **ssh**. If the user's public key has not been placed in the **~/.ssh/authorized_keys** file on the client, **bootsys** prompts the user asking whether to copy the public key files **~/.ssh/id_rsa.pub** and **~/.ssh/id_dsa.pub** files to the client's **~/.ssh/authorized_keys** file using **ssh**. If **bootsys** starts an **ssh-agent** process, that process will be killed before **bootsys** exits.

Important: The bootsys command does not attempt to determine if you have modified the value for AuthorizedKeysFile in the **sshd** configuration file **/etc/opt/ssh/sshd_config** on the client system. If you modify this value Ignite-UX **ssh** support will not work. Also, the **bootsys** command does not attempt to locate non-default locations for the user's public/private key files.

Two public/private key pairs are involved when communicating with a remote system: a host key and a user key. The host key identifies the computer systems involved in the communication, and the user key identifies the specific user. The first time **ssh** is used to communicate with a remote system, unless your system administrator has already registered the remote system in the file **/etc/ssh/ssh_known_hosts**, the user is prompted with a message similar to:

```
The authenticity of host 'test4 (10.1.48.124)' can't
be established. RSA key fingerprint is
3d:6b:c6:ce:b0:58:60:2a:53:c1:19:b5:ec:84:77:b1.
Are you sure you want to continue connecting (yes/no)?
```

This prompt is asking the user to accept the key identifying the remote host. Answer "yes" to this question if you are certain that you are connected to the intended client system. For more information about host keys, see the VERIFYING HOST KEYS section of **ssh(1)**.

The host key prompt above is only issued once. Upon answering "yes", that host's public key is stored in the user's **~/.ssh/known_hosts** file and is validated for each new ssh session. If the private host key changes for any reason, a message similar to

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now
(man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
4b:85:69:9a:ed:9d:b9:0a:e0:23:d7:d9:1c:c0:38:0e.
Please contact your system administrator.
Add correct host key in /home/smith/.ssh/known_hosts to get
rid of this message.
Offending key in /home/smith/.ssh/known_hosts:32
RSA host key for test4 has changed and you have requested
strict checking.
Host key verification failed.
```

is issued. This may occur for several reasons. It is possible that OpenSSH has been removed and reinstalled, HP-UX has been reinstalled, a networking misconfiguration problem exists so that you are connected to the wrong client, or a genuine breach of security has occurred. Before taking any action, the user should determine what has caused the host key to change. If the host key changed for a legitimate reason, such as reinstallation of the operating system of the client, it is safe to remove the offending entry (in this case, line 32) from the known_hosts file. See **ssh(1)** and **ssh-keygen(1)** for more information.

For more information on the features, benefits, and assistance with troubleshooting ssh, please review the material available at the following URL:

http://www.docs.hp.com/en/internet.html#Secure%20Shell

RETURN VALUE

bootsys will exit with a status of 0 (zero) if all clients were rebooted successfully. A non-zero value is returned if any problem occurred.

EXAMPLES

Boot the following three systems immediately from the Ignite-UX server, such that the clients wait for the **Ignite** UI to configure them:

bootsys -w *client1 client2 client3*

Automatically install *client1* specifying the configuration for it to use:

bootsys -a -i *'HP-UX B.11.23 Default' client1*

Automatically install *client1* providing 10 minutes for users to log off:

bootsys -a -s *600 client1*

Automatically install *client1* using a different IP address than what it is currently assigned:

bootsys -a *client1:1.2.3.45*

Automatically install the systems listed, and ignore clients that cannot be accessed:

bootsys -a -d *client1 client2 client3*

If a client had **bootsys** executed on it, but you want to reboot back to HP-UX off the disk on which **bootsys** modified the **AUTO** file and the **-r** option was used, you can execute the command **/tmp/bootsys_prep -u** on the client. If **-r** was not used, you can interact with the **ISL** to get the system to boot from the normal kernel **/stand/vmunix**, and the changes will automatically be undone. The command **/tmp/bootsys_prep** is left on the client by the **bootsys** command and is removed

when its **-u** option is used. Note that the **AUTO** file is automatically restored if you installed to a different disk from the original boot disk. The **bootsys_prep -u** command is executed automatically the first time you boot from a disk on which **bootsys** has previously been executed:

```
boot primary isl
ISL> hpux /stand/vmunix
```

Automatically install *B.11.23* on the systems listed, and ignore clients that cannot be accessed:

bootsys -a -R *B.11.23* **-d** *client1 client2 client3*

WARNINGS

The client's current network settings are used only during install and they are overwritten by any "final" settings in the configuration files.

FILES

/opt/ignite/boot/Rel_*release***/INSTALLFS**
/opt/ignite/boot/Rel_*release***/VINSTALLFS**
/opt/ignite/boot/Rel_*release***/WINSTALLFS**
The RAM file system used by PA-RISC install clients. Configuration information available at client boot time is stored in the first 8KB of this file. Normally the latter two are symbolically linked to the first on the Ignite-UX server.

/opt/ignite/boot/Rel_*release***/IINSTALLFS**
The RAM file system used by Itanium®-based install clients. Configuration information available at client boot time is stored in the first 8KB of this file. By default, **instl_adm** keeps all of the RAM file systems in sync with each other.

/opt/ignite/boot/Rel_*release***/INSTALL**
The kernel booted by 32-bit PA-RISC-install clients.

/opt/ignite/boot/Rel_*release***/VINSTALL**
The kernel booted by V-class PA-RISC-install clients.

/opt/ignite/boot/Rel_*release***/WINSTALL**
The kernel booted by 64-bit PA-RISC-install clients.

/opt/ignite/boot/Rel_*release***/IINSTALL**
The kernel booted by Itanium®-based install clients.

/var/opt/ignite/INDEX
The **INDEX** file used on the Ignite-UX server to define the configurations available. The **-i** option may be used to specify one of these configurations.

SEE ALSO

ssh(1), ssh-agent(1), ssh-add(1), instl_adm(1M), isl(1M), setboot(1M), remsh(1M), ignite(5), ssh-config(5), http://www.docs.hp.com/en/internet.html#Secure%20Shell

NAME

check_net_recovery − compare contents of network recovery image to a running system

SYNOPSIS

/opt/ignite/bin/check_net_recovery -s *Ignite-UX_server* [**-?**]

DESCRIPTION

check_net_recovery compares the files on a currently running system with a network recovery archive created by **make_net_recovery.** A list of files on the current system is compared to the list of files in the archive, and a report is generated showing those files that have been added, deleted, and changed since the archive was created.

The system administrator may use this report to assist in determining whether a new archive should be created with **make_net_recovery.**

Options

check_net_recovery recognizes the following options:

-s *Ignite-UX_server*
Specifies the hostname of the *Ignite-UX server* containing the recovery archive.

-? Displays the help screen.

Report Format

The **check_net_recovery** report consists of for sections: files added, files deleted, files changed, and a one-line summary.

Lines beginning with **Added:** list those files that have been added under locations archived by **make_net_recovery** since the recovery image was created. These files will be lost if the system is recovered from the archive.

Lines beginning with **Deleted:** list the files that have been deleted from locations archived by **make_net_recovery** since the recovery image was created. These files will reappear if the system is recovered from the archive.

Lines beginning with **Changed:** list those files in the archive which have been modified, and indicates which attributes have changed (mode, modification time, or size). It also displays the values of these attributes in the archive and on the running system. The format of the record is a line beginning with the string **Changed:,** followed by the path of the file, followed by a tab-separated list of modification records indicating which attributes have changed.

Changed: <file path>[TAB]moderec[TAB]mtimerec[TAB]sizerec

where moderec is of the form

```
mode: oldmode --> currentmode
mtime: oldtime --> currenttime
size: oldsize --> currentsize
```

Only the modification records corresponding to those attributes that have changed are displayed.

The final line of the report begins with the word **Summary:** and the total number of files in the archive, on the system, and the number of files added, deleted, and changed on the system since the archive was created.

It is important to note that check_net_recovery only compares files in locations archived by **make_net_recovery.** See *make_net_recovery(1M)*, in particular the discussion of the **-f content_file** and **-x content-options** arguments and the *Including and Excluding From Archive* section. Files added, removed, or modified outside of the locations included in the archive are not analyzed by check_net_recovery and are not included in the summary statistics printed at the end of the report.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

check_net_recovery -s hpfcdn

may produce the following output:

```
Generating new list of files
Comparing file lists

Added: /etc/opt/resmon/log/client.log.old
Added: /etc/opt/resmon/log/registrar.log.old
Changed: /dev/console   mtime: 1094234510 --> 1094241446
Changed: /dev/log       mtime: 1094234510 --> 1094243483
Changed: /dev/null      mtime: 1094234509 --> 1094236571
Changed: /dev/pts/ta    mode: 020666 --> 020620 mtime: 1094154713 --> 1094249204
Changed: /dev/rmt/0mn   mtime: 1094156268 --> 1094236568
Changed: /dev/rmt/c0t3d0BESTn   mtime: 1094156268 --> 1094236568
Changed: /dev/telnetm   mtime: 1094141210 --> 1094236132
Changed: /etc/mnttab    mtime: 1094233598 --> 1094236184
Changed: /etc/utmp      mtime: 1094227632 --> 1094241445
Changed: /etc/utmpx     mtime: 1094228740 --> 1094241467        size: 2100 --> 2
Changed: /tmp   mtime: 1094234510 --> 1094249191
Changed: /usr/lib       mtime: 1094167167 --> 1094234913
Changed: /usr/lib/ignite_bootlif        mtime: 1094154713 --> 1094234914
Changed: /var/adm/wtmp  mtime: 1094227632 --> 1094241445        size: 2520 --> 2
Summary: Archive has 33626 records, system has 33626 records.  2 Added, 0 Delete
```

This example was generated very soon after creating a recovery image. As time passes and the system is used, more modifications will occur, resulting in the archive image becoming outdated. It is important to remember that only those files in locations included in the archive are analyzed by the check-recovery utilities.

SEE ALSO

make_net_recovery(1M), ignite(5).

NAME

check_tape_recovery − compare contents of a tape recovery image to a running system

SYNOPSIS

/opt/ignite/bin/check_tape_recovery [**-?**]

DESCRIPTION

check_tape_recovery compares the files on a currently running system with a tape recovery archive created by **make_tape_recovery.** A list of files on the current system is compared to the list of files in the archive, and a report is generated showing those files that have been added, deleted, and changed since the archive was created.

The system administrator may use this report to assist in determining whether a new archive should be created with **make_tape_recovery.**

Options

check_tape_recovery recognizes the following option:

-? Displays the help screen.

Report Format

The **check_tape_recovery** report consists of for sections: files added, files deleted, files changed, and a one-line summary.

Lines beginning with **Added:** list those files that have been added under locations archived by **make_tape_recovery** since the recovery image was created. These files will be lost if the system is recovered from the archive.

Lines beginning with **Deleted:** list the files that have been deleted from locations archived by **make_tape_recovery** since the recovery image was created. These files will reappear if the system is recovered from the archive.

Lines beginning with **Changed:** list those files in the archive which have been modified, and indicates which attributes have changed (mode, modification time, or size). It also displays the values of these attributes in the archive and on the running system. The format of the record is a line beginning with the string **Changed:,** followed by the path of the file, followed by a tab-separated list of modification records indicating which attributes have changed.

Changed: <file path>[TAB]moderec[TAB]mtimerec[TAB]sizerec

where moderec is of the form

```
mode: oldmode --> currentmode
mtime: oldtime --> currenttime
size: oldsize --> currentsize
```

Only the modification records corresponding to those attributes that have changed are displayed.

The final line of the report begins with the word **Summary:** and the total number of files in the archive, on the system, and the number of files added, deleted, and changed on the system since the archive was created.

It is important to note that check_tape_recovery only compares files in locations archived by **make_tape_recovery.** See *make_tape_recovery(1M)*, in particular the discussion of the **-f content_file** and **-x content-options** arguments and the *Including and Excluding From Archive* section. Files added, removed, or modified outside of the locations included in the archive are not analyzed by check_tape_recovery and are not included in the summary statistics printed at the end of the report.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's

specified locale.

EXAMPLES

check_tape_recovery

may produce the following output:

```
Generating new list of files
Comparing file lists

Added: /etc/opt/resmon/log/client.log.old
Added: /etc/opt/resmon/log/registrar.log.old
Changed: /dev/console   mtime: 1094234510 --> 1094241446
Changed: /dev/log       mtime: 1094234510 --> 1094243483
Changed: /dev/null      mtime: 1094234509 --> 1094236571
Changed: /dev/pts/ta    mode: 020666 --> 020620 mtime: 1094154713 --> 1094249204
Changed: /dev/rmt/0mn   mtime: 1094156268 --> 1094236568
Changed: /dev/rmt/c0t3d0BESTn   mtime: 1094156268 --> 1094236568
Changed: /dev/telnetm   mtime: 1094141210 --> 1094236132
Changed: /etc/mnttab    mtime: 1094233598 --> 1094236184
Changed: /etc/utmp      mtime: 1094227632 --> 1094241445
Changed: /etc/utmpx     mtime: 1094228740 --> 1094241467        size: 2100 --> 2
Changed: /tmp   mtime: 1094234510 --> 1094249191
Changed: /usr/lib       mtime: 1094167167 --> 1094234913
Changed: /usr/lib/ignite_bootlif        mtime: 1094154713 --> 1094234914
Changed: /var/adm/wtmp  mtime: 1094227632 --> 1094241445        size: 2520 --> 2
Summary: Archive has 33626 records, system has 33626 records.  2 Added, 0 Delete
```

This example was generated very soon after creating a recovery image. As time passes and the system is used, more modifications will occur, resulting in the archive image becoming outdated. It is important to remember that only those files in locations included in the archive are analyzed by the check-recovery utilities.

SEE ALSO

make_tape_recovery(1M), ignite(5).

NAME

copy_boot_tape − replicate PA-RISC boot tape

SYNOPSIS

/opt/ignite/bin/copy_boot_tape -u|**-p** *no_rewind_tape_device* [**-d** *directory*] [**-abmr**] [**-?**]

DESCRIPTION

copy_boot_tape may be used to make a copy of a boot tape created with **make_tape_recovery** or **make_boot_tape**. It does not handle a boot tape created by **make_ipf_tape**. If the system has two tape devices, the copy may be created from tape to tape. If there is only one tape device available, there must be a directory with enough space for all of the contents of the tape. **copy_boot_tape** uses **dd** to extract the boot area of the tape, and uses either **dd** or **pax** to extract the system archive. If the recovery kit consists of a single tape, **dd** is used. If it is a multi-tape recovery kit, indicated by **-m**, **pax** is used. In the case where the tape contents are extracted to disk, the boot area is put in a file in the target directory entitled **bootimage**, and the system archive is put in either a file **systemimage** (via **dd**) or extracted into the directory **systemimage** (via **pax**).

Options

copy_boot_tape recognizes the following options:

- **-u** *no_rewind_tape_device* from which to read the recovery tape.
- **-p** *no_rewind_tape_device* to which to write the recovery tape.
- **-d** *Directory* to be used for unpacking the contents of the boot tape. The default is **/var/tmp**.
- **-a** When used with the **-u** option, extract only the system archive from the tape. When used with the **-p** option, write only the system archive to tape. (The tape will not be bootable.)
- **-b** When used with the **-u** option, extract only the boot area from the tape. When used with the **-p** option, write only the boot area to tape. (The tape will be bootable without software to install.)
- **-m** indicates that the boot tape is the first of a multi-tape recovery kit. (Tape-to-tape copy is not allowed for multi-tape kits.)
- **-r** Overwrite the system archive on a boot tape with the contents of the file **systemimage** in the directory indicated by the **-d** option.
- **-?** Display the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

copy_boot_tape will exit with a value of 0 (zero) if contents of the tape are successfully extracted and copied to either another tape or file system, and will exit with a non-zero value if any problem occurred.

EXAMPLES

Extract the boot area and the system archive to the files **bootimage** and **systemimage** in the directory */extra_space*: (This will prompt for a blank tape in **/dev/rmt/c0t0d0DDS1n** to copy the extracted files to another tape.)

```
copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -d /extra_space
```

Extract the boot area and the system archive to a second tape on the system:

```
copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -p /dev/rmt/c1t1d0DDS1n
```

Extract only the boot area to the file **bootimage** in the directory */extra_space*:

copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -b -d */extra_space*

Extract only the boot area to a second tape on the system: (This will create a bootable tape without content to install. This is useful for starting a media install to be completed from the network.)

copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -p /dev/rmt/c1t1d0DDS1n -b

Extract only the system archive, to file **systemimage** in the directory */extra_space*:

copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -a -d */extra_space*

Extract only the system archive to a second tape on the system: (This will create an archive tape that is not bootable.)

copy_boot_tape -u /dev/rmt/c0t0d0DDS1n -p /dev/rmt/c1t1d0DDS1n -a

Replace the archive file on a bootable tape with an archive file **systemimage** on disk in the directory */extra_space*: (This system archive needs to closely resemble the one that is being replaced; i.e., the same operating system, architecture, and archive method.)

copy_boot_tape -p /dev/rmt/c1t1d0DDS1n -r -d */extra_space*

Create a bootable tape using the file **bootimage** in the directory */extra_space*. The **bootimage** file has to be a LIF file and contain at minimum **ISL**, **AUTO**, and **HPUX**. Use **make_medialif** to create the LIF file and **lifls** to display its contents:

copy_boot_tape -p /dev/rmt/c1t1d0DDS1n -b -d */extra_space*

SEE ALSO

dd(1), lifls(1), mt(1), pax(1), make_boot_tape(1M), make_ipf_tape(1M), make_medialif(1M), make_tape_recovery(1M).

NAME

instl_adm − maintain Ignite-UX configuration files

SYNOPSIS

/opt/ignite/bin/instl_adm [**-d**] [**-F** *filesystem*|*LIF_volume*] [**-h** *hostname*] [**-s** *SD_server*[:*depot_directory*]] [**-g** *gateway*] [**-m** *netmask*] [**-t** *install_tftp_server*] [**-S** *dhcp_server*] [**-a** *message_file*|-] [**-?**]

/opt/ignite/bin/instl_adm [**-d**] [**-c**] [**-f** *configuration_file*] [**-F** *filesystem*|*LIF_volume]* [**-?**]

/opt/ignite/bin/instl_adm -T [**-i** *index_file* | **-D** *client_directory* | **-f** *configuration_file*] [**-?**]

/opt/ignite/bin/instl_adm [**-b**] [**-F** *filesystem*|*LIF_volume*] [**-?**]

DESCRIPTION

instl_adm is used on a network install boot server to set, modify, or display the default parameters used when a client connects to the server to install the HP-UX operating system. **instl_adm** may be used to specify the network parameters using the **-h**, **-s**, **-g**, **-m**, **-t**, and **-S** options. An informative message may be given to install users by using the **-a** option. More complex configuration parameters may be stored in a *configuration_file* and applied or tested for syntax by using the **-f** and **-T** options respectively. (See *instl_adm*(4) for details on the syntax of *configuration_file*.)

Options

instl_adm recognizes the following options:

-d Display the current configuration. The format of this information is described in *instl_adm*(4).

The standard output resulting from the **-d** option may be stored in a file, edited, and reapplied using the **-f** option.

-f *configuration_file*

The contents of *configuration_file* will be treated as configuration information and by default is copied into the first 8K of the ***INSTALLFS** file. If the **-T** option is used, only the syntax of the file is checked. If the **-c** option is used, it is copied into the given *LIF_volume* as the file named **CONFIG**. The syntax of *configuration_file* is always checked before it is applied.

-F *filesystem* | *LIF_volume*

Specifies that **instl_adm** should modify/display the configuration stored in *filesystem* or *LIF_volume*. If the specified file is a file system image, **instl_adm** operates on the first 8KB of that image. If the specified file is a LIF volume (like a bootable tape/CD image) and the **-c** option is not given, **instl_adm** operates on the first 8KB of all LIF files matching ***INSTALLFS**. If the specified file is a LIF file and the **-c** option is given, the LIF file entitled **CONFIG** is operated on in totality.

If the **-F** option is not specified or if the **-F** option is used to specify any of the ***INSTALLFS** files under **/opt/ignite/boot/Rel_***release*, all ***INSTALLFS** files under **/opt/ignite/boot/Rel_***release* are updated to have the same first 8KB during a modification operation. Similarly, when operating on a LIF volume, all ***INSTALLFS** files will be updated to have the same first 8KB.

-h *hostname* Specifies the default *hostname* to be supplied to the client during a network install. The IP address of the *hostname* given is determined and stored along with the host name. If *hostname* is given as an IP address, it is looked up and stored as the default. Use this option only if a specific host name and IP address are reserved for install purposes. This should be a temporary IP address used only during installs. Once the client installation is complete, it must be

assigned its own unique IP address.

Supplying this default is not recommended if multiple installs are to execute simultaneously, because only one active host may occupy a given IP address at any given time.

The use of a Dynamic Host Configuration Protocol (DHCP) server is the recommended method of supplying a default host name and IP address to installation clients. See the *bootpd*(1M) manpage for details on DHCP.

-s *SD_server*[:*depot_directory*]
Specifies the default Software Distributor (SD) server and depot directory that is to be used during the install as the source for the *swinstall*(1M), or *swm*(1M) commands. The SD server and depot information is normally retrieved from the configuration files produced by the *make_config*(1M) command, which overrides the default supplied by this option.

It is recommended that the **make_config** command be used to update the configuration files corresponding to the associated depots instead of using this option.

-g *gateway* Specifies the default IP address for the gateway system through which the client system may reach the *install_tftp_server* and the *SD_server* system. (See *route*(1M).)

If no default *gateway* is given, the default is the same as the client host's IP address, and wild-card routing is performed (on networks that support it).

-m *netmask* Specifies the default *netmask* for the client system to use in reaching the install and SD server systems. This is necessary if your network uses subnetworks. The netmask is the same as what is supplied to the **ifconfig** command (see *ifconfig*(1M)). *netmask* may be supplied either in dot-notation (for example, *255.255.248.0*) or as a hexadecimal number with a leading 0x (such as, *0xfffff80*).

-t *install_tftp_server*
Specifies the default system to be used as the Ignite-UX install server. The *install_tftp_server* is normally the same system that is being used as the boot-server, which is normally the same as the system on which **instl_adm** is run. *install_tftp_server* is a system that has the Ignite-UX product loaded and is configured with *tftp*(1) access to the files listed under **FILES**.

-S *dhcp_server* Restricts the system used as the DHCP server for obtaining networking information to the one specified. If not set, any system that responds with either a BOOTP or DHCP response will be used.

-a *message_file*|-
instl_adm stores the contents of *message_file* for use as the opening message displayed to users performing an install. If this option is omitted, any current message is kept. To remove the message completely, use an empty file such as **/dev/null** as the *message_file*. If the **-** character is given for *message_file*, **instl_adm** prompts the user to enter the message on the command line. Once the message is entered, Ctrl-D (end of file) should be typed to end the message.

-c Causes **instl_adm** to operate on the LIF file named **CONFIG** instead of the first 8KB of the file named ***INSTALLFS**. This option is only valid when the **-F** option is used to instruct **instl_adm** to operate on an install tape image. The install tape image is a logical interchange format file. (See *lif* (4).)

On a tape, CD-ROM or DVD, the **CONFIG** file is the location of the standard configuration shipped by Hewlett-Packard Company. This file defines the

default file system and software configuration that is selectable during an install. This file is referenced by the **INDEX** file on the media. On an Ignite-UX server, there may be numerous configuration files referenced by the **INDEX** file.

-T Tests for proper syntax of all the configuration files listed in the **INDEX** file, as well as the configuration file data in the first 8K of the ***INSTALLFS** file (when no **-f** or **-D** option is given). The **INDEX** file to use may be changed using the **-i** option. If the **-f** option is given, only the supplied file is checked for syntax. If the **-D** option is given, not only is the syntax of the configuration file associated with the given client directory checked, but runs through a set of sanity checks to determine its validity. See **FILES** below for the location of the **INDEX** and ***INSTALLFS** files. **instl_adm** no longer treats uninitialized integer variables as an error. That functionality has been moved to **instl_dbg -l**.

-i *index_file* Used with the **-T** option to specify an alternate **INDEX** file to be read for the list of configuration files that are checked for syntax.

-D *client_directory*
Used with the **-T** option causes detailed configuration checks to be run against the configuration for the specified client. This may be used to ensure the client's configuration has a good chance of installing successfully. This option may only be used after the client has been booted and has created the data files in the respective *client_directory*. The *client_directory* is of the form **/var/opt/ignite/clients/0x***LLA*.

-b Clears any previous configuration values set.

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

instl_adm returns 0 (zero) value if the operation was successful, and returns a non-zero value if either a failure occurred or if the configuration file has syntax errors.

EXAMPLES

Display the current message and defaults:

```
instl_adm -d
```

Test the syntax of all the configuration files listed in the **INDEX** file:

```
instl_adm -T
```

Test the configuration of a client for any problems:

```
instl_adm -T -D /var/opt/ignite/clients/0xLLA
```

Change the message contents of file *new_message* while leaving the current network defaults in place:

```
instl_adm -a new_message
```

Install new default parameters for the default gateway and netmask. The **-d** option will cause the information to be displayed after installation:

```
instl_adm -d -g 192.168.48.1 -m 255.255.248.0
```

Extract the current default configuration, edit it, and reinstall the new configuration:

```
instl_adm -d > /tmp/cfg
```

vi */tmp/cfg*
instl_adm -f */tmp/cfg*

Create a customized cold-install tape that gives a message to the user. This uses **dd** to extract the tape data, **instl_adm** to modify it, and **dd** to write a new tape.

mt -t /dev/rmt/0m rew
dd if=/dev/rmt/0m of=*/tmp/tape.image* **bs=2k**
instl_adm -F */tmp/tape.image* **-a message_file**
mt -t /dev/rmt/0m rew
dd if=*/tmp/tape.image* **of=/dev/rmt/0m bs=2k**

AUTHOR

Ignite-UX was developed by Hewlett-Packard Company.

FILES

/opt/ignite/boot/Rel_*release***/INSTALLFS**
/opt/ignite/boot/Rel_*release***/VINSTALLFS**
/opt/ignite/boot/Rel_*release***/WINSTALLFS**
/opt/ignite/boot/Rel_*release***/IINSTALLFS**
PA-RISC 32-bit, PA-RISC 64-bit (V class), PA-RISC 64-bit (non V class), and Itanium®-based file systems used by install clients. Configuration information available at client boot-time is stored in the first 8KB of these files. These are the default files **instl_adm** modifies.

/opt/ignite/boot/Rel_*release***/INSTALL**
/opt/ignite/boot/Rel_*release***/VINSTALL**
/opt/ignite/boot/Rel_*release***/WINSTALL**
/opt/ignite/boot/Rel_*release***/IINSTALL**
PA-RISC 32-bit, PA-RISC 64-bit (V class), PA-RISC 64-bit (non V class), and Itanium®-based kernel booted by install clients.

/var/opt/ignite/INDEX
Contains the list of configuration files that belong to each system configuration.

/var/opt/ignite/config.local
This file is a place holder for local customizations that would be applied to all the standard configurations on the system. It is a convenient place to add post configuration script selections.

/var/opt/ignite/data/Rel_*release***/**
The recommended directory for storing configuration files generated by the *make_config*(1M) command.

/opt/ignite/data/Rel_*release***/**
The location of the default file system configuration file **config** that is shipped with Ignite-UX.

/opt/ignite/data/Rel_*release***/config**
Holds the default configuration information provided by Hewlett-Packard. Modification of this file is neither recommended nor supported. Any modifications are lost if the Ignite-UX product is reloaded. Customizations should be restricted to the **config.local** file, or by saving modified configurations from the **ignite** user interface.

/var/opt/ignite/clients/0x*LLA***/config**
The configuration file used by the install client represented by the **0x***LLA hardware. The directory in which this file exists has other associated data files for that client, as well as those that are used by the* **-T**/**-D** options.

SEE ALSO

ifconfig(1M), instl_bootd(1M), make_config(1M), manage_index(1M), route(1M), swinstall(1M), swm(1M) (on HP-UX 11.31 or later), hosts(4), instl_adm(4), ignite(5).

NAME

instl_bootd − boot protocol server for Ignite-UX clients

SYNOPSIS

/opt/ignite/lbin/instl_bootd [**-s**] [**-r** *reuse_time*] [**-t** *timeout*] [**-d** *debug_level*] [**-c** *command*] [**-C** *command*] [**-P** *port*] [**-b** *boot_file*] [*instl_boottab*] [**-?**]

DESCRIPTION

instl_bootd is a boot protocol daemon that responds to boot requests from clients wishing to install an operating system using the Internet Boot Protocol (BOOTP) as defined in **RFC951** and **RFC1048**. **instl_bootd** may respond to clients without the server's prior knowledge of the client (unlike **bootpd** which requires a client to be registered with the server prior to a boot request).

When **instl_bootd** receives a boot request, it allocates an Internet Protocol (IP) address from a list of available addresses held in the **instl_boottab** file. **instl_bootd** then responds to the client by returning the IP address and the *boot_file*.

Options

instl_bootd recognizes the following options:

-s Allow **instl_bootd** to run as a stand-alone daemon. This option is used when starting the daemon manually or via a system startup script, but not when starting with **inetd** (see *inetd*(1M)). Without this option, **instl_bootd** expects an open socket associated with the server port as its standard input. With this option, **instl_bootd** opens the socket itself. Using **inetd** is the preferred startup method.

-r *reuse_time* Specify the minimum amount of time (in minutes) that must elapse before an IP address may be reused. The default value for *reuse_time* is five minutes. *reuse_time* should correspond to the amount of time a client requires to perform the initial phase of the boot process (that is, transferring the kernel into memory). Once **instl_bootd** allocates an IP address to a client, it will not reallocate the same address to another client for *reuse_time* minutes. This prevents IP address collision during the client's boot phase. **instl_bootd** does not respond to clients (and clients cannot boot) if it cannot successfully allocate an IP address old enough to satisfy the *reuse_time* requirement. In this case, a diagnostic message is logged to **syslogd** (see *syslogd*(1M)). If this condition occurs frequently, adding more IP addresses to **instl_boottab** may be necessary.

-t *timeout* Specify how long (in minutes) **instl_bootd** remains running after responding to a boot request. A *timeout* value of zero means that **instl_bootd** will never exit. The default *timeout* value is 15 minutes. Increasing *timeout* (or setting to it zero), prevents **instl_bootd** from being started and stopped excessively during heavy use. Decreasing *timeout* is useful if **instl_bootd** is rarely used, and its resources need to be reclaimed sooner. The **-t** option has no meaning when used with the **-s** option.

-d *debug_level* Specify the detail of messages logged to **syslogd** (see *syslogd*(1M)). The default value of zero means that only errors and warnings are logged. Values of 1, 2, or 3 cause increasingly more verbose message logging.

-c *command* Specify a *command* to be executed using **system** (see *system*(3S)) when **instl_bootd** responds to a new client's boot request. Two arguments are passed to *command*: the LAN card station address and the IP address allocated to the client.

-C *command* Similar to the **-c** option. Client systems often send multiple boot requests during the boot process. The **-c** option causes **instl_bootd** to execute *command* only the first time it responds to a boot request from a client. The **-C**

option causes **instl_bootd** to execute *command* every time it responds to a boot request.

-P *port* Override the default User Datagram Protocol (UDP) port numbers defined by *services*(4) for **instl_boots** and **instl_bootc**. Default values are 1067 and 1068 respectively. This causes **instl_bootd** to listen and respond to boot requests on *port* and *port*+1. *port* becomes the server port (**instl_boots**) **and** *port*+1 becomes the client port (**instl_bootc**). **This option is useful if more than one daemon is in use, or if the default ports conflict with another service. Some clients may only specify a limited set of port numbers. When using a non-default** *port*, ensure it is set to a value that may be matched by the client hardware.

-b *boot_file* Change the boot file path from the default **/opt/ignite/boot/boot_lif** to *boot_file*. *boot_file* is a Logical Interchange Format (LIF) volume that the client uses to access other boot utilities (see *lif*(4), *hpux*(1M) and *isl*(1M)). *boot_file* must be accessible using the **tftp** service (see *tftp*(1)).

instl_boottab Use the file *instl_boottab* instead of the default file **/etc/opt/ignite/instl_boottab** as the source of available IP addresses to allocate to clients. See below for a description of this file.

-? Displays the help screen.

instl_boottab Description

The *instl_boottab* file contains the list of temporary IP addresses used by clients during the initial phase of their boot process. A permanent IP address for the client is prompted for and used when the boot process is complete, at which time the temporary IP address may be reused to service other boot requests.

The *instl_boottab* file may contain one or more IP addresses. If it contains no addresses, the server will not accept boot requests.

The following is a sample *instl_boottab* file:

```
# This is a comment line
1.23.45.67 # A single entry.
1.23.45.68:::reserve # Reserve when allocated.
1.23.45.69:080009123456:19930225100240:reserve # Reserved
1.23.45.70:080009123457:19930225111433
```

Comments are denoted by the **#** character and may be used anywhere except between fields. Blank lines are also allowed. A template *instl_boottab* is provided in **/etc/opt/ignite/instl_boottab** and contains a header comment explaining the syntax and sample entries, which are commented out.

Each non-comment line of the *instl_boottab* file contains one IP address entry with the IP address specified in "dot" notation (for example, 12.34.123.89). In addition to the IP address, each entry may contain three additional fields, each separated by the **:** (colon) character. The additional fields may be empty or absent. All four fields are defined as follows:

Field 1 Defines the IP address of the entry in "dot" notation. This field cannot be empty and must begin in the first column of the line.

Field 2 Indicates the LAN card station address of the last client to use the entry. It is added automatically by **instl_bootd**, but may be added manually and is useful when field 4 is set to **reserve**.

Field 3 Indicates the date and time (in YYYYMMDDhhmmss format) when the entry was last allocated. This field is updated by **instl_bootd**.

Field 4 May be set manually to the keyword **reserve**, to indicate that the entry may be allocated only to the client with a station address matching field 2. If field 2 is empty, the entry may be allocated to any host. After allocation, field 2 is updated to indicate the

client to which it is allocated. Once field 2 is set, the entry cannot be allocated to any other client.

When **instl_bootd** receives a boot request, it searches *instl_boottab* for an IP address issued to the client during a prior boot request. If the search is unsuccessful, **instl_bootd** searches for the IP address with the oldest time stamp.

An IP address may be marked with a **reserve** keyword, which causes **instl_bootd** to allocate the address to the client specified in field 2. Once a **reserve** address is allocated to a client, it cannot be allocated to a different client unless *instl_boottab* is manually edited to change the reserved status.

Once an IP address is allocated to a client, **instl_bootd** updates fields 2 and 3 of *instl_boottab*. Field 2 identifies the client by its LAN card station address; field 3 indicates the date and time the allocation occurred.

Since *instl_boottab* is modified after each boot request and response, the system administrator should move the file to a temporary location while editing it, and then return it to its permanent location when done. All boot requests are denied when the file is in its temporary location. The *instl_boottab* file is automatically reread after modification.

Multiple **instl_bootd** processes may share a single *instl_boottab* file. Appropriate file locking and retry mechanisms are implemented to ensure that this is possible. Even so, each process must have a unique network port number (see the **-P** option).

Anonymous Itanium®-Based Clients

The **instl_bootd** daemon can support anonymous Itanium®-based clients; however, support is not enabled by default. To enable support, you need to change the following line in **/etc/inetd.conf**:

```
bootps dgram udp wait root /usr/lbin/bootpd  bootpd
```

to:

```
bootps dgram udp wait root /opt/ignite/lbin/instl_bootd  instl_bootd
```

After modifying **/etc/inetd.conf**, execute the command **inetd -c** to force **inetd** to reread its configuration file. If there are any running **bootpd** daemons, they should be killed at this time. Making this change disables the bootpd daemon; the system will not be able to provide general **bootp** or DHCP services. If disabling general **bootp** or DHCP services on a boot helper or Ignite-UX server causes a problem for your environment, do not use this method. Note that **instl_bootd** does not support general DHCP requests and should only be used for the initial boot request.

The *Ignite-UX Administration Guide* contains more information on alternative methods for allowing anonymous Itanium®-based client support with Ignite-UX.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

WARNINGS

If several clients attempt to boot simultaneously and not enough IP addresses are listed in *instl_boottab*, some clients may be denied boot services. From the client perspective, it will appear that the **instl_bootd** services are not working. If this happens, check the *syslogd*(1M) logfile on the server to verify this condition. Adding more IP addresses to **instl_boottab** may help avoid this situation.

DEPENDENCIES

HP 9000 systems not supporting BOOTP use *rbootd*(1M) to provide booting services. *rbootd*(1M) serves older HP 9000 systems by translating boot packets into **BOOTP** and forwarding them to the **instl_bootd** server.

If you use the **/var/adm/inetd.sec** file to restrict usage of the server, the **instl_boots** services must allow the IP address **0.0.0.0** or clients will not be able to boot using this daemon.

In addition to the services provided by **instl_bootd**, the **tftp** service must also be configured on the server system, and *boot_file* must be accessible through the **tftp** service (see *tftp*(1) and *tftpd*(1M)).

AUTHOR

bootpd was developed by Carnegie Mellon University and Stanford University.
instl_bootd was derived from **bootpd** by Hewlett-Packard Company.

FILES

/etc/opt/ignite/instl_boottab
default IP address database

/opt/ignite/boot/boot_lif
default boot_file (LIF volume)

/etc/inetd.conf contains the command line used by *inetd*(1M) to start **instl_bootd**.

/etc/services network services port number database

/var/adm/syslog/syslog.log
default location of the **syslogd** logfile (see *syslogd*(1M)).

SEE ALSO

tftp(1), bootpd(1M), inetd(1M), instl_adm(1M), swinstall(1M), syslogd(1M), tftpd(1M), instl_adm(4), services(4), ignite(5).
Installation and Update Guide for your specific HP-UX operating system release
DARPA Internet Request For Comments RFC951, RFC1048

NAME

instl_combine − combine a LIF file and file system for use on CD/DVD

SYNOPSIS

/opt/ignite/lbin/instl_combine [**-F** *LIF_volume]* **-C** *file_system_image* [**-?**]

DESCRIPTION

instl_combine is used when making an installation CD (or DVD) that contains a boot LIF volume and a mountable file system (either ISO9660/CDFS or HFS). In the case of an HFS file system, it is used to combine the given *LIF_volume* and *file_system_image* into one file that may then be written to a disk or CD device and used for booting as well as mounting. For an ISO9660 file system with the *LIF_volume* already included as a boot partition, **instl_combine** copies the LIF directory header to the beginning of the *file_system_image*. In either case, **instl_combine** modifies the specified *file_system_image*.

The choice of which file system to use (HFS or ISO9660) depends on what system architecture the image is for (PA-RISC or Itanium®-based) and the size of the image (greater than or less than 2GB). The sections below describe when each format is allowed and how **instl_combine** is used with each format.

Options

instl_combine recognizes the following options:

-F *LIF_volume*
Specifies where the *LIF_volume* is located.

-C *file_system_image*
Specifies where the *file_system_image* is located.

-? Displays the help screen.

When to Use an ISO9660 File System

An ISO9660 file system may be used in all situations (PA-RISC/Itanium®-based, >2GB image). However there are a few drawbacks to using this file system layout. Drawbacks such as restricted file-name lengths, and the use of tools (**mkisofs**) not supplied with the all HP-UX Operating Environments (OEs). (See the **DEPENDENCIES** section).

When using an ISO9660 format file system, the *LIF_volume* should be included as an El-Torito boot section. This allows the *LIF_volume* to be located toward the front of the image, and thus allows the entire image to be greater than 2GB without causing problems for programs that access the LIF content during the boot process.

When the *LIF_volume* is included as an El-Torito boot section, **instl_combine** is used with just the **-C** *file_system_image option. In this case*, **instl_combine** only copies/adjusts the LIF directory header to the beginning of the image where it may be found by the **lif** commands and boot programs.

The *LIF_volume* is not required to be included as an El-Torito boot section if the *file_system_image* plus the *LIF_volume* is less than 2GB. In this case, the *LIF_volume* may be wrapped around the *file_system_image* by specifying both **-C** and **-F** options.

When to Use an HFS File System

An HFS file system may only be used to create media for use on PA-RISC systems, and may only be used when the media image size is less than 2GB.

When using an HFS file system, **instl_combine** modifies the *file_system_image* (specified with **-C**) by wrapping the *LIF_volume* (specified with **-F**) around the *file_system_image*. The data portion of the *LIF_volume* is appended to the *file_system_image*, and the LIF directory header is inserted at the beginning.

If an HFS file system is used, the recommended approach to make the file system image is to create a LVM logical volume large enough to hold the contents to be copied to the CD/disk. Then mount the logical volume, copy the desired files to the mounted volume, unmount the volume, and use the **dd** command to extract the file system image from the logical volume device to a regular file.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Example 1

Create a bootable PA-RISC Ignite-UX LIF image specifying the configuration that has been created on the server especially for use with the CD. See *make_medialif* (1M) and *instl_adm*(4) for details on modifying the configuration files for use with CD/disk media.

```
make_medialif -c "B.11.23 CD cfg" -r B.11.23 -l /var/tmp/ilif
```

Example 2

Create an HFS file system image for PA-RISC that contains the archive entitled *myarchive.gz*, then use **instl_combine** to combine the LIF file created above with the file system image. Note that a 2KB (or a multiple of 2KB) fragment size is required for a CD device.

```
lvcreate -L 550 -n myvol vg00
newfs -F hfs -f 2048 /dev/vg00/rmyvol
mkdir /mnt
mount /dev/vg00/myvol /mnt
cp myarchive.gz /mnt/myarchive.gz
umount /mnt
dd if=/dev/vg00/rmyvol of=/var/tmp/fs_image bs=1024k
/opt/ignite/lbin/instl_combine -F /var/tmp/ilif -C /var/tmp/fs_image
```

Example 3

Create an ISO9660 format CD/DVD image (that may be greater than 2GB in size) using the open source **mkisofs** utility: (the */var/tmp/ilif* file is the result of running **make_medialif** as shown above).

Step 1: Create a pseudo-root directory containing the files to be copied to the CD/DVD file system. For example:

```
mkdir /var/tmp/cdroot
cp goldenimage.tar.gz /var/tmp/cdroot
```

Step 2: Copy the *LIF_volume* created by **make_medialif** into the pseudo root:

```
cp /var/tmp/ilif /var/tmp/cdroot/ilif
```

Step 3: Run **mkisofs** to create the file system image and save it in the file: */var/tmp/cdfs.iso*. Note that even with the **-U**,**-max-iso9660-filenames**, and **-D** options, there are limitations to the lengths of file names, etc. See *mkisofs*(8) for details.

```
mkisofs -U -max-iso9660-filenames -D \
    -o /var/tmp/cdfs.iso \
    -b ilif -no-emul-boot /var/tmp/cdroot
```

Step 4: Run **instl_combine** to relocate the LIF header to the beginning of the ISO9660 image:

```
instl_combine -C /var/tmp/cdfs.iso
```

Step 5: Write the */var/tmp/cdfs.iso* image as a raw file using open source software such as **cdrecord** or software that is supplied with CD/DVD writers. If a PC is used to write the image to media, be aware that not all software that comes with CD/DVD writers support writing a raw image. If this is the case, obtain a different application. The "Adaptec" (tm) "Easy CD Creator" (tm) application is one application that has this feature (use the "File->Create CD from CD Image" menu).

Note: The DVD created in this example will only work on PA-RISC systems as it has not been

created in El-Torito format.

Example 4

Create an Ignite-UX LIF image suitable for use on Itanium®-based architecture systems. Note that **make_medialif** by default creates a *LIF_volume* that matches the system on which the command is run. This example works even when run on a PA-RISC system to create media for an B.11.23 Itanium®-based system, but requires specifying the **-o** *ipf* **-r** *B.11.23 options.*

```
make_medialif -c "B.11.23 CD cfg" -o ipf -r B.11.23 \
    -l /var/tmp/IPF.lif
```

Example 5

Create a bootable CD/DVD for Itanium®-based architecture systems. Note that Itanium®-based systems require an ISO9660 El-Torito format CD/DVD.

Step 1: Create a pseudo-root directory containing the files to be copied to the CD/DVD file system. For example:

```
mkdir /var/tmp/cdroot
cp goldenimage.tar.gz /var/tmp/cdroot
```

Step 2: Copy the *LIF_volume* created by **make_medialif** into the pseudo root. (See the example above for creating a *LIF_volume* for Itanium®-based systems.)

```
cp /var/tmp/IPF.lif /var/tmp/cdroot/IPF.lif
```

Step 3: Copy the generic Itanium®-based boot partition into the pseudo root:

```
cp /opt/ignite/boot/Rel_release/EFI_CD_image \
     /var/tmp/cdroot/EFI_CD_image
```

Step 4: Run **mkisofs** to create the file system image and save it in the file: */var/tmp/cdfs.iso*. This command line will create two boot partitions in the image: one to contain the Itanium®-based boot partition and another to contain the *LIF_volume*. Note that even with the **-U**, **-max-iso9660-filenames**, and **-D** options, there are limitations to the lengths of file names, etc. See the *mkisofs*(8) manpage for details.

```
mkisofs -U -max-iso9660-filenames -D  \
     -o /var/tmp/cdfs.iso  \
     -no-emul-boot -b  EFI_CD_image -eltorito-alt-boot \
     -no-emul-boot -b IPF.lif /var/tmp/cdroot
```

Step 5: Run **instl_combine** to relocate the LIF header to the beginning of the ISO9660 image:

```
instl_combine -C /var/tmp/cdfs.iso
```

Step 6: Now write this */var/tmp/cdfs.iso* image as a raw file using open source software such as **cdrecord** or software that comes with CD/DVD writers.

DEPENDENCIES

The latest cdfs driver patch may be needed when accessing media created by the **mkisofs** command. If the system experiences problems such as the time stamps of files being off by a few hours, or files that cannot be accessed but appear to be there, make sure the system has the latest cdfs driver patch such as PHKL_34153 (B.11.11). The kernel(s) used by Ignite-UX should have these patches included for use during the install. It is up to the user to ensure the patches are loaded on systems on which the media will be manually mounted.

SEE ALSO

cdrecord(1), make_medialif(1M), newfs(1M) instl_adm(4), lif(4), ignite(5), mkisofs(8).

NAME

instl_dbg − parse and debug an Ignite-UX client's configuration files

SYNOPSIS

/opt/ignite/bin/instl_dbg -D *client_directory* [**-f** *file*] [**-v**] [**-a** {**l**|**r**}]
[**-cdlsh**|**-A**] [**-V** *var*[=*value*]] [**-S** *swsel*[=**TRUE**|=**FALSE**]]
[**-U** *use_model*[=**TRUE**|=**FALSE**]] [**-?**]

DESCRIPTION

instl_dbg is used to parse a client's configuration files for syntax errors, place all relevant configuration information into one file, display/set variables, software selections, and use models, and detect any other errors that may occur during the execution of an installation due to faulty configuration files.

The **instl_dbg** command should be executed on the Ignite server. The directory given in the **-D** option must point to a client directory usually located in **/var/opt/ignite/clients**. (See *add_new_client*(1m) for more information about adding Ignite-UX clients.) The client directory must contain **io.info**, **hw.info**, **host.info**, and **config.sys**. It may optionally contain a **config** file. (For more information about these files see *instl_adm*(4).)

instl_dbg will parse the specified client's configuration files as well as any of the server's configuration files that may be referenced by the client's configuration files. It will first scan for any syntax errors. If any are found, execution is halted, and an error is reported. After the syntax is checked, substitutions for variables, use models, and software selections (**sw_sel**) will occur. Finally, a single, unified configuration file will be written (if the **-f** option is specified). More options may be given so that **instl_dbg** will perform more thorough checking or provide more detail.

Options

instl_dbg recognizes the following options:

-D *client_directory*
Specifies the location of the client's configuration files.

-f *file* Specifies the file to which **instl_dbg** will write the final configuration. To write the configuration file to stdout, use **-** for *file.* If no filename is given, **instl_dbg** will not produce an output file. All logic expressions are evaluated in the file, thus making it easier to determine what the end result will be.

-v Instructs **instl_dbg** to produce more **verbose** output in the configuration file. Verbose output includes configuration index definitions, sw_sel definitions, and software category definitions. To display verbose information such as the state of a variable, **sw_sel**s, or use model before and after command line substitution the "very, very verbose" option may be used. Replace the **-v** option with the **-vvv** option. The **-vvv** option implies the **-v** option.

-a {**l**|**r**} Instructs **instl_dbg** to perform access checks for all selected **sw_sel**s. An access check verifies that the remote server may be reached and the specified software may be retrieved. The tests attempt to retrieve the **sw_sel** in a similar manner as the actual installation does. The tests may be performed on the local server when **l** (ell) follows this option, or they may be run on the remote client system when **r** follows this option. The tests will only be performed on the client system if it may be logged into with the **remsh** command. Note that when tests are performed from the server, the tests cannot always tell if the client will be able to access a **sw_sel**.

-c Instructs **instl_dbg** to execute system configuration (sanity) checks and display the results to stdout.

-d Instructs **instl_dbg** to display the disk and file system layout details to stdout.

-l Instructs **instl_dbg** to perform a lint-type checks on the configuration files. These checks look for possible errors such as not initializing a variable with the **init** keyword, redefining disk layout structures (volume groups, physical disks, etc.), and overwriting instead of appending lists.

-s Instructs **instl_dbg** to display all **sw_sel**, variables, and use model attributes to stdout.

-h Instructs **instl_dbg** to display the hardware summary and details read in from the client's hw.info and io.info files.

-A Instructs **instl_dbg** to display all of the optional detail from the **-c**, **-d**, **-l**, and **-s** options.

-V *var*[=*value*] The **-V** option will display the value of *var* if *var* is not immediately followed by the equal (=) sign. If the equal sign follows the variable, *var* is set to *value*, and *var* is not displayed to stdout. The possible values may be a number, a number followed by a **size suffix**, or a string. A size suffix may be K or KB for kilobytes, M or MB for megabytes, or G or GB for gigabytes.

-S *swsel*[=**TRUE**|=**FALSE**]
The **-S** option will display the value of the **sw_sel**, *swsel*, if *swsel* is not immediately followed by the equal (=) sign. If the equal sign follows the sw_sel, then *swsel* is set to **TRUE** or **FALSE**, and *swsel* is not displayed to stdout.

-U *use_model*[=**TRUE**|=**FALSE**]
The **-U** option will display the value of the quoted use model, *use_model*, if *use_model* is not immediately followed by the equal (=) sign. If the equal sign follows the use model, *use_model* is set to **TRUE** or **FALSE**, and *use_model* is not displayed to stdout.

-? Displays the help screen.

Note: The equal sign must *not* be separated by a space when assigning a value to a variable, **sw_sel**, or use model. Also, setting a variable, **sw_sel**, or use model does not modify any configuration files. It only modifies the internal setting much like the user interface would change them.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

instl_dbg returns the following values:

0 No errors during execution.

1 Critical errors, such as out of memory errors or command line errors will produce an exit value of 1.

2 If there were errors while parsing the configuration files, an exit code of 2 is returned.

4 Errors occurring during software access checks result in an exit code of 4.

8 Warnings from lint-type checks will produce an exit code of 8.

16 The inability to write an output file will yield an exit code of 16.

The error codes are additive, therefore, if more than one type of error occurs during the execution of the program an exit code reflecting these errors will be returned. For example, an exit code of 12 (twelve) will occur when there are warnings from the lint-type checks and any software access

checks failed.

EXAMPLES

Debug iux_client's configuration files and print the final configuration file to stdout:

instl_dbg -D */var/opt/ignite/clients/iux_client* **-f -**

Debug iux_client's configuration files, print the disk and file system information to stdout, and save the final configuration file to *my_cfg.out*:

instl_dbg -D */var/opt/ignite/clients/iux_client* **-d -f** *my_cfg.out*

Perform lint-type checks on **instl_dbg**'s configuration files:

instl_dbg -D */var/opt/ignite/clients/iux_client* **-l**

Debug iux_client's configuration files, run the system checks, and print the value of **_hp_root_disk** to the screen:

instl_dbg -D */var/opt/ignite/clients/iux_client* **-c -V _hp_root_disk**

Debug iux_client's configuration files, change the value of use model "Create /export volume", and print the verbose output to *my_cfg.out*:

```
instl_dbg -D /var/opt/ignite/clients/iux_client \
    -U "Create /export volume"=TRUE -v -f my_cfg.out
```

Debug iux_client's configuration files, show the effects upon the disk layout when the value of **_hp_disk_layout** and **_hp_pri_swap** are changed, and print the "very, very verbose" to the screen as well as the verbose output to *my_cfg.out*:

```
instl_dbg -D /var/opt/ignite/clients/iux_client -d \
    -V _hp_disk_layout="Whole disk (not LVM) with HFS" \
    -V _hp_pri_swap= 500MB  -vvv -f  my_cfg.out
```

Debug iux_client's configuration files, select **sw_sel** "LangTools", and verify that all **sw_sel**s may be accessed from iux_client:

```
instl_dbg -D /var/opt/ignite/clients/iux_client -a r \
    -S LangTools=TRUE
```

AUTHOR

instl_dbg was developed by Hewlett-Packard Company.

SEE ALSO

add_new_client(1M), instl_adm(4), ignite(5).

NAME

make_boot_tape − make a bootable tape to connect to an Ignite-UX server

SYNOPSIS

/opt/ignite/bin/make_boot_tape [**-d** *device_file_for_tape*] [**-f** *configuration_file*] [**-t** *tmp_directory*] [**-v**] [**-?**]

/opt/ignite/bin/make_boot_tape [**-d** *device_file_for_tape*] [**-g** *gateway*] [**-m** *netmask*] [**-t** *tmp_directory*] [**-v**] [**-?**]

DESCRIPTION

The tape created by **make_boot_tape** is a bootable tape that contains just enough information to boot a PA-RISC system and connect to the Ignite-UX server where the tape was created. Once the target system has connected with the Ignite-UX server, it may be installed or recovered using Ignite-UX. The tape is not a fully self-contained install tape; an Ignite-UX server must also be present. The configuration information and software to be installed on the target machine reside on the Ignite-UX server, not on the tape. If you need to build a fully, self-contained recovery tape, see *make_tape_recovery*(1M) or *make_medialif* (1M). **make_boot_tape** is used in situations when you have target machines that cannot boot via the network from the Ignite-UX server. This happens either because the machine does not support booting from the network or because it is not on the same subnet as the Ignite-UX server. In this case, booting from a tape generated by **make_boot_tape** means you do not need to set up a boot helper system. A tape created by **make_boot_tape** may be used to kick off a normal Ignite-UX installation. It may also be used to recover from recovery configurations saved on the Ignite-UX server. There is no "target-specific" information on the boot tape. Only information about the Ignite-UX server is placed on the tape. Thus, it is possible to initiate an installation of any target machine from the same boot tape provided that the same Ignite-UX server is used. Likewise, the target machine may be installed with any operating system configuration that is available on the Ignite-UX server.

Typically, the **make_boot_tape** command is run from the Ignite-UX server that you wish to connect with when booting from the tape later.

A key file that contains configuration information is called ***INSTALLFS**. This file exists on the Ignite-UX server at **/opt/ignite/boot/Rel**_*release*/***INSTALLFS** and is also present on the tape created by **make_boot_tape**. See *instl_adm*(4) for details on the configuration file syntax. Unless the **-f** option is used, the configuration information already present in the ***INSTALLFS** file is used on the tape as well. The **make_boot_tape** command will never alter the ***INSTALLFS** file on the Ignite-UX server; it will only change the copy that is placed on the tape.

Options

make_boot_tape recognizes the following options:

-d *device_file_for_tape*

Specifies the device file of the tape device where the boot tape will be created. For the widest usage of the tape, choose a device file that will create a tape that will work in any drive. For example, a tape created with a DDS-1/no compression device file will work in any DDS drive. One of the examples shows how to create this device file. Also, see *mksf* (1M) for information on how to create a DDS-1 device file. On certain systems, it is also possible to boot from a DLT tape drive. By default, the **/dev/rmt/0m** device will be used.

-f *configuration_file*

Specifies that the contents of *configuration_file* will be treated as the configuration information in the ***INSTALLFS** file that is written to the tape. The contents of *configuration_file* will replace any configuration information already present in the ***INSTALLFS** file that will written to the tape. The syntax of *configuration_file* is always checked before it is applied.

See the examples section for a method to use the configuration information already present in **/opt/ignite/boot/Rel**_*release*/***INSTALLFS** as a

starting point for what is written to the tape.

-g *gateway* Specifies the default Internet address for the gateway system through which the target system may reach the Ignite-UX server. This is useful when the target systems are not on the same subnet as the Ignite-UX server.

The configuration already in **/opt/ignite/boot/Rel_***release***/*INSTALLFS** is used as a starting point in this case. The *gateway* specified is added to this data. If your situation requires a netmask to be set (see **-m** below), you will need to specify a netmask with the **-m** option. By default, when a gateway is specified with **-g**, any existing netmask in **/opt/ignite/boot/Rel_***release***/*INSTALLFS** will be ignored.

-m *netmask* Specifies the default netmask for the client to use in reaching the Ignite-UX server. This is necessary if your network uses subnetworks. The netmask is the same as that supplied to the **ifconfig** command (see *ifconfig*(1M).) *netmask* may be supplied either in dot notation (for example, **255.255.248.0**) or as a hexadecimal number with a leading 0x (for example, **0xfffff80**).

The configuration already in **/opt/ignite/boot/Rel_***release***/*INSTALLFS** is used as a starting point in this case. The *netmask* specified is added to this data.

-t *tmp_directory*

Specifies that the directory *tmp_directory* be used to store all the temporary files created by **make_boot_tape**. The default is **/var/tmp**. The disk space requirements for this directory vary somewhat, but typically 50MB of free disk space is necessary.

-v Enables verbose mode. Information is written about the execution of the command to stdout.

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

make_boot_tape will return 0 (zero) if the operation was successful. It will return non-zero if a failure occurred.

EXAMPLES

Create a boot tape on the default tape drive (**/dev/rmt/0m**).

```
make_boot_tape
```

Create a boot tape on a specified (non-default) tape drive. Create a DDS1 device file for the tape drive first. Show as much information about the tape creation as is possible:

```
ioscan -fC tape     # to get the hardware path
mksf -v -H <hardware path> -b DDS1 -n -a
make_boot_tape -d /dev/<devfile created by mksf> -v
```

Create a boot tape and replace the configuration information contained in the **INSTALLFS** file. Use the **/tmp** directory for all temporary files instead of the default **/var/tmp**:

```
instl_adm -d > tmp_configuration_file
# edit tmp_configuration_file as appropriate
```

```
make_boot_tape -f tmp_configuration_file -t /tmp
```

Create a boot tape and specify a different gateway IP address. Set the netmask value as well. All other configuration information is from what is already in the **/opt/ignite/boot/Rel_***release***/INSTALLFS** file.

```
make_boot_tape -g 15.23.34.123 -m 255.255.248.0
```

NOTES

make_boot_tape is currently limited to PA-RISC systems only.

AUTHOR

Ignite-UX was developed by Hewlett-Packard Company.

SEE ALSO

instl_adm(1M), make_medialif(1M), make_net_recovery(1M), make_tape_recovery(1M), instl_adm(4), ignite(5).

NAME

make_bundles − create SD bundles in a depot

SYNOPSIS

/opt/ignite/bin/make_bundles { **-b**| **-B**} [**-i**] [**-n** *name*] [**-t** *title*] [**-c** *category*] [**-o** *psf*] [**-r** *revision*] *depot_path* [**-?**]

/opt/ignite/bin/make_bundles [**-b**] [**-p**| **-f**] [**-i**] [**-c** *category*] [**-o** *psf*] *depot_path* [**-?**]

/opt/ignite/bin/make_bundles { **-p**| **-f**| **-B** } [**-i**] [**-c** *category*] [**-o** *psf*] [**-l** *file*| *product/fileset*...] *depot_path* [**-?**]

DESCRIPTION

make_bundles may be used to create *bundle* container objects in a Software Distributor (SD) depot (see *sd*(4)). **make_bundles** is used most often when dealing with individual patches that are SD products. Since **make_config** only operates at the bundle level of software objects within a depot, using **make_bundles** prior to using **make_config** will ensure that Ignite-UX has access to all software contained in the depot.

Options

The *depot_path* is the only required parameter. If no options other than the **-c** or **-o** options are given, the **-b** and **-p** options are assumed by default.

-b Causes **make_bundles** to operate only on products/filesets that are not currently contained inside a software bundle. If the **-b** option is given without an **-f** or **-p** option, one bundle will be created to contain all filesets not in a bundle.

-B Causes **make_bundles** to create one bundle containing all products/filesets in the depot or those specified on the command line whether they are currently in a bundle or not. It cannot be given with the **-b**, **-f**, or **-p** options.

-p Causes **make_bundles** to create one bundle for each product in the depot. If used with the **-b** option, only products that have contents not within an existing bundle will be placed inside a new bundle.

The bundle names that will be created will be of the form: **b#**_*Product and will be truncated to the 16-character bundle limit. The number following the* **b** will be used only if the name is not unique with a simple **b_** prefix.

-f Similar to the **-p** option, except one bundle for each fileset will be created. **-f** may also be used with the **-b** option.

The bundle names that will be created will be of the form: **b#**_*ProductFileset and will be truncated to the 16-character bundle limit. The number following the* **b** will be used only if the name is not unique with a simple **b_** prefix.

-i Causes **make_bundles** to set the **is_reference** attribute to **true** on each bundle it creates. Bundle wrappers with **is_reference** set to **true** will automatically be installed any time any fileset or product within the bundle is installed. If **is_reference** is **false**, only the filesets and products selected will be installed; the bundle wrapper will not be installed unless it is also selected.

-n *name* When used with the **-B** or when neither the **-f** nor **-p** options are used, it supplies the tag name for the new bundle to be created. If not specified, it will default to the base name of *depot_path* with all illegal bundle-name characters removed, and and it will be truncated to 16 characters.

-t *title* When used with **-B** or when neither the **-f** nor **-p** options are used, it supplies the one line description of the bundle to be created. If not specified, it will default to *depot_path*.

-c *category*	Supplies the category for the bundle(s) that will be created. The default is to leave the bundles uncategorized.
-o *psf*	Causes **make_bundles** to create a product specification file and write it to the given *psf* without actually modifying the target depot. This may be useful in editing the bundle names or descriptions before running **swpackage** to create the bundles in the depot.
-r *revision*	Use the specified revision string for the revision of the bundle to be created. This option is ignored when the **-f** or **-p** options are used, in which case the revision is obtained from the respective fileset or product contained in the bundle (unless the fileset or product does not have a revision of its own).
-l *file*	Supplies a file name which lists the specified products to be bundled. Cannot be used with the **-b** option.
product/fileset	Supplies one or more product/fileset(s) to be bundled. Cannot be used with the **-b** option.
depot_path	The directory path name to the SD depot. *depot_path* must reside on the same system, and the user running **make_bundles** must have insert permission for the depot.
-?	Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

make_bundles will return 0 (zero) if the operation was successful, and it will return 1 if a failure occurred. If the **-b** option is given and all software in the depot is already in a bundle, **make_bundles** will return 2.

EXAMPLES

Create one bundle for each product not already in a bundle within the depot */depots/patches.11.23*. Note that the **-b** and **-p** options are implied:

```
make_bundles /depots/patches.11.23
```

Create one bundle containing everything in the depot, specifying a name, title, and category:

```
make_bundles -B -n Patches -c Patches \
      -t  "11.23 approved patches" /depots/patches.11.23
```

Create one bundle for every product in the depot, and set the category to *all-products*:

```
make_bundles -p -c all-products /depots/patches.11.23
```

Create an SD product specification file (PSF) containing the information from the /var/opt/ignite/depots/Rel_B.11.31/patches_11.31 directory so that all information is present to create the bundle using swpackage:

```
make_bundles -b -i -n PB_Sept_2007 \
    -t "Site patches for September 2007" \
    -o ./PB-Sept_2007.psf -r 1.1 \
    /var/opt/ignite/depots/Rel_B.11.31/patches_11.31
```

Note: The swpackage command necessary to package the bundle created above may be found in the newly generated PSF.

FILES

`/opt/ignite/lib/sd_opts`

Contains the list of SD options used by the **`swlist`** and **`swpackage`** commands run by **`make_bundles`** (as well as many other Ignite-UX commands). These options take precedence over the **`/var/adm/sw/defaults`** file in order to ensure the SD commands behave in an expected manner. Modifications to this file may cause **`make_bundles`** to not work as expected.

SEE ALSO

make_config(1M), sd(4), ignite(5).

NAME

make_config − build Ignite-UX configuration files for SD depots

SYNOPSIS

/opt/ignite/bin/make_config -r *release* [**-o** *os_release*] [**-l** *level*] [**-p**] [**-s** *release_relative_source_depot]* [**-v**] [**-x** *option*] [**-d** *percent*] [**-?**]

/opt/ignite/bin/make_config -s *source_depot* **-c** *configuration_file* [**-a 700**|**800**|**both**] [**-f** *software_file*] [**-o** *os_release*] [**-l** *level*] [**-p**] [**-d** *percent*] [**-v**] [**-x** *option*] [**-?**]

DESCRIPTION

make_config constructs Ignite-UX configuration files which correspond to Software Distributor (SD) depots. When an SD depot is used as part of the Ignite-UX process, it must have a config file which describes the contents of the depot to Ignite-UX. This command may automatically construct such a config file given the name of an SD depot on which to operate. This command should be run when adding or changing a depot that will be used by Ignite-UX.

make_config only generates information for SD bundles within a depot. All HP software is packaged in SD bundle form; products not contained in bundles are ignored.

make_config uses the names of the depot(s) and configuration file(s) to give it clues about what type of configuration file to produce. If the name contains a "700", the resulting configuration file will only be used on Series 700 target machines. If the name contains an "800", the resulting configuration file will only be used on Series 800 target machines. The **-a 700**|**800**|**both** option performs a similar function, with appropriate arguments, to specify the type of configuration file.

Generally **make_config** produces a **sw_sel** clause for each bundle it finds in the depot. The tag for the **sw_sel** is the same as the bundle tag. When multiple bundles in the depot have the same tag (but different architecture, revision, or vendor attributes), **make_config** will build multiple **sw_sel** clauses with tags differentiated by revision, architecture, or vendor, and make them exrequisites of each other (see *instl_adm*(4)). If these bundles are "always load" bundles, this will result in an error and no file will be produced. **make_config** produces a header in the file that shows the name of the file produced, its creation time, and the Ignite-UX version installed when **make_config** was used to create the file. Note that later versions of Ignite-UX should always be able to parse a config file produced by an earlier version.

Errors are always logged to **stderr**. More detail may often be found in the log file at:

/var/opt/ignite/logs/make_config

Options

make_config recognizes the following options:

-r *release*

Construct configuration files for each of the depots found in the default depots directory for the specified release. Ensure that the newly constructed configuration files are included in each of the configuration clauses for the specified release in the file **/var/opt/ignite/INDEX**.

For example, if the specified release is B.11.23, a configuration file will be generated for each depot found in the **/var/opt/ignite/depots/Rel_B.11.23** directory. The generated configuration files will be placed in the **/var/opt/ignite/data/Rel_B.11.23** directory. The configuration files will have the same name as their associated depots, with an appended **_cfg**. For instance, the **/var/opt/ignite/data/Rel_B.11.23/core_700_cfg** configuration file will be generated from the **/var/opt/ignite/depots/Rel_B.11.23/core_700** depot.

If **-s** *release_relative_source_depot* is also specified, the depot named must be relative to the directory implied from the release specified by **-r**.

-s *source_depot*
Specify a single source depot to process. The depot is specified just as in the SD commands. If the depot is on the local machine, specify the depot's path name. If the depot is on a remote machine, specify *remote_host_name*:*remote_depot_path*.

-c *configuration_file*
Specify the name of the resulting configuration file.

-a 700|800|both
Specify the architecture of the bundles which should be processed (matches the *bundle.architecture* attribute). By default, all bundles are processed. If the depot/configuration file name contains 700 or 800, only the respective configuration file types are processed. For example, if you specify "**-a 700**", only bundles which are marked as 700 or 700/800 will be processed.

This option also forces the resulting configuration file to be used only on target machines of the appropriate architecture. The **both** argument may be used to force **make_config** to process all bundles, even if the depot name or configuration file name contains a "700" or "800".

-f *software_file*
Specifies that only the SD software selections specified in the *software_file* will be used when generating the configuration file. By default, all software bundles in a depot are processed.

-o *os_release*
Specify the OS release of the bundles you wish to process. Each bundle supplied by HP has an *sd*(4) attribute entitled *os_release* that describes which OSs on which the bundle may be installed. The *sd*(4) *os_release* attribute matches what is returned from **uname -r**. By default, all bundles within a depot are processed. By specifying an *os_release* value with **-o**, it is possible to select only those bundles in a depot that may be installed on a particular OS.

-l *level*
Determines what level in the file system hierarchy for which to build **impacts** statements. For example, if *level* is 1, **impacts** statements might be generated for **/usr** and **/var**. If level is 2, **impacts** statements could be generated for **/usr/local**, **/usr/bin**, and so forth. It is not recommended to specify a level greater than 2 for performance reasons. The default is 1.

-d *percent*
Decreases the disk space impacts by this percentage for bundles that **make_config** identifies as patch bundles. Since patch bundles typically replace files already installed on a client with just an incremental increase in size, disk space impacts for these bundles can be overstated. The allowable range for *percent* is 1 to 75. Caution is urged when using this option as it may lead to file system overflow and cause the installation to fail during the **swinstall** phase. If this problem happens, the configuration file(s) should be recreated with a smaller *percent* value and the installation repeated. Note that **make_config** will add an impact to **/var** equal to the sum of all the disk space impacts before this reduction is taken which accounts for **swinstall** potentially saving old copies of patched files under **/var/adm/sw/save**.

-p Preview. This instructs **make_config** not to make any changes, but to write messages to stdout explaining what would be done if preview were not enabled. Note that no depots are read in preview mode. The only actions are to display which depot(s) will be processed and which configuration file(s) will be generated.

-v Verbose mode. Write more detail about what was processed to the log file.

-x *option*

Specify particular optional behaviors.

The options available via **-x** are:

allow_missing_content

By default, an error is given if any bundles that an B.11.31 or later operating environment calls out as "required" are missing from the depot. If **-x allow_missing_content** is provided, then only a warning message will be given in this case and the config file will be produced excluding any missing bundles.

no_sd_server

This option instructs **make_config** to generate **sw_source** statements in the configuration files which do not reference a particular SD server or depot. This is useful for generating configuration files which will end up on install media. The assumption is that the depot is found at **/** on the CD/DVD media. If the depot is not located at **/**, that will require additional work by specifying the **sd_depot_dir** attribute yourself.

sd_host_name=*hostname|ip*

sw_source.sd_server entries in the configuration file will have a different IP address for the server than the default. This is useful if your server has multiple IP addresses and you wish to use one other than the default.

load_order=*number*

This option allows you to specify a relative load order in the **sw_source** statement in the configuration file. It is only valid when used with the **-c** option. If the depot contains *HPUXEnvironment*, *HPUXBaseOS*, *OpEnvironments* or *LanguagesUI* bundles, the load order is automatically 0 (or 1 in certain circumstances) and this option is ignored. If the depot contains extension software, the load order is automatically six (6) and this option is ignored. For all other depots, this option may be used to change the default load order of five (5) to a desired load order greater than zero (0).

no_full_sd_spec

This option specifies that the **sw_sel.sd_software_list** entries in the configuration file will reference bundles by their SD tags instead of by their full software specification. This is useful when the underlying depot will be undergoing frequent changes to the SD revision attributes and you do not wish to run **make_config** each time the depot has changed. Note that this option is dangerous since there are no guarantees that the resulting configuration file will closely match the depot over time.

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Process all of the depots found in **/var/opt/ignite/depots/Rel_B.11.23** and create configuration files for them in **/var/opt/ignite/data/Rel_B.11.23**. Make sure all of the created configuration files are in each of the B.11.23 cfg clauses in **/var/opt/ignite/INDEX**:

make_config -r *B.11.23*

Process a single depot (**/var/opt/ignite/depots/Rel_B.11.23/***depot1*) **found in /var/opt/ignite/depots/Rel_B.11.23** and create the configuration file for it in

/var/opt/ignite/data/Rel_B.11.23/*depot1_cfg*:

make_config -r *B.11.23* **-s** *depot1*

To process the depot found in */var/mydepot* and create a configuration file for it at */home/me/my_cfg_file*. (Do not modify **/var/opt/ignite/INDEX**.)

make_config -s */var/mydepot* **-c** */home/me/my_cfg_file*

SEE ALSO

instl_bootd(1M), make_depots(1M), manage_index(1M), print_manifest(1M). instl_adm(4), ignite(5).

NAME

make_depots − build Ignite-UX SD depots

SYNOPSIS

make_depots -d *target_depot* [**-r** *release*] [*option*] [**-?**]

make_depots -r *release* [*option*] [**-?**]

DESCRIPTION

make_depots copies bundles from a Software Distributor (SD) source into a depot which may be used by other Ignite-UX tools. While it may be invoked directly by a user, it is provided mainly for use by other Ignite-UX commands. It goes beyond what **swcopy** provides in that it knows about the organization of Ignite-UX depots and the ignitability of certain application bundles.

Certain category bundles are considered "core" category bundles, and are treated differently than other category bundles. These categories are *HPUXEnvironments* for releases prior to B.11.11, and *HPUXBaseOS*, *OpEnvironments*, and *LanguagesUI* for releases beginning with B.11.11. These behavioral differences are described below.

Errors are always logged to **stderr**. More detail may be found in the log file (see -l below).

Options

make_depots recognizes the following options:

-d *target_depot* Specifies the target depot. If *target_depot* has trailing slash characters (**/**), a warning is given and the characters are stripped. See the **Target Depot** section for more details.

-r *release* Specify the release of the target depot. It must be of the form: **B.**[**0-9**][**0-9**]**.**[**0-9**][**0-9**], such as **B.11.23**. See the **Target Depot** section for more details.

-s *source_depot*

Specifies the source depot. The *source_depot* may be one of four depot types. If *source_depot* contains a ":" character, it must be a depot of the form **host:path**. If *source_depot* is a regular file, it must be a serial depot (the output from **swpackage** with **target_type=tape** directed into a file) and it must be an absolute path name. If *source_depot* is a directory, it must be a local depot. If *source_depot* is a character device file, it must be a DDS tape. If *source_depot* is a block-device file, it must be an unmounted CD. The CD will be mounted during the execution of this command and unmounted when it has completed. The default *source_depot* is **/dev/rmt/0m**.

-o *os_revision* Specifies the operating system revision which will be processed. It must be of the form **B.**[**0-9**][**0-9**]**.**[**0-9**][**0-9**], such as **B.11.23**. For "core" category bundles, *os_revision* must match the architecture exactly. For all other category bundles, *os_revision* must pattern match the bundle **os_revision**. If both **-r** and **-o** options are provided, a warning is issued if they are not the same. The default action is to process all operating system revisions in the source depot. However, the default target depot organization of Ignite-UX is structured such that it is advisable to use this option if the source depot has more than one operating system revision.

-a [*700*|*800*] Specifies which architecture which will be processed. See the **Target Depot** section for more details.

-m *release_suffix*

Specifies a release suffix to add to the depot name. See the **Target Depot** section for more details.

-x *swcopy_option*
: Specifies an option that is passed onto the **swcopy** command invoked by **make_depots** (see *swcopy*(1M)).

-i
: Specifies that only bundles that are marked as "ignitable" should be processed. Ignitable bundles are those that have the **hp_ii.factory_integrate** attribute set to **TRUE**, and may be included on factory-installed HP-UX systems.

-v
: Specifies verbose mode.

-p
: Specifies preview mode. Commands are displayed to **stdout** instead of actually being executed. To view what bundles will be copied, also specify the **-v** option.

-l *logfile*
: Specifies the log file path name for this command. The default is **/var/opt/ignite/logs/make_depots**.

-k
: Inhibits the ejection of DDS tape media after processing is complete. The default behavior is to eject tapes after they have been processed.

-?
: Displays the help screen.

Target Depot

The name of the target depot is determined as follows:

1. If the **-d** option is specified and *target_depot* is an absolute path name, it is used as is. No additional changes are made to the path name. Warnings are given if either the **-r** or **-m** options are used in this case.
2. If the **-d** option is specified and *target_depot* is a relative path name, the **-r** option must also be specified and together the target depot will be located under **/var/opt/ignite/depots/Rel**_*release*/*target_depot*/.
3. If the **-d** option is not specified, the **-r** option must be specified and the target depot will be located under **/var/opt/ignite/depots/Rel**_*release*/.
4. If the source depot contains "core" category bundles, all bundles in the source depot will be packaged in a depot with a name beginning with **core**. If the source depot does not contain any "core" category bundles, those bundles will be packaged in a depot with a name beginning with **apps**. Regardless of whether there are "core" bundles in the source depot, bundles may be filtered out by using the **-i**, **-o** or **-a** options.
5. If the **-a** option is used, the depot name will be appended with either **_700** or **_800**, as appropriate. If the **-a** option is not used and a bundle has an architecture (see *sd*(4)) of 700/800, the target depot will not be appended with either a **_700** or **_800**. If the **-a** option is not used but a bundle has an architecture of either 700 or 800 only, the depot name will be appended with either **_700** or **_800**, as appropriate.
6. If the **-m** option is used, the depot name will be appended with *_release_suffix*.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Process all of the bundles found on a CD containing *B.11.23* bits at */dev/dsk/c0t3d0*:

make_depots -r *B.11.23* **-s** */dev/dsk/c0t3d0*

Process only Series *700 B.11.23* bundles found on a protected CD and pass along a codeword and customer_ID for access:

```
make_depots -r B.11.23 -a 700 -s /dev/dsk/c0t3d0 -o B.11.23 \
    -x codeword="jona than alli sonk" -x customer_ID="Mykids Company"
```

Process only Series 800 ignitable B.11.23 bundles found on a remote depot on client *molly* under a directory entitled *peaches*:

```
make_depots -r B.11.23 -d peaches -s molly:/var/depot/Source \
    -i -o B.11.23 -a 800
```

SEE ALSO

ignite(5).

NAME

make_ipf_tape − create an ANSI labeled bootable tape for an Integrity system

SYNOPSIS

/opt/ignite/bin/make_ipf_tape
-f *config_file* [**-f** *config_file* ...] |
-F *config_file* [**-F** *config_file ...* **-C** *cfg_clause ...]* | **-c** *cfg_clause*
[**-C** *cfg_clause*]
[**-d** *tape_device_file*]
[**-n** *vol_name*]
[**-l** *LIF_file*]
[**-r** *release*]
[**-s** *script_file* ...]
[**-V**]
[**-w** *output_file*]
[**-?**]

DESCRIPTION

make_ipf_tape creates an ANSI labeled bootable tape for an Integrity system. This tape contains all the required boot and install components for the system that the tape is created on. It does not contain the recovery archive.

Errors are always logged to **standard error.**

Options

make_ipf_tape recognizes the following options:

-f *config_file* Specifies an Ignite-UX config file that will go to the **CONFIG** in the LIF volume. If multiple **-f** options are given, the specified files are concatenated into **CONFIG** in the order they appear on the command line. (This **CONFIG** file is referenced in a **cfg** clause in an Ignite-UX **INDEX** file also placed in the LIF volume; this **cfg** clause uses a default **description** string.) Either this option or one of the **-F**, and **-c** options is required.

-F *config_file* Specifies an Ignite-UX config file similar to **-f**, but creates multiple **CONFIG1**,**2**,... LIF files in the LIF volume, each with a corresponding cfg clause in the **INDEX** file whose tag-string is based on **-C** options intermixed with **-F** options. Either this option or one of the **-f**, and **-c** options is required.

-c *cfg_clause* Specifies an Ignite-UX **cfg** clause (tag-string) in the **/var/opt/ignite/INDEX** file from which to get config file information. (Enclose **cfg_clause** in quotes if it contains whitespace.) All config files listed in this clause are concatenated together in the order specified in the **INDEX** file to form a single config file named **CONFIG** in the LIF volume. If there is a description for this **cfg** clause, it is preserved in the **INDEX** file in the LIF volume; otherwise a default is provided. Either this option or one of the **-f**, and **-F** options is required.

-C *cfg_clause* Specifies the tag-string to use in the **INDEX** file in the LIF volume for the **cfg** clause that references the **CONFIG** file (or with **-F**, the next **CONFIG1**,**2**,... file). (Enclose **cfg_clause** in quotes if it contains whitespace.) With one or more **-f** options, the default tag-string is "HP-UX *release* Default", where *release* is as described in the **-r** option. With the **-c** option, the default tag-string is the same as the one specified with **-c**.

-d *tape_device_file*
Specifies a tape drive with full path name. By default, the non-rewind device file **/dev/rmt/0mn** will be used.

-n *volume_name*
Specifies the name of tape volume. Normally, this defaults to the first six characters of "HP" plus month and date the tape created.

-l *LIF_file* Specifies where the resulting LIF volume (boot image) file is to be written. The default is /var/tmp/HPUXIUXLIF.

-r *release* Specifies the OS release for this ANSI bootable tape. The default *release* is the output of **uname -r**, that is, matching the release of the HP-UX system on which the command is run.

-s *script_file* Specifies a script to include in a **SCRIPTS** file in the LIF volume. Multiple **-s** options may be specified. The default script files, which are always included, are listed in the **FILES** section.

-V Includes the **/opt/ignite/Version** file as a **VERSION** file in the LIF volume.

-w *output_file* Specifies a file name for writing out volume label information.

RETURN VALUE

The **make_ipf_tape** command returns **0** upon successful completion of the task, **1** upon issuing a usage message, or **2** upon any other error.

EXAMPLES

Example 1

Create an ANSI labeled bootable tape using the configuration files under **/var/opt/ignite/recovery/latest** and with Ignite_UX version file, then put on a tape in a non-rewindable device **/dev/rmt/1mn:**

```
make_ipf_tape -f /var/opt/ignite/recovery/latest/system_cfg
              -f /var/opt/ignite/recovery/latest/control_cfg
              -f /var/opt/ignite/recovery/latest/archive_cfg
              -d /dev/rmt/1mn -w /tmp/volume_label -V
```

Example 2

Create an ANSI labeled bootable tape using the configuration file **/home/root/myconfig** and with user input volume name "mytape" :

```
make_ipf_tape -f /home/root/myconfig -n mytape
```

FILES

The Integrity boot tape format is different from its counterpart of PA-RISC. It is an ANSI labeled bootable tape and it has following files on tape. Each data file is preceded and followed by files containing ANSI file labels.

ANSI*Volume***Label**
This information can be user specified. See -n option for details.

ANSI*File***Labels**
These file labels are used by HP software to find the files on tape and cannot be customerized by users. They can be file header label and trailer label.

Descriptor*Block*
This file is for EFI firmware to identify an EFI boot tape.

Boot*Loader*
EFI executable responsible for performing OS-specific boot functionality.

FPSWA.EFI
The HP-UX floating point simulator.

AUTO A file indicates the BOOTLOADER command arguments controlling boot.

IINSTALL
Integrity kernel booted by Integrity install clients.

IINSTALLFS
This file is a install file system matching the install kernel.

HPUXIUXLIF
A LIF volume supplies following files needed by Ignite-UX.

INDEX Not normally used in tape recovery, may have custom user recovery configuration to point to network depot.

CONFIG
Ignite-UX tape recovery archive configuration file contents

INSTCMDSIA
gzip-compressed **tar** archive of commands that execute in the RAM file system during an install on Integrity systems.

SYSCMDSIA
gzip-compressed **tar** archive of commands needed to load the rest of the software on Integrity systems. Each *release* has its own version.

RECCMDSIA
gzip-compressed **tar** archive of recovery commands used in the recovery shell on Integrity systems.

SCRIPTS
Holds HP-UX post-load control scripts

SEE ALSO
ansitape(1M), make_tape_recovery(1M), make_medialif(1M), make_sys_image(1M), save_config(1M), instl_adm(4)

NAME

make_medialif − create bootable Ignite-UX LIF media image file

SYNOPSIS

/opt/ignite/bin/make_medialif
-f *config_file* [**-f** *config_file* ...] |
-F *config_file* [**-F** *config_file ...* **-C** *cfg_clause ...]* | **-c** *cfg_clause*
[**-C** *cfg_clause*]
[**-r** *release*] [**-o 32**|**64v**|**64w**|**IA** | **-a**]
[**-s** *script_file* ...] [**-S**]
[**-l** *LIF_file*] [**-vVR**] [**-d** *tmpdir*] [**-?**]

/opt/ignite/bin/make_medialif -m
[**-r** *release*] [**-o 32**|**64v**|**64w**|**IA**]
[**-l** *LIF_file*] [**-vVR**] [**-d** *tmpdir*] [**-?**]

DESCRIPTION

make_medialif creates a bootable LIF media image file (a Logical Interchange Format volume inside an ordinary file). This image file may be copied to a DDS tape or to a writable CD or DVD to create Ignite-UX install media.

Errors are always logged to **stderr**.

Options

make_medialif recognizes the following options:

-f *config_file* Specifies an Ignite-UX config file to put in a LIF file named **CONFIG** in the LIF volume. If multiple **-f** options are given, the specified files are concatenated into **CONFIG** in the order they appear on the command line. (This **CONFIG** file is referenced in a **cfg** clause in an Ignite-UX **INDEX** file also placed in the LIF volume; this **cfg** clause uses a default **description** string.) Either this option or one of the **-F**, **-c**, or **-m** options is required.

-F *config_file* Specifies an Ignite-UX config file similar to **-f**, but creates multiple **CONFIG1**,**2**,... LIF files in the LIF volume, each with a corresponding cfg clause in the **INDEX** file whose tag-string is based on **-C** options intermixed with **-F** options. Either this option or one of the **-f**, **-c**, or **-m** options is required.

-c *cfg_clause* Specifies an Ignite-UX **cfg** clause (tag-string) in the **/var/opt/ignite/INDEX** file from which to get config file information. (Enclose **cfg_clause** in quotes if it contains whitespace.) All config files listed in this clause are concatenated together in the order specified in the **INDEX** file to form a single config file named **CONFIG** in the LIF volume. If there is a description for this **cfg** clause, it is preserved in the **INDEX** file in the LIF volume; otherwise a default is provided. Either this option or one of the **-f**, **-F**, or **-m** options is required.

-C *cfg_clause* Specifies the tag-string to use in the **INDEX** file in the LIF volume for the **cfg** clause that references the **CONFIG** file (or with **-F**, the next **CONFIG1**,**2**,... file). (Enclose **cfg_clause** in quotes if it contains whitespace.) With one or more **-f** options, the default tag-string is "HP-UX *release* Default", where *release* is as described in the **-r** option. With the **-c** option, the default tag-string is the same as the one specified with **-c**.

-r *release* Specifies the OS release for this LIF volume (boot image), such as **B.11.23**. Physical media can only be used to boot one HP-UX OS release, although multiple types of install kernels and RAM file system files may be included for that release using the **-a** option. The default *release* is the output of **uname -r**, that is, matching the release of the HP-UX system on which the command is run.

-o 32|64v|64w|IA

Specifies whether the LIF volume (boot image) should support hardware (install kernel) type 32-bit (PA-RISC), 64-bit (V-class PA-RISC), wide 64-bit (PA-RISC), or Itanium®-based. The default is to match the system on which the command is run. For convenience, **64** may be used as a synonym for **64v**, and any of **ia**, **ipf**, or **IPF** may be used as a synonym for **IA**. This option cannot be combined with the **-a** option.

Note: Incompatible **-r** and **-o** options or equivalent default values cause an error report. The **-o 32** and **-o 64v** types are only supported by HP-UX OS release B.11.11. The **-o IA** type is only supported by HP-UX OS releases B.11.23 and higher.

-a Specifies that all boot kernel and RAM file system files found in **/opt/ignite/boot/Rel_***release* should be included, not just the pair specified with the **-o** option or matching the system by default. This can increase the size of the LIF volume. If any RAM file system file other than **INSTALLFS** is a symbolic link to **INSTALLFS**, this is preserved in the LIF volume. This option cannot be combined with the **-o** option.

-s *script_file* Specifies a script to include in a **SCRIPTS** file in the LIF volume. Multiple **-s** options may be specified. The default script files, which are always included, are listed in the **FILES** section. All scripts are packaged using **tar**, and then compressed using **gzip**, before being put into the **SCRIPTS** file. A *script_file* must be specified as an absolute path name. Scripts defined as described in the **Command and Script Execution Hooks** section of *instl_adm*(4) must be provided here or with the **-S** option.

-S Specifies that all script files found in both **/var/opt/ignite/scripts** and **/opt/ignite/data/scripts** should be included in the **SCRIPTS** file. All script files found there are combined with any **-s** script arguments and packaged in the manner previously described.

-l *LIF_file* Specifies where the resulting LIF volume (boot image) file is to be written. The default path is **/var/opt/ignite/local/uxinstlf.recovery**. You must have write access to this file, but need not necessarily be superuser.

-v Specifies that the command perform a **lifls -l** on the resulting LIF file upon success. It also displays the location of the image file, the release, and configuration.

-V Includes the **/opt/ignite/Version** file as a **VERSION** file in the LIF volume.

-R Includes the **/opt/ignite/boot/Rel_***release***/RECCMDS** expert recovery commands file (and/or **RECCMDSIA** for IA systems) as a **RECCMDS** (and/or **RECCMDSIA**) file in the LIF volume, according to which kernel plus RAM file system files are selected.

-d *tmpdir* Specifies that the directory *tmpdir* be used to store all the temporary files used by **make_medialif**. The default is **/var/tmp**. The disk space requirements in this directory vary somewhat, but typically 100KB of free disk space is necessary.

-m Specifies the creation of a minimal LIF volume (boot image) file. This file is sufficient to boot and contact an Ignite-UX net server. Unless the **-o** option is used, this "boot helper" media contains all boot kernel and RAM file system files found in **/opt/ignite/boot/Rel_***release*, similar to the **-a** option, plus the **ISL**, **AUTO**, and **HPUX** files, but no **INDEX**, **CONFIG**, **INSTCMDS***, **SYSCMDS***, or **SCRIPTS** files. If any RAM file system file other than

INSTALLFS is a symbolic link to **INSTALLFS**, this is preserved in the LIF volume. This option cannot be combined with any of **-f**, **-c**, **-C**, **-a**, **-s**, or **-S**.

-? Displays the usage text.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

The **make_medialif** command returns **0** upon successful completion of the task, **1** upon issuing a usage message, or **2** upon any other error.

EXAMPLES

Example 1

Create a boot LIF image using the configuration file **/home/root/***myconfig and place it in* **/home/root/uxinstlf**:

```
make_medialif -f /home/root/myconfig -l /home/root/uxinstlf
```

Example 2

To create a golden image operating system archive entitled **/var/tmp/***myimage.gz*, possibly built with **make_sys_image** (a **gzip**'ed **tar** archive of a B.11.23 system) on a bootable, installable DDS tape medium to automatically load the archive. First create a file based upon the template file **/opt/ignite/data/examples/core.cfg**:

```
cp /opt/ignite/data/examples/core.cfg /home/root/config_dds
chmod u+w /home/root/config_dds
```

In */home/root/config_dds* comment out all the **nfs_source**, **ftp_source** and **remsh_source** lines, and then change or uncomment the following objects:

software object	field	value
sw_source	**source_type** **source_format** **load_order** **change_media**	**"MT"** **ARCHIVE** **0** **FALSE**
sw_sel	**sw_source** **archive_path** **archive_type** **impacts**	*label of above sw_source* **"1"** **gzip tar** *as returned by archive_impact*

Then create a LIF header that combines this information with the default configuration as follows:

```
make_medialif -f /opt/ignite/data/Rel_B.11.23/config
```

Finally, create a bootable installation DDS medium using the image created above by putting a DDS1 density, no-rewind and no-compression, writable DDS tape in the **/dev/rmt/c0t3d0DDS1n** drive, and executing the following commands:

```
mt -t /dev/rmt/c0t3d0DDS1n rew
dd if=/home/root/uxinstlf_dds of=/dev/rmt/c0t3d0DDS1n obs=2k
dd if=/var/tmp/myimage.gz of=/dev/rmt/c0t3d0DDS1n obs=10k
mt -t /dev/rmt/c0t3d0DDS1n rew
```

Note that the **obs** argument in the second **dd** command should be **5120** if the archive is **cpio** format.

Example 3

To create a boot LIF image (as in Example 2) with a CD rather than a DDS tape, create a file based upon the template file **/opt/ignite/data/examples/core.cfg**, and execute the following commands:

cp /opt/ignite/data/examples/core.cfg */home/root/config_cd"*
chmod u+w */home/root/config_cd*

In */home/root/config_cd* comment out the **nfs_source**, **ftp_source** and **remsh_source** lines and change or uncomment the following objects:

software object	field	value
sw_source	**source_type**	**"DSK"**
	source_format	**ARCHIVE**
	load_order	**0**
	change_media	**FALSE**
sw_sel	**sw_source**	*label of above sw_source*
	archive_path	**"myimage.gz"**
	archive_type	**gzip tar**
	impacts	*as returned by archive_impact*

Then create a LIF header that combines this information with the default configuration by executing the following command:

make_medialif -f /opt/ignite/data/Rel_B.11.23/config
-f */home/root/config_cd* **-l** */home/root/uxinstlf_cd*

There are two ways you may proceed to store the archive: with an HFS file system or a CD file system. An HFS file system may be created using standard HP-UX tools, which may be more familiar. A CD file system (ISO-9660) is more portable with other operating systems, but if non-standard features such as long file names are used, portability may be limited. In either case, there needs to be available disk space of slightly more than twice the size of the archive. See the examples in *instl_combine*(1M) on how to create an ISO-9660 file system using the *mkisofs*(8) command.

To use an HFS file system, there must be sufficient free disk space to store the archive in an empty logical volume if you are using LVM, or an unmounted disk if you are using whole-disk. This example assumes the use of LVM to create a logical volume big enough to contain the archive. A fragment size of 2048 (or a multiple of 2048) is required if the image is used on a CD. There must also be slightly more free space in the existing file system layout for the **dd** command output. Assume that the archive is about 250MB. Execute the following commands:

lvcreate -L 270 -n *archvol* **vg00**
newfs -F hfs -f 2048 */dev/vg00/rarchvol*
mkdir */tmpcd*
mount */dev/vg00/archvol /tmpcd*
cp */var/tmp/myimage.gz /tmpcd*
umount */tmpcd*
dd if=*/dev/vg00/rarchvol* **of=***/var/tmp/fs_image_hfs* **bs=***1024k*

To use a CD file system, some public domain or retail CD tool set must be installed. You must have a utility that creates the CD file system and copy files into it. This approach is very similar to that of the HFS method. Again,using a logical volume to contain the CD file system is recommended. In the end, there must be a regular file that contains the CD file system; for example **/var/tmp/fs_image_cdfs**.

Use **instl_combine** to combine the LIF image and file system by using one of the following two commands, depending upon which file system is used:

/opt/ignite/lbin/instl_combine -F */home/root/uxinstl_cd*
-C */var/tmp/fs_image_hfs*
/opt/ignite/lbin/instl_combine -F */home/root/uxinstl_cd*
-C */var/tmp/fs_image_cdfs*

Using a CD writing utility, transfer either **/var/tmp/fs_image_hfs** or **/var/tmp/fs_image_cdfs** to a writable CD in raw mode. The resulting CD is a bootable, installable Ignite-UX medium. The image may also be transferred to a raw disk. A system booted from that disk

may be helpful to test the image before writing it to CD.

Example 4

To create a boot LIF image (as in Example 2) with an archive on a B.11.23 client for a B.11.11 client, you would execute the following command:

```
make_medialif -f /opt/ignite/data/Rel_B.11.11/config
    -f /home/root/config_cd -r B.11.11 -l /home/root/uxinstlf_cd
```

All subsequent commands in Example 2 should be executed.

Example 5

Create a boot LIF image (as in Example 2) with a software depot that is to be included on the media. Assume that the software depot has been created and is located in */var/depots/source*.

There must be a file that describes the content of this depot. Create the configuration file for */var/depots/source* in */home/root/source_cfg*:

```
make_config -s /var/depots/source -c /home/root/source_cfg
```

In the **sw_source** section of */home/root/source_cfg*, remove the **sd_server** and **sd_depot_dir** lines, and change the **source_type** from **"NET"** to **"MT"**. Package the depot into a serial depot in */var/tmp/serialdepot*. It is possible to use **swpackage** to write directly to the tape and avoid this step if disk space is low, but in most cases using an intermediate serial depot may be easier:

```
swpackage -s /var/depots/source -x target_type=tape
    @ /var/tmp/serialdepot
```

On DDS media, the depot must be the third file on the tape. Note that a single archive may come before the depot, and any number of archives may come after the depot with each one referenced by an appropriate **archive_path** in a unique **sw_sel** in a configuration file, subject to tape capacity. If multiple operating system archives are put onto a single medium, it is advisable to use an **exrequisite** attribute in the **sw_sel** so that only one archive is selected. Also, note that in order to interactively select bundles to load from the depot, the variables **run_ui** and **control_from_server** must be correctly set (see Example 6). Use **make_medialif** to create the LIF header and include the new configuration file for the depot:

```
make_medialif -f /opt/ignite/data/Rel_B.11.23/config
    -f /home/root/source_cfg -f /home/root/config_dds
    -l /home/root/uxinstlf_dds
```

Then create the DDS medium with the following commands:

```
mt -t /dev/rmt/c0t3d0DDS1n rew
dd if=/home/root/uxinstlf_dds of=/dev/rmt/c0t3d0DDS1n obs=2k
dd if=/var/tmp/myimage.gz of=/dev/rmt/c0t3d0DDS1n obs=10k
dd if=/var/tmp/serialdepot of=/dev/rmt/c0t3d0DDS1n obs=10k
mt -t /dev/rmt/c0t3d0DDS1n rew
```

Example 6

To create a bootable DDS medium that allows the installation of a configuration defined on an Ignite-UX server, for example the **"HP-UX B.11.11 Default"** 64-bit capable configuration, you would execute the following command:

```
make_medialif -c "HP-UX B.11.11 Default" -r B.11.11 -o 64v
    -l /home/root/uxinstlf
```

Note that the installation proceeds according to how the variables **run_ui** and **control_from_server** are set in the **INSTALLFS** or **WINSTALLFS** (if this is a 64-bit image) files. For example, if a completely unattended installation is desired with a window of opportunity to interrupt the process, these variables must be set:

```
run_ui=false
control_from_server=false
```

```
env_vars += "INST_ALLOW_WARNINGS=10"
env_vars += "INST_BATCH_MODE_TIMEOUT=10"
```

Place these variables in the **WINSTALLFS** before writing the image to medium using the same tape drive as before:

```
instl_adm -F /home/root/uxinstlf -d > /tmp/cfg
vi /tmp/cfg     # add the above lines
instl_adm -F /home/root/uxinstlf -f /tmp/cfg
mt -t /dev/rmt/c0t3d0DDS1n rew
dd if=/home/root/uxinstlf of=/dev/rmt/c0t3d0DDS1n obs=2k
mt -t /dev/rmt/c0t3d0DDS1n rew
```

Example 7

Unlike DDS tape in which only one software depot is supported, an installation CD may contain multiple software depots. Assume that there are three software depots that contain various engineering, CAD, and documentation applications in */var/depots/eng_apps*, */var/depots/cad_apps* and */var/depots/doc_apps* respectively. On the CD, these depots are stored under the **depots** directory. Also, assume that we have the same archive and configuration file */home/root/config_cd* as in Example 3.

To create configuration files for these depots:

```
make_config -s /var/depots/eng_apps -c /home/root/eng_cfg
make_config -s /var/depots/cad_apps -c /home/root/cad_cfg
make_config -s /var/depots/doc_apps -c /home/root/doc_cfg
```

In the **sw_source** section in all three files, remove the **sd_server** line, change the **source_type** from **"NET"** to **"DSK"**, and then change the **sd_depot_dir** from **"/var/depots/***XXX***_apps"** to **"depots/***XXX***_apps"**, where *XXX* is either **eng**, **cad** or **doc** as appropriate.

To create the LIF header that captures all of this, you would execute this command:

```
make_medialif -f /opt/ignite/data/Rel_B.11.23/config
    -f /home/root/config_cd -f /home/root/eng_cfg
    -f /home/root/cad_cfg -f /home/root/doc_cfg
    -l /home/root/uxinstlf_cd
```

If an interactive installation from these depots is desired, make sure **run_ui** and **control_from_server** are set as described in Example 6.

Since the image will include these depots, as well as the archive, you may assume the total size will fit in 500MB. To create the image, you would:

```
lvcreate -L 500 -n archvol vg00
newfs -F hfs -f 2048 /dev/vg00/rarchvol
mkdir /tmpcd
mount /dev/vg00/archvol /tmpcd
cp /var/tmp/myimage.gz /tmpcd
swcopy -s /var/depots/eng_apps \* @ /tmpcd/depots/eng_apps
swcopy -s /var/depots/cad_apps \* @ /tmpcd/depots/cad_apps
swcopy -s /var/depots/doc_apps \* @ /tmpcd/depots/doc_apps
umount /tmpcd
dd if=/dev/vg00/rarchvol of=/var/tmp/fs_image_hfs bs=1024k
```

Use **instl_combine** to combine the LIF image and file system:

```
/opt/ignite/lbin/instl_combine -F /home/root/uxinstl_cd
                               -C /var/tmp/fs_image_hfs
```

Now using a CD writing utility, transfer */var/tmp/fs_image_hfs* to a writable CD in raw mode. The resulting CD is a bootable, installable Ignite-UX medium. The image may be transferred to a raw disk. A system booted from that disk may be helpful to test the image before writing it to CD.

Example 8

Suppose there is no archive, but instead an entire core operating system depot to install from DDS tape. Assume that the core depot is in */var/depots/core*.

To create a configuration file for this depot, you would:

```
make_config -s /var/depots/core -c /home/root/core_cfg
```

In the **sw_source** section of */home/root/core_cfg*, remove the **sd_server** and **sd_depot_dir** lines, and then change the **source_type** from **"NET"** to **"MT"**. To package the depot into a serial depot in */var/tmp/serialdepot*, you would:

```
swpackage -s /var/depots/core -x target_type=tape
    @ /var/tmp/serialdepot
```

Use **make_medialif** to create the LIF header and include the configuration file for the core depot as follows:

```
make_medialif -f /opt/ignite/data/Rel_B.11.23/config
    -f /home/root/core_cfg -l /home/root/uxinstlf_dds
```

There is no archive to put onto the medium; however, because the depot must be the third file on the tape, an empty file must be placed on the medium. The **mt** command is used to do this. Create the DDS medium with the following commands:

```
mt -t /dev/rmt/c0t3d0DDS1n rew
dd if=/home/root/uxinstlf_dds of=/dev/rmt/c0t3d0DDS1n obs=2k
mt -t /dev/rmt/c0t3d0DDS1n eof
dd if=/var/tmp/serialdepot of=/dev/rmt/c0t3d0DDS1n obs=10k
mt -t /dev/rmt/c0t3d0DDS1n rew
```

Example 9

To load both the core operating system (one or more **HPUXEnvironment** bundles) and standard applications, it is best to have separate configuration files. Some applications do not load correctly at **load_order 0** with the core operating system. So assume you also have additional applications to load in the depot */var/depots/apps*. Use **make_config** to create a separate configuration file for the applications depot as in Example 5. Create the LIF header using both of those configuration files:

```
make_medialif -f /opt/ignite/data/Rel_B.11.23/config
    -f /home/root/core_cfg -f /home/root/apps_cfg
    -l /home/root/uxinstlf_dds
```

Now, combine the core and applications depots into a single global depot and create a serial depot:

```
swcopy -s /var/depots/core \* @ /var/depots/global
swcopy -s /var/depots/apps \* @ /var/depots/global
swpackage -s /var/depots/global -x target_type=tape
    @ /var/tmp/serialdepot
```

Combine the DDS medium:

```
mt -t /dev/rmt/c0t3d0DDS1n rew
dd if=/home/root/uxinstlf_dds of=/dev/rmt/c0t3d0DDS1n obs=2k
mt -t /dev/rmt/c0t3d0DDS1n eof
dd if=/var/tmp/serialdepot of=/dev/rmt/c0t3d0DDS1n obs=10k
mt -t /dev/rmt/c0t3d0DDS1n rew
```

Example 10

On a B.11.23 operating system, create a minimal boot image for a B.11.11 64w-bit system with the **-m**, **-r** and **-o** options:

```
make_medialif -r B.11.11 -o 64w -m -l /tmp/lif_test
lifls -l /tmp/lif_test
```

This will list the contents:

```
volume ISL10 data size 960926 directory size 3 04/06/30 13:59:56
filename     type    start   size   implement  created
================================================
ISL         -12800    16      242       0          04/06/30 13:59:56
AUTO        -12289    264     1         0          04/06/30 13:59:56
HPUX        -12928    272     1024      0          04/06/30 13:59:56
WINSTALL    -12290    1296    83837     0          04/06/30 14:00:03
WINSTALLFS  -12290    85136   49152     0          04/06/30 14:00:06
PAD         BIN       134288  256       0          04/06/30 14:00:06
```

Example 11

On a B.11.11 operating system, create a boot image that includes all boot kernels, file systems and all commands under **/opt/ignite/boot/Rel_***B.11.11*:

make_medialif -a -r *B.11.11* **-c "***HP-UX B.11.11 Default***" **
-l */tmp/lif_test*

lifls -l */tmp/lif_test*

This will list the contents:

```
volume ISL10 data size 960926 directory size 3 04/06/30 13:50:25
filename    type    start   size     implement  created
============================================================
ISL         -12800 16      242       0          04/06/25 09:09:08
AUTO        -12289 16      1         0          04/06/25 09:09:08
INDEX       BIN    24      1         0          04/06/25 09:09:08
CONFIG      BIN    32      410       0          04/06/25 09:09:08
HPUX        -12928 376     1024      0          04/06/25 09:09:08
FWWKAR6     BIN    464     1         0          04/06/25 09:09:08
FWWKAR7     BIN    472     1         0          04/06/25 09:09:08
FWWKAR8     BIN    480     1         0          04/06/25 09:09:08
INSTALL     -12290 488     68184     0          04/06/25 09:09:08
INSTALLFS   -12290 68672   49152     0          04/06/25 09:09:08
VINSTALLFS  -12290 68672   49152     0          04/06/25 09:09:08
WINSTALLFS  -12290 68672   49152     0          04/06/25 09:09:08
VINSTALL    -12290 117824  73806     0          04/06/25 09:09:08
WINSTALL    -12290 191632  83837     0          04/06/25 09:09:12
INSTCMDS    BIN    275472  30243     0          04/06/25 09:09:15
RECCMDS     BIN    305720  987       0          04/06/25 09:09:15
SYSCMDS     BIN    306712  77644     0          04/06/25 09:09:22
SCRIPTS     BIN    384360  45        0          04/06/25 09:09:22
PAD         BIN    384408  256       0          04/06/25 09:09:22
```

Note: The boot image created above is specific to PA-RISC systems, and will not work on Itanium®-based systems.

Example 12

On a B.11.23 operating system, create a boot image for a Itanium®-based system with the **-o** option, you would:

make_medialif -c "*HP-UX B.11.23 Default***" -o** *IA* **-l** */var/tmp/lif_test*
lifls -l */var/tmp/lif_test*

This will list the contents:

```
volume ISL10 data size 960926 directory size 3 04/06/30 14:52:46
filename    type    start   size     implement  created
============================================================
ISL         -12800 16      242       0          04/06/30 14:52:46
```

```
AUTO        -12289 264     1         0          04/06/30 14:52:46
INDEX       BIN    272     1         0          04/06/30 14:52:46
CONFIG      BIN    280     92        0          04/06/30 14:52:46
HPUX        -12928 376     1024      0          04/06/30 14:52:46
IINSTALL    -12290 1400    205161    0          04/06/30 14:52:56
IINSTALLFS -12290 206568  116224    0          04/06/30 14:52:56
INSTCMDSIA BIN    322792  60112     0          04/06/30 14:52:56
RECCMDSIA   BIN    382904  1815      0          04/06/30 14:52:56
SYSCMDSIA   BIN    384720  160242    0          04/06/30 14:53:07
SCRIPTS     BIN    544968  45        0          04/06/30 14:53:07
PAD         BIN    545016  256       0          04/06/30 14:53:07
```

More Examples

More examples, including examples describing how to use **mkisofs** and how to create media for Itanium®-based systems, are found in *instl_combine*(1M).

WARNINGS

If **make_medialif** is being used to create an image to install an archive, it is *imperative* that there be consistency between the release of the archive (B.11.23 for example), the release specified in the configuration file(s) (a line that reads **release=B.11.23**), and the version of **SYSCMDS** selected by the **-r** option (**-r B.11.23**) of **make_medialif**. Failure to provide this consistency will lead to problems with the installation of the archive on the client, problems when booting the installed client, or other similar catastrophic and unrecoverable problems.

FILES

/opt/ignite/boot/boot_lif
default boot-file (LIF volume) used to supply the **ISL** and **HPUX** files.

/opt/ignite/boot/Rel_*release***/INSTALL**
32-bit kernel booted by 32-bit install clients.

/opt/ignite/boot/Rel_*release***/VINSTALL**
64-bit kernel booted by 64-bit install clients.

/opt/ignite/boot/Rel_*release***/WINSTALL**
64-bit kernel booted by wide 64-bit install clients.

/opt/ignite/boot/Rel_*release***/IINSTALL**
Itanium®-based kernel booted by Itanium®-based install clients.

/opt/ignite/boot/Rel_*release***/INSTALLFS**
/opt/ignite/boot/Rel_*release***/VINSTALLFS**
/opt/ignite/boot/Rel_*release***/WINSTALLFS**
/opt/ignite/boot/Rel_*release***/IINSTALLFS**
32-bit, 64-bit, wide 64-bit, and Itanium®-based RAM file systems used by install clients. Configuration information available at client boot time is stored in the first 8 Kb of this file. Some of these files are actually linked.

/opt/ignite/data/Rel_*release***/INSTCMDS**
gzip-compressed **tar** archive of commands that execute in the RAM file system during an install on PA-RISC systems.

/opt/ignite/data/Rel_*release***/INSTCMDSIA**
gzip-compressed **tar** archive of commands that execute in the Itanium®-based RAM file system during an install on Itanium®-based systems.

/opt/ignite/data/Rel_*release***/RECCMDS**
gzip-compressed **tar** archive of recovery commands used in the recovery shell on PA-RISC systems.

/opt/ignite/data/Rel_*release***/RECCMDSIA**
gzip-compressed **tar** archive of recovery commands used in the recovery shell on Itanium®-based systems.

/opt/ignite/data/Rel_*release***/SYSCMDS**
gzip-compressed **tar** archive of commands needed to load the rest of the software on PA-RISC systems. Each *release* has its own version.

/opt/ignite/data/Rel_*release***/SYSCMDSIA**
gzip-compressed **tar** archive of commands needed to load the rest of the software on Itanium®-based systems. Each *release* has its own version.

/opt/ignite/data/scripts/os_arch_post_l
/opt/ignite/data/scripts/os_arch_post_c
post_load_script and post_config_script shell scripts that are run when an operating system archive has been used.

SEE ALSO
archive_impact(1M), bootsys(1M), instl_adm(1M), instl_bootd(1M), instl_combine(1M), make_sys_image(1M), save_config(1M), instl_adm(4), ignite(5), mkisofs(8).

NAME

make_net_recovery − network based system recovery archive creation

SYNOPSIS

/opt/ignite/bin/make_net_recovery -s *Ignite-UX_server*
[**-a** *archive_server:archive_directory*] [**-A**] [**-b**] [**-d** *tag_string*] [**-f** *content_file*] [**-i**|**-ib**] [**-l** *LLA*] [**-n** *number_archives*] [**-p**] [**-P** *s*|*w*|*e*] [**-m** *tar*|*cpio*|*pax*] [**-r**] [**-u**] [**-v**] [**-x** *content-options*] [**XToolkit_Options**] [**-?**]

DESCRIPTION

make_net_recovery creates a system recovery archive and stores the archive on the network. The archive created by **make_net_recovery** is specific to the system for which it was created and its identity includes hostname, IP address, networking information, etc. In the event of a root disk failure, the recovery archive may be installed via Ignite-UX to restore the system.

The contents of the system recovery archive will always include all files and directories which are considered essential for bringing up a functional system. This "essential list" is predefined by **make_net_recovery**. By running **make_net_recovery** in interactive mode, the directories and files which make up the "essential list" may be displayed. In addition to the essential list, data may be included in the archive on a disk/volume group, file, or directory basis. Nonessential files and directories may also be excluded.

Options

make_net_recovery recognizes the following options:

-s *Ignite-UX_server*
Specifies the hostname of the *Ignite-UX server*. The configuration files, defaults and contents files for the client system will be written to the *Ignite-UX server* in **/var/opt/ignite/clients/0x***LLA***/recovery**. The **make_net_recovery** tool will NFS mount the per-client directory to access this information.

-a *archive_server:archive_directories*
Specifies the NFS server and location onto which to store the archive. The archive directory must be NFS exported (see the section **Exporting Archive Directory**), and sufficient disk space is required. The default is the hostname of the Ignite-UX server followed by the directory which holds the archive, e.g., **Server-host:/var/opt/ignite/recovery/archives/***hostname*. The *hostname* is the name of the system being archived. Each **make_net_recovery** client will create a subdirectory named for the client hostname under the specified directory to store the archives.

-A Based on the files that are specified for inclusion, this option determines which disk(s) and/or volume group(s) contain those specified files, and includes all files from those disk(s) and/or volume group(s) in the archive.

-b When used in combination with the **-i** option, causes **make_net_recovery** to run in the background after the interactive user interface (UI) completes.

-d *tag_string*
One line *tag_string* for the system recovery archive. If the *tag_string* includes spaces, it must be enclosed in quotation marks. The *tag_string* will be displayed when choosing the archive as a configuration from the Ignite-UX interactive user interface. The default *tag_string* is **Recovery Archive**. The *tag_string* is specified in the **INDEX** file as the value for the **cfg** keyword, and must not exceed 80 characters.

-f *content_file*
Location of the file which identifies keywords to specify inclusions and exclusions for the archive. The default is **/var/opt/ignite/clients/0x***LLA***/recovery/archive_content**. This

default file is located on the Ignite-UX server and accessed by the client through an NFS mount. The absolute path name to the **archive_content** file must be supplied as an argument to the **-f** option. This option may be useful when there is a desire to manage multiple files which specify the content of the archive. The **-f** option is not allowed when using the **-x** or **-A** options to specify the contents of the archive. (See the **Including and Excluding From Archive** section for file format.)

-i Causes **make_net_recovery** to run interactively to allow selection of files and directories that are to be included in the recovery archive. The options **-x**, **-A** and **-f** are not allowed with **-i.** It is preferable to use the **ignite** GUI menu command on the Ignite-UX server when running an interactive **make_net_recovery** session. Running it from **ignite** ensures that any server configuration of NFS mounts is already done. It also provides a better progress report and an easier to use interface.

-l *LLA*

The LLA (link-level address) of the system being archived. Used to create the per-client directory on the Ignite-UX server.

-n *number_archives*

Specifies the number of archives that should remain on the server at any given time. The default is two (2). If *number_archives* is two and there are already two archives present when a third is being created, **make_net_recovery** will remove the oldest archive after successfully creating the newest archive.

-p Previews the processing that would take place without actually creating the archive. This is a way of verifying the directory **/var/opt/ignite/recovery/latest** (which is linked to the latest archive directory of the form **/var/opt/ignite/recovery/***date*,*time*) on the local (client) system. To access this directory on a specified Ignite-UX server, the directory is located at **/var/opt/ignite/clients/0x***LLA***/recovery/***date*,*time*. The directory contains the files **archive_cfg**, **control_cfg**, and **system_cfg** that were created with the configuration information desired. It also contains the file **flist** that lists the files that make up the archive. It is best not to modify this file. However, it may be edited to exclude some files/directories from the archive by deleting the entire line, if desired. Only files or directories that are known to be user created should be deleted. The files that end in **_cfg** contain configuration information that may be changed. For example, converting from HFS to VxFS. No further checks are done by **make_net_recovery**. The creation of the System Recovery Archive may then be resumed using the **-r** option.

-P *s*|*w*|*e*

When a disk or volume group is partially included in the system recovery archive, generate an ERROR (e), WARNING (w), or SUPPRESS (s) any warning messages that would normally be generated when partial inclusions occur. The default is **w**, causing WARNING messages to be produced when partial inclusions of disks and/or volume groups are detected. When **e** is specified, an error message will be displayed to both stdout and to the log file, and execution of **make_net_recovery** will stop once the error message is displayed.

-m *tar*|*cpio*|*pax*

Specify in which format (**tar**, **pax**, or **cpio**) the files/directories image in the archive will be stored. **tar**, **pax**, or **cpio** may be specified. If this option is not specified, **tar** is the default format.

-r Resumes creation of the System Recovery Archive after the **-p** option was used to create the **/var/opt/ignite/recovery/latest** directory, and its ***_cfg** files have possibly been edited. If the **-A** and **-r** options are both used, the **-A** option will be ignored, since the file list, **flist**, and other configuration files, ***_cfg**, have been created and possibly modified.

-u Updates the Ignite-UX software from the Ignite server specified by the **-s** option. This is done only when the version of software on the server is newer than the client. The software update uses the depot on the server in the location: **/var/opt/ignite/depots/recovery_cmds**. If this depot does not exist, you may use the **/opt/ignite/lbin/pkg_rec_depot** command to create it. When the **-u** option causes the software to be updated, it then automatically restarts the command with the same options.

-v Display verbose progress messages while creating the system recovery archive. Includes information such as which volume groups/disks will be included in the system recovery archive.

-x include=*file*|*directory*
Includes the *file* or *directory* in the recovery archive, but does not cross any mount points. Note, file names may NOT end with a space.

-x inc_cross=*file*|*directory*
Includes the *file* or *directory* in the recovery archive and crosses mount points to access any directories that are mounted or files contained in directories that are mounted. This option is for crossing local files system mounts only; not remote file system mounts. Note, file names may NOT end with a space.

-x inc_entire=*disk*|*vg_name*
Includes all file systems contained on the specified disk or volume group. Use a block device file (e.g., "/dev/dsk/c0t5d0") when specifying a whole-disk (non-volume manager) file system. Use the volume group name (such as **vg00**) when you want all file systems that are part of that LVM volume group to be included in the archive.

-x exclude=*file*|*directory*
Excludes the *file* or *directory* from the archive. When a directory is specified, no files beneath that directory will be stored in the archive. If the excluded directory is an unmounted file system shown in the **/etc/fstab** file, a WARNING ("Filesystem xxx is not mounted. It will be ignored.") message will be displayed.

Note, file names may NOT end with a space.

-x print_manifest_args=<*print_manifest arguments*>
Passes the given <*print_manifest arguments*> to the **print_manifest** command. This can be done to reduce the amount of time required by **make_net_recovery.** Please see *print_manifest*(1M) for more details.

XToolkit-Options
The **make_net_recovery** command supports a subset of the standard *X Toolkit* options to control the appearance of the GUI when the **-i** option is specified. The supported options are: **-bg**, **-background**, **-fg**, **-foreground**, **-display**, **-name**, **-xrm**, and **-synchronous**. See the *X*(1) manual entry for a definition of these options.

-? Displays the help screen.

Including and Excluding From Archive

The contents of the archive may be controlled from the contents file (see **-f**). The full path for the contents file is **/var/opt/ignite/clients/0x***LLA***/recovery/archive_content** on the Ignite-UX Server. This file consists of keyword identifiers which specify the inclusion of files, directories, or entire disks and volume groups. The keyword identifiers also instruct **make_net_recovery** whether to follow mount points when creating the system recovery archive. The contents file has the following keyword identifiers:

include *filename* | *directory*
Includes the specified *filename* or *directory* and all subdirectories and associated files. Mount points are not crossed and symbolic links are not followed. Note, file names may NOT end with a space.

inc_cross *filename* | *directory*
Include the specified *filename* or *directory* and all subdirectories and files contained underneath subdirectories. Local mount points are crossed but symbolic links are not followed. Note, file names may NOT end with a space.

inc_entire *volume group* | *disk*
Include the entire specified *volume group* (e.g., "vg00") or *disk* (block device - e.g., "/dev/dsk/c0t5d0"). Do not specify a disk if it is part of a volume group.

inc_all_affected
Is equivalent to using -A option. Based on the files that are specified for inclusion, this option determines which disk(s) and/or volume group(s) contain those specified files, and includes all files from those disk(s) and/or volume group(s) in the archive.

exclude *filename* | *directory*
Exclude the specified *filename* or *directory* and all subdirectories and files contained under the subdirectories. Note, file names may NOT end with a space.

make_net_recovery reads the contents file to generate the list of files that will be used to create the system recovery archive. The contents file may be modified by hand or by running **make_net_recovery** in interactive mode. When modifying the contents file, keep the following points in mind:

- No essential file or directory may be excluded. Exclusions of essential files or directories will be ignored.
- Exclusions take precedence over inclusions. Anything that is both included and excluded will be excluded from the archive.
- The ordering of inclusions and exclusions within the defaults file is not significant.
- File names may NOT end with a space.
- The files and directories under NFS or LOFS mounts will not be archived.

Using Settings From Previous Archive Creation

The defaults file stores input specified by interacting with the **make_net_recovery** GUI. Options are preserved until the next archive is generated by interacting with the GUI. Command-line options will override settings in the defaults file. The full path for the defaults file is **/var/opt/ignite/clients/0x***LLA***/recovery/defaults** on the Ignite-UX server. This directory is accessed via NFS from the client.

defaults file
RECOVERY_LOCATION=*15.1.2.3***:/var/opt/ignite/recovery/archives/***client_name***
RECOVERY_DESCRIPTION=*"Recovery Archive"*
SAVE_NUM_ARCHIVES=*2*
ARCHIVE_TYPE=*tar*

Saving the LIF Area

The LIF area of a disk will be archived and restored if it is different from the default LIF area. This means if either the auto-boot line in the **AUTO** LIF file is not "hpux" or the LIF files in addition to **ISL**, **HPUX**, **LABEL**, and **AUTO** are present, then the LIF files will be copied to **/usr/lib/ignite_bootlif**. These LIF files will be restored to the LIF area unless a LIF file with the same name already exists or the **AUTO** file contains something other than **hpux**.

Using the Recovery Archive

To recover a failed system using the network recovery archive:

- If the client system is being replaced, or the LAN card has changed since **make_net_recovery** was last used, you should manually rename the old client directory prior to starting the recovery. Not doing so will cause a new directory to be created and you will not see the recovery archives created under the old client directory. To rename the client directory, obtain the new LAN address (you can use the **LanAddress boot-ROM** command in the information menu), then use the **mv** command. For example:
  cd /var/opt/ignite/clients
  mv *0x00108305463A 0x0060B0C43AB7*
  If you have already booted the new system, you will need to remove the new client before renaming the old directory. Be careful not to remove the old directory containing the recovery information.
- Boot the system using either a network boot, a tape created using **make_boot_tape**, or using the **bootsys** command if the system is still running.
- Do not interact with ISL.
- Select: **[Install HP-UX]**.
- From the Ignite-UX GUI, select the icon for the client.
- Choose **Install/New Install**.
- Select the recovery configuration to use.

Exporting Archive Directory

The directory used to store the archives must be exported from the archive server to each client. Exporting the archive directory from the system where the archive will be stored enables **make_net_recovery** to create and access the archive via NFS. The archive server by default is the Ignite-UX server but may be changed using the **-a** option to be a different remote server, or even the local client if you want to capture the archives as part of the client's regular backup. Note however, that if the archives are stored on the client itself, they must be put onto a remote server if the client ever needs to be recovered using them.

For security reasons, it is best to export each client-specific archive directory to just the individual client. If the recovery archive creation is initiated from the **ignite** GUI on the Ignite-UX server, and the archive server is the same as the Ignite-UX server, the **/etc/dfs/dfstab** or **/etc/exports** file will be edited automatically so the archive may be stored in the desired location. Otherwise, if **make_net_recovery** is run directly on the client, the following steps are required to be performed on the archive server.

- On the archive server, create a directory for each client to contain the archive of the client's files. It is important that the directory be owned by the user **bin**. Replace *client* in the commands below with the hostname of the client. If you use the **-a** option to **make_net_recovery** to specify an alternate location for the archives, you will need to use that path instead of the default which is shown below.
  mkdir -p /var/opt/ignite/recovery/archives/*client*
  chown bin:bin /var/opt/ignite/recovery/archives/*client*
- For 11.31 or later, edit **/etc/dfs/dfstab** to add an entry for each client. Replace *client* with the client's fully qualified hostname in the example shown:
  share -F nfs -o sec=sys,anon=2,rw=*client*
  /var/opt/ignite/recovery/archives/*client*
  For other releases, edit **/etc/exports** to add an entry for each client. Replace *client* with the client's hostname in the example shown:
  /var/opt/ignite/recovery/archives/*client* **-anon=2,access=***client*
- For 11.31 or later, run the **shareall** command to have the edits to the exports file take effect:
  /usr/sbin/shareall -F nfs
  For other releases, run the **exportfs** command to have the edits to the exports file take effect:
  /usr/sbin/exportfs -av

Networking Features

Two NFS mount points are established on the client by **make_net_recovery**. The **/var/opt/ignite/clients** directory on the Ignite-UX server is mounted to the client system to store configuration files which describe the client configuration and location of the recovery archive. The second mount point is made to the *archive_server:archive_directory* (see the **-a** option) and is used to store the recovery archive of the client system. After successful or unsuccessful completion of the system recovery archive, the NFS mount points are unmounted.

The NFS mount for the archive directory may be exported on a per-client basis. A separate archive directory is used for each client. This allows the NFS export of each directory only to the individual client owning the archive, which provides security.

If the default NFS mount options are unsuitable for your network environment, you can pre-mount those directories with whatever options are necessary. **make_net_recovery** will discover that they are already mounted and not attempt to mount them. For example, you can do the following if the read and write buffers had to be 1KB in size:

```
mount -F nfs -orsize=1024,wsize=1024 archive_server:archive_directory \
  /var/opt/ignite/recovery/arch_mnt
mount -F nfs -orsize=1024,wsize=1024 archive_server:/var/opt/ignite/clients \
  /var/opt/ignite/recovery/client_mnt
```

The actual arguments must match those for the **-a** option if it is used in the first **mount** command.

If the client system does not have the most recent versions of Ignite-UX tools, the Ignite-UX GUI uses **swinstall** to install the **recovery package** which includes all necessary files to perform the recovery.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

make_net_recovery returns the following values:

0 No warnings or failures occurred; the execution completed successfully.

1 A failure occurred.

2 A warning occurred.

EXAMPLES

Since **make_net_recovery** actually runs on the client instead of the server, all examples have the **-s** *server* command-line option. This option is required for the command, so that the client (local system) will know what system to contact as the Ignite-UX server.

Create a system recovery archive by interacting with the Ignite-UX GUI from the Ignite-UX server:

```
export DISPLAY=hostname:0
ignite
```

Create a system recovery archive from the client, using settings from the last invocation of the Ignite-UX GUI:

```
make_net_recovery -s myserver
```

Create a system recovery archive with all the files/directories on the disk(s)/volume group(s) containing the files specified by the default essentials file list **/opt/ignite/recovery/mnr_essentials** or the user-defined version of this file, that replaces this file, **/var/opt/ignite/recovery/mnr_essentials**:

```
make_net_recovery -s myserver -A
```

Create a system recovery archive that includes files from all file systems in the **vg00** volume group:

make_net_recovery -s *myserver* **-x inc_entire=vg00**

Create a system recovery archive that includes all of the **vg00** and **vg01** volume groups, but that excludes the */depots* directory:

make_net_recovery -s *myserver* **-x inc_entire=vg00 -x inc_entire=vg01 -x exclude=***/depots*

Preview the creation of the System Recovery Archive:

make_net_recovery -p

Use the -u option to have the Ignite-UX software automatically updated when needed from the Ignite server:

make_net_recovery -s *myserver* **-A -u**

This example assumes that the **/var/opt/ignite/depots/recovery_cmds** software depot has been created on the Ignite server. This depot is created by running the **ignite** GUI at least once to make a recovery archive of a system or by running **pkg_rec_depot** (see also *pkg_rec_depot*(1M)). Once created, this depot will automatically be updated each time Ignite-UX is updated on the server.

Use the -x option to pass arguments to the **print_manifest** command. In this example the -d option will be passed so that disk capacity collection via **diskinfo** is skipped to improve performance. Also the -e option will be passed which causes the output file to be printed with HP-PCL3 control codes for enhanced printer output.

make_net_recovery -s *myserver* **-A -x** *print_manifest_args="-de"*

WARNINGS

General Backup/Recover Not Recommended

The **make_net_recovery** toolset is intended only to create or recover a recovery archive. The recovery archive will include the operating system and a reasonable amount of user data. It is *NOT* intended to be a general purpose backup and restoration tool, and should not be used for that objective.

Creating and Restoring from a Minimal-Recovery Archive

Creating a minimal-recovery archive means creating a recovery archive that contains just enough information to bring a system back up into a minimal-operating state. This allows you to then restore all additional information from a backup created with a general backup/restore utility.

When restoring from a minimal-recovery archive, the boot process will often contain errors due to the missing content. These errors are corrected once the regular system backup is restored and the system is rebooted.

Once the system has been rebooted, you may see the following note:

NOTE: The "/opt/upgrade/bin/tlinstall" command was not part of the system that was installed. It was not possible to run it in order to correctly set the permissions of the "transition" symlinks. You may consider running: "/opt/upgrade/bin/tlinstall -vf" after the system is completely restored.

If software has not yet been installed using SD, the tlinstall command may need to be executed at this time as described above.

Standards May Impose Limits on What May Be Archived

The **pax** command is used to create and recover recovery archives. There maybe limitations in the **pax** command that impose limits on what can or cannot be placed into a recovery archive. Some examples of this are:

- ustar format archives may contain raw uids and gids up to 2097152.
  Because the text user and group name are stored, it may be possible to recover uids and gids larger than 2097152.
- cpio format archives are strictly limited to uids and gids up to 262144.
- ustar format archives cannot contain a file name pointed to by a link that is more than 100 bytes long (required by POSIX.1).
- pax format archives can contain files that are larger then 8 GB, ustar and cpio can not.

Disks Will be Reformatted

If any file from a disk or volume group is included in the recovery archive, that disk (or all disks in the volume group) will be reformatted during the recovery, and only the files included will be recovered. Any files that were not included in the archive, will have to be restored from normal backups.

Disks and volume groups that did not have any files included in the archive are not reformatted during a recovery and are reimported and remounted at the end of the recovery.

Logical Volume Physical Extent Allocation Not Preserved

The **make_net_recovery** tool captures enough information from the system so that during a recovery it may reconstruct all visible aspects of the prior LVM configuration. This includes logical volume and volume group names, attributes, and even minor number values. The tool also ensures that the new logical volumes reside on the same disks within the volume group as they did before.

make_net_recovery does not, however, ensure that logical volumes are extended in the same exact order as they were originally. This means the LVM physical extents allocated to a logical volume may be in a different location on the disk than before. The recovery tools use a very specific and complex algorithm for extending logical volumes to ensure success (such as extending contiguous volumes before non-contiguous). An example effect of this is that swap/dump volumes will reside on the root disk ahead of some other volumes even though that may not have been the original layout.

Logical Volume Distributed Allocation Policy Not Preserved

If logical volumes that are part of the volume groups being archived were configured using the distributed allocation policy (also known as "extent based stripes"), those volumes will be re-created during a recovery with this policy turned off.

VxVM Disk Groups

The root-disk groups managed by VERITAS Volume Manager (VxVM) may be included in the Ignite-UX archive since the B.3.8 release. However, prior to B.3.8 release,the VxVM disk groups cannot be included in the Ignite-UX archive. If they are included, **make_net_recovery** will error. Those disk groups will be left undisturbed and reintegrated into the system after the recovery is complete.

LVM Disk Mirrors Not Restored

The **make_net_recovery** tool will create a recovery backup for a system with mirrored disks but it will not restore the mirrored disk configuration. If the system is later recovered, previously mirrored volumes will no longer be mirrored. They may be manually remirrored after the system is up. Using the **config.local** file in the clients directory, you may specify the LVM commands to restore mirrored disks to be executed automatically after the system has been restored. For more details, see the **/opt/ignite/share/doc/diskmirror.pdf** white paper.

File System Volume Size May Be Modified

The file system volume size(s) in the recovery archive may be modified when the archive is installed. By default, Ignite-UX will ensure 10% free space for each volume and modify the file system volume size accordingly.

If you do not want Ignite-UX to modify the file system volume size(s) automatically, add:

init _hp_ignore_sw_impact=1

to the **/var/opt/ignite/clients/0x***LLA***/recovery/latest/system_cfg** file.

Warning: Setting **_hp_ignite_sw_impact** to 1 may cause recoveries to fail if any file system sizes are reduced below the minimum required.

Note that changes to the system_cfg file must be done after a preview has been created (-p). After making the above changes, resume the creation of the tape recovery archive (-r).

File Names with Non-Printable Characters

Although permissible within HP-UX, it is inadvisable to use characters that do not have a printable graphic on the hardware you commonly use, or that are likely to confuse your terminal. Filenames with these characters cause a warning message to be displayed by **make_net_recovery.** In addition, files that contain these non-printable characters are not included in the archive.

Running Commands in Single User Mode

HP recommends that the **make_tape_recovery** and **make_net_recovery** commands be run in default mode of multi-user (run level 3 or 4 depending on how the system is configured). However, HP does support execution of **make_tape_recovery** in single-user mode using the supported method documented in Ignite-UX FAQs. (See Section 11, Network/Tape Recovery, Question 23.) A copy of the Ignite-UX FAQs is installed with Ignite-UX in **/opt/ignite/share/doc/FAQ**, and the latest version of the Ignite-UX FAQ may be obtained by sending an email to **iux_faq@hpfcdn.fc.hp.com**. HP does not recommend running **make_net_recovery** in single-user mode.

Non-Responding NFS Servers

The **make_tape_recovery** and **make_net_recovery** tools can handle non-responding automounted file systems accessed via indirect maps or the -hosts map without hanging. However, the tools will hang if there are automounted file systems accessed via direct maps or directly mounted NFS file systems.

Unmounted File Systems in /etc/fstab file

If unmounted file system in **/etc/fstab** is detected, a WARNING message will be displayed and the **make_tape_recovery** and **make_net_recovery** will complete with return code 2.

Data Recovery from Raw Logical Volumes

Only file system data is included in recovery archives created by **make_net_recovery**. Data included in raw logical volumes (like those that contain database data) is not included in recovery archives, and must be backed up as part of the overall backup stategy of a system.

Auto Port Aggregate (APA) Cloning Limitation

If a system has an auto port aggregate (APA) configured, Ignite-UX does not support cloning using that recovery archive. It is only supported to recover the archive to the system on which it was created. This assumes that the lan interfaces are still configured for APA, on the switch they are connected to, in exactly the same way as when the archive was created.

DEPENDENCIES

The Ignite-UX GUI must be run from the Ignite-UX Server, see *ignite*(5). **make_net_recovery** depends on several other Ignite-UX tools. When running the Ignite-UX server GUI, Ignite-UX checks whether the client system that **make_net_recovery** runs on has the same versions of Ignite-UX tools.

If running **make_net_recovery** from the command line without ever interacting with the Ignite-UX GUI, commands will need to be installed using *swinstall*(1M) from the Ignite-UX server to the client system on which **make_net_recovery** will be run.

make_net_recovery requires the following filesets of the **Ignite-UX** product be installed on the system:

```
Ignite-UX.MGMT-TOOLS
Ignite-UX.RECOVERY
```

AUTHOR

Ignite-UX and **make_net_recovery** were developed by the Hewlett-Packard Company.

DIAGNOSTICS

All major steps within network recovery are logged on the server and displayed via the Ignite-UX Server GUI.

FILES

/opt/ignite/recovery/mnr_essentials
Lists the files and directories that are considered essential and are always included in the archive if they exist on the system.

/var/opt/ignite/recovery/mnr_essentials
Lists the files and directories that are essential, but acts as the user modifiable version so that the original mnr_essentials file may be maintained. When this file exists, its content is checked before the file **/opt/ignite/recovery/mnr_essentials**.

/var/opt/ignite/clients/0x*LLA***/CINDEX**
This file contains a list of Ignite-UX configurations that are specific to the particular client with a network LLA as shown in the path. This file supplies Ignite-UX with the configuration files created by **make_net_recovery** and provides a list of client-specific selections from which a user may choose during a system recovery.

/var/opt/ignite/clients/0x*LLA***/recovery**
The per-clients recovery directory. It holds the client's recovery configuration, log, and status files as described below.

/var/opt/ignite/clients/0x*LLA***/recovery/archive_content**
Supplies files and directories to be included or excluded. Using the **-x** command line arguments will cause this file to be ignored.

/var/opt/ignite/clients/0x*LLA***/recovery/latest**
A symlink to the *date,time* directory containing the newest set of recovery files as described below.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***
Directory containing files pertaining to the **make_net_recovery** command that was run at the date and time indicated in the directory name. An example path looks like:

/var/opt/ignite/clients/0x*080009123456***/***2000-12-20,13:50***

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/system_cfg**
Configuration file which describes the file system and networking configuration of the system (generated by the *save_config*(1M) command).

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/archive_cfg**
Configuration file which supplies the location and access method to the archive containing the files to be restored.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/control_cfg**
Configuration file which supplies control parameters and the command scripts to import volume groups that will be preserved and not created during the recovery.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/flist**
"File" list file which supplies the list of files that are to be archived. This is a plain text file, but should not be modified. Each line is in a special format that, if altered, could cause problems with the archive process.

/var/opt/ignite/clients/0x*LLA***/recovery/config.local**
An optional configuration file that the user may create to add configuration information to be used during the recovery of the client. For example, you may want to add to this file to a **post_config_cmd** to remirror disks that the recovery process unmirrored. See the document **/opt/ignite/share/doc/diskmirror.pdf** for an example. Once this file is created, **make_net_recovery** will automatically add it to any new configurations that it adds to the **CINDEX** file.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/recovery.log**
Default log file location for **make_net_recovery**.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/manifest**
Software and hardware manifest information installed and configured for the system at the time the archive was created. See *print_manifest*(1M).

/var/opt/ignite/recovery/archives/*hostname*
The default location on the Ignite-UX server for the client to store the recovery archive. The *hostname* directory must be NFS exported to the individual client with the matching hostname.

/var/opt/ignite/clients/0x*LLA***/recovery/defaults**
Supplies the default options to **make_net_recovery**. Created when run interactively using the **-i** option or by using the **ignite** GUI.

/var/opt/ignite/clients/0x*LLA***/recovery/client_status**
File used to communicate the status of the **make_net_recovery** command back to the **ignite** GUI running on the Ignite-UX server.

SEE ALSO

make_boot_tape(1M), make_medialif(1M), make_tape_recovery(1M), manage_index(1M), pkg_rec_depot(1M), print_manifest(1M), save_config(1M), swinstall(1M), instl_adm(4), ignite(5).

NAME

make_sys_image − create compressed file archive of a running system

SYNOPSIS

/opt/ignite/data/scripts/make_sys_image -s *IP | hostname | local*
[**-d** *directory | device*] [**-f** *file*] [**-l** *1 | 2*] [**-n** *filename*] [**-r** *remsh_user*] [**-m** *c | t | p*] [**-c** *z | g | n*] [**-a** *archive_size*] [**-R**] [**-g** *file_list*] [**-t** *y|n|o|p*] [**-i**] [**-p**] [**-u**] [**-v**] [**-w** *log_file*] [**-x**] [**-L**] [**-?**]

DESCRIPTION

make_sys_image is used to create a system archive of a running single-user or multi-user system. This archive may be in **cpio, pax,** or **tar** format, and may be compressed with either **compress** or **gzip**. **make_sys_image** may be called as a **post_config_script** by Ignite-UX as part of the installation process, or invoked at the command line on a running system. The archive may be written to a file, a tape, or a raw disk device on a remote or the local system. If the archive destination is a remote system, that system must have a **.rhosts** entry for the local system, or an **NFS** mount available. The options for this command may be a combination of command line arguments and environment variables passed in with the Ignite-UX configuration file via the **env_vars** keyword.

The archives created by **make_sys_image** are designed for use with Ignite-UX as part of an archive-based ignition process. Before the archive is used for installing systems, the file system impacts of the archive must be obtained. See *archive_impact*(1M). **make_sys_image** does not capture file system or disk layout information. Use **save_config** to capture this information. For details on integrating the script into the GUI, see *manage_index*(1M) for **/var/opt/ignite/INDEX**. For details on using the script non-interactively, see *instl_adm*(4) for information on the **post_config_script** and **env_vars** keywords.

Options

The **make_sys_image** command recognizes the following options:

-s *IP|hostname|local*
IP address or resolvable hostname of the server to send the archive to or *local* if the destination is a local file system, **NFS** mount, disk or tape. If the destination is not local, permission to remotely access the server is required. If a hostname is used in this case, it must be resolvable through whatever means the name service switch is set to use (see *switch*(4) for more information).

-d *directory|device*
Destination directory (for example, */var/tmp/foo*) or device file name (for example, **/dev/rmt/0m**). The default is **/var/tmp/**.

Note: Archives written to tape or raw disk cannot be used directly; they must first be extracted using **dd** onto a file system where they may be accessed by an Ignite-UX server. To use a directory that is an **NFS** mount point on the local system, use **-s** *local* to specify the server destination.

-f *file*
File contains a user-defined list of files to be reset to the **newconfig** state or ignored altogether. This list may be in addition to the default level **2** behaviors, or it may replace the default level **2** reset and/or ignore behaviors.

The *file* may have one or two sections headed by the keywords **RESET** or **NO_ARCHIVE**. Following each keyword are the paths of files or directories (one per line), or regular expressions that describe the files or directories. Each keyword may be combined with the **ONLY** keyword. If the **ONLY** keyword is used, the files in that section replace the default level **2** behavior. The keyword lines must begin with a '+', and each keyword (except **ONLY**) must occur only once in the file. The keywords are case sensitive and must be in all uppercase letters. This option overrides the **-l** option and sets the level to **2**.

-l *1|2* Level of system identity clean-up:

1 None (overridden to **2** if the **-f** option is used).

2 Reset network and login information. Exclude device files, log files, and contents of **/stand** except **/stand/system** and **/stand/kernrel**. Remove hardware specific drivers from the **/stand/system** that gets archived. Files reset to **newconfig** state:

/.profile
/etc/rc.config.d/hpetherconf
/etc/rc.config.d/hpfcgsc_lanconf
/etc/rc.config.d/hpfcmsconf
/etc/rc.config.d/netconf
/etc/rc.config.d/netdaemons
/etc/rc.config.d/namesvrs
/etc/rc.config.d/mailservs
/etc/rc.config.d/xfs
/etc/hosts
/etc/mail/sendmail.cw
/etc/ntp.conf
/etc/vue/config
/var/adm/sw/security

The default level is **2**. If any of the above is a directory, then all files below that directory will be reset. However permissions will be preserved as to what the original file had in those cases where the original file permissions differ from the newconfig version.

-n *filename* The *filename* of the archive file other than the default host name. If the **-d** option is a device file, the **-n** option is ignored.

Note: When the **-n** option is used, a **.Z** or **.gz** is not automatically appended to the file name; it must be added manually.

-r *remsh_user* remsh user name. This is the user name used for remsh commands. If this option is omitted, the current user name is used.

-m *c | t | p* Archive method: **c** for **cpio, p** for **pax,** and **t** for **tar**. **tar** is the default.

-c *z | g | n* Compression method: **z** for **compress**, **g** for **gzip**, or **n** for no compression. **gzip** is the default.

-a *archive_size* The calculated size required for the archive may be passed to this command. The archive size must be a decimal number. The archive size is usually created by running the **/opt/ignite/lbin/list_expander -s** command. This will save one call to **list_expander** during **make_sys_image**'s execution.

-R Run make_sys_image in recovery mode. This will set the level of clean up to **1**, overriding the **-l** option value. This option is required if using the **-g** option.

-g *file_list* The indicated file contains a list of files to be archived. This list should be generated by a call to the **list_expander** command. The file may either contain the name of the file or directory only, or it may contain an entire line of information in the form of: [<**fl_name_len**>] **<file_name>** [**<mode> <last-mod> <bytes> <blocks>** [**<cksum>**]]. **The -R** option must precede the **-g** option.

-t *y|n|o|p* Test for the existence of a needed pax patch.

y Yes, run the tests along with make_sys_image (this is the default).

n No, do not run the tests. This may be used on clients that the user knows contain the patch.

o Run the tests only; that is, do not make a system image. It always exits and returns a **4** if a patch exists that fixes a pax defect; a **3** if it is an enablement patch (for example PAX-Enh on B.11.23); a **1** if the requested format is not allowed on this system and a **0** if there are no pax patch(es) that the system requires or could use.

p Run the tests only; prompt for cancel if a pax patch is required or could be applied to the system. It always exits after the prompt, and returns the same values as the **o** option exception if the patch(es) listed are non-enablement patches it will return a **3** if the user answered [**yY**] to cancel.

-i Remove the bundle definition for the English language. This is useful only for creating archives for use in multi-language or non-English installations where the language specific parts of the core operating system are coming from another source.

-p Preview mode; performs all of the checks and creates the command line, but does not modify any files other than the **make_sys_image.log**.

-u Check the archive destination for sufficient storage capacity. The capacity of raw devices is not easily checked, so this option has no effect in the case where a raw device is the archive destination. The archive size is an approximation and is generally 5% larger than the final archive. The default is to not check capacity.

-v Verbose mode; writes all of the files that are being modified or removed and the command line that will create the system archive to **stdout**.

-w *log_file* Write to the specified log file.

-x Print the list of files that will be set to the **newconfig** state and the files that will be explicitly excluded using level **2** to both **stdout** and **/tmp/excluded_files**. This doesn't include files excluded because they are sockets, named pipes, the local archive destination or on remote file systems. If the **-f** option is used, the output will reflect the effects of that file. With this option selected, no archiving action is taken.

-L Create a copy of any non-standard LIF files (as defined for both whole and LVM disks) found on the boot disk. The boot disk that is used is taken from the first non-commented line in the **/stand/bootconf** file. The LIF volume for storing these LIF files is **/usr/lib/ignite_bootlif**, and is created only if non-standard LIF files are present. This option creates only the backup file. Without this option, the backup, if needed, is created and the recovery media is created with this information stored for later recovery. For more information on the disks, LIF files, and LIF volume creation, see *bootconf*(4), *lif*(4), *mkboot*(1M), and *lifinit*(1). The main purpose of this option is to allow LIF diagnostics and any changes to the **AUTO** LIF file to be saved by the Ignite-UX recovery commands (see *make_net_recovery*(1M)).

-? Display the help screen.

NOTE: If all of the defaults are used, a **gzip** compressed **tar** archive of the system with all of the host and network information removed will be placed on the Ignite-UX server in */var/tmp/myhost.gz*.

Environment Variables

Environment variables are to be on separate lines following the **env_vars+=** keyword in the Ignite-UX configuration file, ***INSTALLFS**.

SERVER=*IP*|*local*	Equivalent to the **-s** option.
DEST_DIR=*directory*|*device*	Equivalent to the **-d** option.
ARCHIVE_NAME=*archive_filename*	Equivalent to the **-n** option.
METHOD=*c*|*t*	Equivalent to the **-m** option.
COMPRESS=*z*|*g*	Equivalent to the **-c** option.
CLEAN_LEVEL=*1*|*2*	Equivalent to the **-l** option.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Use all of the built-in defaults, and create a **gzip** compressed **tar** archive named *hostname.gz* on the Ignite-UX server in directory **/var/tmp** with all identity information removed. This assumes the server has a **.rhosts** entry for the local system:

post_config_script+="make_sys_image"

Create a **gzip** compressed **tar** archive with all of the identity information intact and **dd** to local tape:

post_config_script+="make_sys_image -s *local* **-l** *1* **-d** */dev/rmt/0m***"**

Extract into **/var/tmp** a **gzip**'d **tar** archive from tape:

cd /var/tmp
dd ibs=*10k* **if=***/dev/rmt/0m* **of=***archive.gz*

Note: Replace **10k** with **5k** for **cpio** archives. Replace **.gz** with **.Z** for **compress**'d archives.

Create a local archive resetting **/etc/motd** and **/etc/issue** to **newconfig** state along with the level **2** default resets and ignore only the directory */data_area* and files or directories beginning with "n" or "a" in **/opt/apps**.

Copy **/opt/ignite/data/scripts/make_sys_image** from an Ignite-UX server to **/tmp** on the archive system:

/tmp/make_sys_image -s *local* **-d** */var/tmp* **-f** */tmp/specific_files*

Contents of */tmp/specific_files*:

 # Files to be reset to newconfig state
 # in addition to the defaults. Note use of upper case.

+ RESET
/etc/motd
/etc/issue

 # Files to be excluded from the archive,
 # override the defaults. Note use of upper case.

+ ONLY NO_ARCHIVE
/data_area
/opt/apps/[na]*

Generate a list of files that will be reset or ignored including those specified in */tmp/specific_files*:

/tmp/make_sys_image -s *local* **-x -f** */tmp/specific_files*

Create a **compress** compressed **cpio** archive to a remote system other than the Ignite-UX server, and

remove the host identity and general network information:

post_config_script+="make_sys_image -s *15.15.15.99* **-l** *2* **-c** *z* **-m** *c***"**

Create an Ignite-UX configuration file that specifies a **cpio** archive and **compress** it to the Ignite-UX server, default destination directory, and default file name:

 # instl_adm defaults:
server="15.1.50.74"
netmask[]="0xfffff800"
route_gateway[0]="15.1.48.1"
route_destination[0]="default"
 # end instl_adm defaults.
env_vars+="INST_ALLOW_WARNINGS=1
METHOD=c
COMPRESS=z"
kbdlang="PS2_DIN_US_English"

WARNINGS

Because of the impact **make_sys_image** can have on a system when running at level 2, it is recommended that no applications be running and no normal users be logged onto the system while it is being archived. If you do not stop all applications while **make_sys_image** is running at level 2, the system may experience application outages because of the configuration changes made temporarily to the system. You should consider rebooting the system after running **make_sys_image** at level 2 in case the configuration changes have impacted any software you did not halt during the archival process.

If **/etc/hosts** is being used for hostname resolution, then the **-s** argument cannot be a hostname. It must be an IP address in this case.

SEE ALSO

compress(1), cpio(1), dd(1), gzip(1), pax(1), rcp(1), tar(1), archive_impact(1M), instl_adm(1M), make_config(1M), make_medialif(1M), manage_index(1M), save_config(1M), instl_adm(4), .rhosts(4), ignite(5), nfs(7).

NAME

make_tape_recovery − tape based system recovery archive creation

SYNOPSIS

/opt/ignite/bin/make_tape_recovery [**-s** *Ignite-UX_server*]
[**-a** *tape_drive*] [**-A**] [**-b**] [**-B** *boot_destination_file*]
[**-D** *tape_volume_name]*
[**-d** *tag_string*] [**-f** *content_file*] [**-i**|**-ib**] [**-I**] [**-l** *LLA*]
[**-n** *number_cfg_directories*] [**-p**] [**-P** *s*|*w*|*e*] [**-m** *tar*|*cpio*|*pax*] [**-r**]
[**-t** *tape_title_string*] [**-u**] [**-v**] [**-x** *content-options*] [**XToolkit_Options**] [**-?**]

DESCRIPTION

make_tape_recovery creates a system recovery archive and stores the archive on a local tape. **make_tape_recovery** is capable of creating system recovery tapes for all tape devices supported by HP-UX systems. On PA-RISC systems the command has the ability to span multiple tapes. On Integrity systems all content must fit on one tape. If multiple tapes are needed to create an entire recovery archive, the make_tape_recovery command must be invoked from a terminal. This is caused by the pax command failing to prompt for a tape change when invoked from the ignite GUI. The archive created by **make_tape_recovery** is specific to the system for which it was created and its identity includes hostname, IP address, networking information, etc. In the event of a root disk failure, the recovery archive may be installed via tape to restore the system.

The contents of the system recovery archive will always include all files and directories which are considered essential for bringing up a functional system. This "essential list" is predefined by **make_tape_recovery**. By running **make_tape_recovery** in interactive mode, the directories and files which make up the "essential list" may be displayed. In addition to the essential list, data may be included in the archive on a disk/volume group, file, or directory basis. Nonessential files and directories may also be excluded.

Options

make_tape_recovery recognizes the following options:

-s *Ignite-UX_server*
Specifies the hostname of the *Ignite-UX server*. The configuration files, defaults and contents files for the client system will be written to the *Ignite-UX server* in **/var/opt/ignite/clients/0x***LLA***/recovery**. The **make_tape_recovery** tool will NFS mount the per-client directory to access this information.

-a *tape_drive*
Specifies the tape drive device file that will be used for archiving by **make_tape_recovery**. The default is legacy device **/dev/rmt/0mn** unless the **/var/opt/ignite/recovery/default** file exists on the system. The tape device file must be a no-rewind mode device special file. On HP-UX 11i v3, when legacy mode devices are disabled, you can specify the agile device name using the **-a** option.

-A Based on the files that are specified for inclusion, this option determines which disk(s) and/or volume group(s) contain those specified files, and includes all files from those disk(s) and/or volume group(s) in the archive.

-b When used in combination with the **-i** option, causes **make_tape_recovery** to run in the background after the interactive user interface (UI) completes.

-B *boot_destination_file*
Specifies the temporary location where the **LIF** volume will be assembled before it is written to tape. The default file is **/var/tmp/uxinstlf.recovery** for HP9000 systems and **/var/tmp/HPUXIUXLIF** for HP Integrity systems. At least 500 MB is required in **/var/tmp** directory where the **LIF** volume will be assembled. The **LIF** volume and other temporary files required by *make_medialif* (1M) or *make_ipf_tape*(1M) will be assembled in the directory specified and then removed after the recovery process

completes.

-D *tape_volume_name*
Specifies the name of the ANSI tape volume. This option is only for IPF tape. By default, Ignite-UX recovery tool uses the first 6 characters of string **HP** "month" "day" as tape volume name.

-d *tag_string*
One line *tag_string* for the system recovery archive. If the *tag_string* includes spaces, it must be enclosed in quotation marks. The *tag_string* will be displayed when choosing the archive as a configuration from the Ignite-UX interactive user interface. The default *tag_string* is **Recovery Archive**. The *tag_string* is specified in the **INDEX** file as the value for the **cfg** keyword, and must not exceed 80 characters.

-f *content_file*
Location of the file which identifies keywords to specify inclusions and exclusions for the archive. The default is **/var/opt/ignite/recovery/archive_content** on the local system if the **-s** option is not used. However, if the **-s** option is given, the default is **/var/opt/ignite/clients/0x***LLA***/recovery/archive_content**. This default file is located on the Ignite-UX server and accessed by the client through an NFS mount. The absolute path name to the **archive_content** file must be supplied as an argument to the **-f** option. This option may be useful when there is a desire to manage multiple files which specify the content of the archive. The **-f** option is not allowed when using the **-x** or **-A** options to specify the contents of the archive. (See the **Including and Excluding From Archive** section for file format.)

-i Causes **make_tape_recovery** to run interactively to allow selection of files and directories that are to be included in the recovery archive. The options **-x**, **-A** and **-f** are not allowed with **-i.** It is preferable to use the **ignite** GUI menu command on the Ignite-UX server when running an interactive **make_tape_recovery** session. Running it from **ignite** ensures that any server configuration of NFS mounts is already done. It also provides a better progress report and an easier to use interface.

-I Causes the system recovery process to be interactive when booting from the tape. By default, when the systems boots from the recovery tape, it will allow ten (10) seconds to interrupt the automatic recovery process in order to make modifications interactively. When the **-I** option is specified, booting from the tape will always present the interactive menus. This option is useful when making configuration changes during the recovery, and may also help prevent an accidental system recovery from a recovery tape.

-l *LLA*
The LLA (link-level address) of the system being archived. Used to create the per-client directory on the Ignite-UX server.

-n *number_cfg_directories*
Specifies the number of configuration and log file directories to be saved on the system or server. The default is two (2). If *number_cfg_directories* is two and there are already two configuration file directories present when a third is being created, **make_tape_recovery** will remove the oldest directory after successfully creating the newest directory.

-p Previews the processing that would take place without actually creating the tape. This is a way of verifying the directory **/var/opt/ignite/recovery/latest** (which is linked to the latest archive directory of the form **/var/opt/ignite/recovery/***date*,*time*) on the local (client) system. To access this directory on a specified Ignite-UX server, the directory is located at **/var/opt/ignite/clients/0x***LLA***/recovery/***date*,*time*. The directory contains the files **archive_cfg**, **control_cfg**, and **system_cfg** that were created

with the configuration information desired. It also contains the file **flist** that lists the files that make up the archive. It is best not to modify this file. However, it may be edited to exclude some files/directories from the archive by deleting the entire line, if desired. Only files or directories that are known to be user created should be deleted. The files that end in **_cfg** contain configuration information that may be changed. For example, converting from HFS to VxFS. No further checks are done by **make_tape_recovery**. The creation of the System Recovery Tape may then be resumed using the **-r** option.

-P *s|w|e*

When a disk or volume group is partially included in the system recovery archive, generate an ERROR (e), WARNING (w), or SUPPRESS (s) any warning messages that would normally be generated when partial inclusions occur. The default is **w**, causing WARNING messages to be produced when partial inclusions of disks and/or volume groups are detected. When **e** is specified, an error message will be displayed to both stdout and to the log file, and execution of **make_tape_recovery** will stop once the error message is displayed.

-m *tar|cpio|pax*

Specify in which format (**tar**, **pax**, or **cpio**) the files/directories image on the tape will be stored. **tar**, **pax**, or **cpio** may be specified. If this option is not specified, **tar** is the default format.

-r Resumes creation of the System Recovery Tape after the **-p** option was used to create the **/var/opt/ignite/recovery/latest** directory, and its ***_cfg** files have possibly been edited. If the **-A** and **-r** options are both used, the **-A** option will be ignored, since the file list, **flist**, and other configuration files, ***_cfg**, have been created and possibly modified.

-u Updates the Ignite-UX software from the Ignite server specified by the **-s** option. This is done only when the version of software on the server is newer than the client. The software update uses the depot on the server in the location: **/var/opt/ignite/depots/recovery_cmds**. If this depot does not exist, you may use the **/opt/ignite/lbin/pkg_rec_depot** command to create it. When the **-u** option causes the software to be updated, it then automatically restarts the command with the same options.

-t *tape_title_string*

Specifies a custom message that will be displayed to the user during a recovery from the tape. This may be useful in identifying the tape. The default title that will be written to the tape is of the form: "Recovery tape created from system: *hostname* on *date*". This message is only displayed when booting directly from tape. If using a 2-step boot from DVD/CD, then the **-d** option can be used to change the description shown in the user interface.

-v Display verbose progress messages while creating the system recovery archive. Includes information such as which volume groups/disks will be included in the system recovery archive.

-x include=*file|directory*

Includes the *file* or *directory* in the recovery archive, but does not cross any mount points. Note, file names may NOT end with a space.

-x inc_cross=*file|directory*

Includes the *file* or *directory* in the recovery archive and crosses mount points to access any directories that are mounted or files contained in directories that are mounted. This option is for crossing local files system mounts only; not remote file system mounts. Note, file names may NOT end with a space.

-x inc_entire=*disk|vg_name*
Includes all file systems contained on the specified disk or volume group. Use a block device file (e.g., "/dev/dsk/c0t5d0") when specifying a whole-disk (non-volume manager) file system. Use the volume group name (such as **vg00**) when you want all file systems that are part of that LVM volume group to be included in the archive.

-x exclude=*file|directory*
Excludes the *file* or *directory* from the archive. When a directory is specified, no files beneath that directory will be stored in the archive. If the excluded directory is an unmounted file system shown in the **/etc/fstab** file, a WARNING ("Filesystem xxx is not mounted. It will be ignored.") message will be displayed.

Note, file names may NOT end with a space.

-x print_manifest_args=<*print_manifest arguments*>
Passes the given <*print_manifest arguments*> to the **print_manifest** command. This can be done to reduce the amount of time required by **make_tape_recovery.** Please see *print_manifest*(1M) for more details.

XToolkit-Options
The **make_tape_recovery** command supports a subset of the standard *X Toolkit* options to control the appearance of the GUI when the **-i** option is specified. The supported options are: **-bg**, **-background**, **-fg**, **-foreground**, **-display**, **-name**, **-xrm**, and **-synchronous**. See the *X*(1) manual entry for a definition of these options.

-? Displays the help screen.

Including and Excluding From Archive

The contents of the archive may be controlled from the contents file (see **-f**). The full path for the contents file is **/var/opt/ignite/clients/0x***LLA***/recovery/archive_content** on the Ignite-UX Server. This file consists of keyword identifiers which specify the inclusion of files, directories, or entire disks and volume groups. The keyword identifiers also instruct **make_tape_recovery** whether to follow mount points when creating the system recovery archive. The contents file has the following keyword identifiers:

include *filename | directory*
Includes the specified *filename* or *directory* and all subdirectories and associated files. Mount points are not crossed and symbolic links are not followed. Note, file names may NOT end with a space.

inc_cross *filename | directory*
Include the specified *filename* or *directory* and all subdirectories and files contained underneath subdirectories. Local mount points are crossed but symbolic links are not followed. Note, file names may NOT end with a space.

inc_entire *volume group | disk*
Include the entire specified *volume group* (e.g., "vg00") or *disk* (block device - e.g., "/dev/dsk/c0t5d0"). Do not specify a disk if it is part of a volume group.

inc_all_affected
Is equivalent to using -A option. Based on the files that are specified for inclusion, this option determines which disk(s) and/or volume group(s) contain those specified files, and includes all files from those disk(s) and/or volume group(s) in the archive.

exclude *filename | directory*
Exclude the specified *filename* or *directory* and all subdirectories and files contained under the subdirectories. Note, file names may NOT end with a space.

make_tape_recovery reads the contents file to generate the list of files that will be used to create the system recovery archive. The contents file may be modified by hand or by running

make_tape_recovery in interactive mode. When modifying the contents file, keep the following points in mind:

- No essential file or directory may be excluded. Exclusions of essential files or directories will be ignored.
- Exclusions take precedence over inclusions. Anything that is both included and excluded will be excluded from the archive.
- The ordering of inclusions and exclusions within the defaults file is not significant.
- File names may NOT end with a space.
- The files and directories under NFS or LOFS mounts will not be archived.

Using Settings From Previous Archive Creation

The defaults file stores input specified by interacting with the **make_tape_recovery** GUI. Options are preserved until the next archive is generated by interacting with the GUI. Command-line options will override settings in the defaults file. The full path for the defaults file is **/var/opt/ignite/clients/0x***LLA***/recovery/defaults** on the Ignite-UX server (if the **-s** *server option is used). This directory is accessed via NFS from the client. If a server is not used, the full path for the defaults file is* **/var/opt/ignite/recovery/defaults**.

defaults file
RECOVERY_TYPE=tape
RECOVERY_LOCATION=*hostname***:/var/opt/ignite/recovery/archives/***client_name*
TAPE_DESTINATION=/dev/rmt/0mn
RECOVERY_DESCRIPTION=*"Recovery Archive"*
SAVE_NUM_ARCHIVES=*2*
ARCHIVE_TYPE=*tar*

Saving the LIF Area

The LIF area of a disk will be archived and restored if it is different from the default LIF area. This means if either the auto-boot line in the **AUTO** LIF file is not "hpux" or the LIF files in addition to **ISL**, **HPUX**, **LABEL**, and **AUTO** are present, then the LIF files will be copied to **/usr/lib/ignite_bootlif**. These LIF files will be restored to the LIF area unless a LIF file with the same name already exists or the **AUTO** file contains something other than **hpux**.

Format of PA-RISC Recovery Tape

A PA-RISC Recovery Tape has two tape files.

The first tape file holds a LIF (Logical Interchange Format) volume. This LIF volume holds HP-UX boot and Ignite-UX install environment content.

filename	description
ISL	Initial System Loader
AUTO	Autoexecute default boot loader command
INDEX	Ignite INDEX file (references other config files)
CONFIG	Ignite system-specific config file
HPUX	HP-UX boot loader
FWWKAR6	Special firmware functionality for specific systems
FWWKAR7	Special firmware functionality for specific systems
FWWKAR8	Special firmware functionality for specific systems
WINSTALL	Install kernel
WINSTALLFS	Install file system image
INSTCMDS	Archive of commands needed for initial install
SYSCMDS	Archive of commands needed for final install
RECCMDS	Archive of commands needed for expert recovery
SCRIPTS	Scripts used during install post_load
VERSION	Ignite-UX version used to create tape
PAD	Additional data to create full tape block

The second tape file holds the recovery archive. The archive format depends on the options used to create the Recovery Tape.

Format of Integrity Recovery Tape

An Integrity Recovery Tape is formatted using ANSI standard tape labels as defined by ANSI Standard X 3.27. An Integrity Recovery Tape also conforms to the boot tape format specified in the Unified Extensible Firmware Interface (UEFI) 2.0 standard.

The ANSI standard label format requires tape files which hold label records. These label tape files must appear before and after each tape file which holds the content needed to support boot and recovery. Thus, each file needed for boot and recovery becomes three files on tape. The first of these three files holds the file label which includes the file name, block size, and other format details. The first file label on tape also holds the tape volume label. The second of the three files holds the actual boot or recovery content. The third of these three files holds the end of file label which includes information in the file label as well as file length. The last file label on tape also holds the end of volume label.

	tape file		ANSI	
HDR	DATA	EOF	filename	description
1	2	3	EFIBOOTHPUX	UEFI 2.0 boot tape descriptor block
4	5	6	BOOTLOADER	HP-UX boot loader (hpux.efi)
7	8	9	FPWSAEFI	Floating point simulation
10	11	12	AUTO	Autoexecute default boot loader command
13	14	15	IINSTALL	Install kernel
16	17	18	IINSTALLFS	Install file system image
19	20	21	HPUXIUXLIF	LIF volume of install environment content
22	23	(24)	ARCHIVE	Recovery archive

Note that Integrity Recovery tapes may not have a label file at the end of the tape which should hold ARCHIVE end of file and end of volume label records. This file is not present due to the method used to write Recovery Tape archives. A future Ignite-UX release may write this label file.

The LIF volume tape file (HPUXIUXLIF) on an Integrity Recovery Tape includes additional Ignite-UX install environment content.

filename	description
INDEX	Ignite INDEX file (references other config files)
CONFIG	Ignite system-specific config file
IINSTALL	Install kernel
IINSTALLFS	Install file system image
INSTCMDIA	Archive of commands needed for initial install
SYSCMDSIA	Archive of commands needed for final install
RECCMDSIA	Archive of commands needed for expert recovery
SCRIPTS	Scripts used during install post_load
VERSION	Ignite-UX version used to create tape
PAD	Additional data to create full tape block

Using the Recovery Tape to Boot a System

To recover a failed system disk or volume group, you would:

- Insert the System Recovery Tape into the tape drive,
- boot the system,
- interrupt the boot sequence to redirect it to the tape drive,
- choose no interaction with ISL, and
- allow the install process to complete automatically.

Using the Recovery Tape to Clone a System

When creating a recovery tape for the purpose of cloning one system to another, it is best to run **`make_tape_recovery`** using the **`-i`** option. This will allow interaction with the installation and

allow any changes necessary (such as changing which disks to use, hostname, archive format, and IP address). If the **-i** option is not used to create the tape, ten (10) seconds will be given to stop the automatic installation. The procedure below describes this situation.

When cloning systems, it is very important to change the networking information to avoid IP-address conflicts between the original system and the new system. You must interact with the installation to make these changes.

- Insert the System Recovery Tape into the tape drive,
- boot the system, and
- interrupt the boot sequence to redirect it to the tape drive.
- Cancel the non-interactive installation by pressing the return key when the following messages are displayed:

  WARNING: The configuration information calls for a non-interactive installation.

  Press <Return> within 10 seconds to cancel batch-mode installation:

- The "Ignite-UX Welcome" screen will be presented.

  Select the option:
  [Install HP-UX]

  Then select the option:
  [ ] Advanced Installation

- Make any desired changes to the file systems, hostname, IP address, time zone, root user password, DNS server, and gateway information.

- Select **[GO]** to proceed with the installation.

Using a recovery tape to clone systems will only work if the two systems are capable of running the same software configuration. This means that the source system must contain a version of HP-UX with all necessary patches and driver software required by the clone system.

If the two systems are different hardware models, a new kernel will automatically be built to suit the new hardware. If the two systems are the same model, the clone will by default use the kernel from the original system.

You may force a kernel to be built by using the "Additional" dialog in the GUI to set the *"Cloning to different HW?"* selector to *TRUE*.

Extracting Files from a PA-RISC Recovery Tape

After the PA-RISC System Recovery Tape has been created, a single file or files may be extracted from tape by seeking to the tape position where the archive is located. The **mt** command may be used to seek to the appropriate location, and **pax** or **tar** may be used to extract files from the archive.

To extract a single file from a PA-RISC recovery archive:

```
mt -t /dev/rmt/0mn rew
mt -t /dev/rmt/0mn fsf 1
tar -xvf /dev/rmt/0m filename
```

A rewind mode device special file may be used in this case to simplify repeated attempts to list and extract archive content. Extracting files from tape may take a long time, especially when archives are large.

Extracting Files from an Integrity Recovery Tape

After the Integrity System Recovery Tape has been created, a single file or files may be extracted from tape by use of the **ansitape** command. The **ansitape** command may be used to find the appropriate tape file and read archive file content. Standard output from this command may be redirected to the **pax** or **tar** command to read archive content and extract files.

To extract a single file from an Integrity recovery archive:

/opt/ignite/lbin/ansitape -xf mt=/dev/rmt/0m archive |.C " tar -xvf -" *filen*

Extraction of archive content will not start until the **ansitape** command locates the archive file on tape. The **ansitape** command requires use of the **f** option to request standard output to process archive files and other files 2 GiB or larger. A rewind mode device special file may be used in this case to simplify repeated attempts to list and extract archive content. Extracting files from tape may take a long time, especially when archives are large.

It is possible to extract a single file from an Integrity recovery archive without using the **ansitape** command:

mt -t /dev/rmt/0mn rew
mt -t /dev/rmt/0mn fsf 22
tar -xvf /dev/rmt/0m *filename*

However, use of the **ansitape** command is recommended in case the format of Integrity recovery tapes changes in the future.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

make_tape_recovery returns the following values:

0 No warnings or failures occurred; the execution completed successfully.

1 A failure occurred.

2 A warning occurred.

EXAMPLES

There are two lines for all but the first **make_tape_recovery** examples given. For each of these examples, the first line assumes that the **make_tape_recovery** command is being executed on a stand-alone system, thus not needing the **-s** *server option. The second line assumes the use of a server.*

Create a system recovery tape by interacting with the Ignite-UX GUI from the Ignite-UX server:

export DISPLAY=*hostname***:0**
ignite

Create a system recovery tape on the stand-alone system using the standard defaults or from the client, using settings from the last invocation of the Ignite-UX GUI:

make_tape_recovery

make_tape_recovery -s *myserver*

Create a system recovery tape with all the files/directories on the disk(s)/volume group(s) containing the files specified by the default essentials file list **/opt/ignite/recovery/mnr_essentials** or the user-defined version of this file, that replaces this file, **/var/opt/ignite/recovery/mnr_essentials**:

make_tape_recovery -A

make_tape_recovery -s *myserver* **-A**

Create a system recovery tape that includes files from all file systems in the **vg00** volume group:

make_tape_recovery -x inc_entire=vg00

make_tape_recovery -s *myserver* **-x inc_entire=vg00**

Create a system recovery tape that includes all of the **vg00** and **vg01** volume groups, but that excludes the

/depots directory:

make_tape_recovery -x inc_entire=vg00 -x inc_entire=vg01 -x exclude=*/depots*

make_tape_recovery -s *myserver* **-x inc_entire=vg00 -x inc_entire=vg01 -x exclude=***/depots*

Preview the creation of the System Recovery Tape:

make_tape_recovery -p

Create a System Recovery Tape that contains the entire root volume group which will run interactively when used and contains a custom tape title:

make_tape_recovery -x inc_entire=vg00 \ -I -t "*Pre-upgrade recovery tape***"**

Use the -u option to have the Ignite-UX software automatically updated when needed from the Ignite server:

make_tape_recovery -s *myserver* **-a** */dev/rmt/1mn* **-A -u**

This example assumes that the **/var/opt/ignite/depots/recovery_cmds** software depot has been created on the Ignite server. This depot is created by running the **ignite** GUI at least once to make a recovery tape of a system or by running **pkg_rec_depot** (see also *pkg_rec_depot*(1M)). Once created, this depot will automatically be updated each time Ignite-UX is updated on the server, assuming this is not a stand-alone recovery tape creation session:

Use the -x option to pass arguments to the **print_manifest** command. In this example the -d option will be passed so that disk capacity collection via **diskinfo** is skipped to improve performance. Also the -e option will be passed which causes the output file to be printed with HP-PCL3 control codes for enhanced printer output.

make_tape_recovery -s *myserver* **-A -x** *print_manifest_args="-de"*

WARNINGS

General Backup/Recover Not Recommended

The **make_tape_recovery** toolset is intended only to create or recover a recovery archive. The recovery archive will include the operating system and a reasonable amount of user data. It is *NOT* intended to be a general purpose backup and restoration tool, and should not be used for that objective.

Creating and Restoring from a Minimal-Recovery Archive

Creating a minimal-recovery archive means creating a recovery archive that contains just enough information to bring a system back up into a minimal-operating state. This allows you to then restore all additional information from a backup created with a general backup/restore utility.

When restoring from a minimal-recovery archive, the boot process will often contain errors due to the missing content. These errors are corrected once the regular system backup is restored and the system is rebooted.

Once the system has been rebooted, you may see the following note:

NOTE: The "/opt/upgrade/bin/tlinstall" command was not part of the system that was installed. It was not possible to run it in order to correctly set the permissions of the "transition" symlinks. You may consider running: "/opt/upgrade/bin/tlinstall -vf" after the system is completely restored.

If software has not yet been installed using SD, the tlinstall command may need to be executed at this time as described above.

Standards May Impose Limits on What May Be Archived

The **pax** command is used to create and recover recovery archives. There maybe limitations in the **pax** command that impose limits on what can or cannot be placed into a recovery archive. Some examples of this are:

- ustar format archives may contain raw uids and gids up to 2097152.
  Because the text user and group name are stored, it may be possible to recover uids and gids larger than 2097152.
- cpio format archives are strictly limited to uids and gids up to
  262144.
- ustar format archives cannot contain a file name pointed to by a
  link that is more than 100 bytes long (required by POSIX.1).
- pax format archives can contain files that are larger then 8 GB,
  ustar and cpio can not.

Disks Will be Reformatted

If any file from a disk or volume group is included in the recovery archive, that disk (or all disks in the volume group) will be reformatted during the recovery, and only the files included will be recovered. Any files that were not included in the archive, will have to be restored from normal backups.

Disks and volume groups that did not have any files included in the archive are not reformatted during a recovery and are reimported and remounted at the end of the recovery.

Logical Volume Physical Extent Allocation Not Preserved

The **make_tape_recovery** tool captures enough information from the system so that during a recovery it may reconstruct all visible aspects of the prior LVM configuration. This includes logical volume and volume group names, attributes, and even minor number values. The tool also ensures that the new logical volumes reside on the same disks within the volume group as they did before.

make_tape_recovery does not, however, ensure that logical volumes are extended in the same exact order as they were originally. This means the LVM physical extents allocated to a logical volume may be in a different location on the disk than before. The recovery tools use a very specific and complex algorithm for extending logical volumes to ensure success (such as extending contiguous volumes before non-contiguous). An example effect of this is that swap/dump volumes will reside on the root disk ahead of some other volumes even though that may not have been the original layout.

Logical Volume Distributed Allocation Policy Not Preserved

If logical volumes that are part of the volume groups being archived were configured using the distributed allocation policy (also known as "extent based stripes"), those volumes will be re-created during a recovery with this policy turned off.

VxVM Disk Groups

The root-disk groups managed by VERITAS Volume Manager (VxVM) may be included in the Ignite-UX archive since the B.3.8 release. However, prior to B.3.8 release,the VxVM disk groups cannot be included in the Ignite-UX archive. If they are included, **make_tape_recovery** will error. Those disk groups will be left undisturbed and reintegrated into the system after the recovery is complete.

LVM Disk Mirrors Not Restored

The **make_tape_recovery** tool will create a recovery backup for a system with mirrored disks but it will not restore the mirrored disk configuration. If the system is later recovered, previously mirrored volumes will no longer be mirrored. They may be manually remirrored after the system is up. Using the **config.local** file in the clients directory, you may specify the LVM commands to restore mirrored disks to be executed automatically after the system has been restored. For more details, see the **/opt/ignite/share/doc/diskmirror.pdf** white paper.

File System Volume Size May Be Modified

The file system volume size(s) in the recovery archive may be modified when the archive is installed. By default, Ignite-UX will ensure 10% free space for each volume and modify the file system volume size accordingly.

If you do not want Ignite-UX to modify the file system volume size(s) automatically, add:

init _hp_ignore_sw_impact=1

to the **/var/opt/ignite/recovery/latest/system_cfg** file.

Warning: Setting **_hp_ignite_sw_impact** to 1 may cause recoveries to fail if any file system sizes are reduced below the minimum required.

Note that changes to the system_cfg file must be done after a preview has been created (-p). After making the above changes, resume the creation of the network recovery archive (-r).

File Names with Non-Printable Characters

Although permissible within HP-UX, it is inadvisable to use characters that do not have a printable graphic on the hardware you commonly use, or that are likely to confuse your terminal. Filenames with these characters cause a warning message to be displayed by **make_tape_recovery.** In addition, files that contain these non-printable characters are not included in the archive.

Running Commands in Single User Mode

HP recommends that the **make_tape_recovery** and **make_net_recovery** commands be run in default mode of multi-user (run level 3 or 4 depending on how the system is configured). However, HP does support execution of **make_tape_recovery** in single-user mode using the supported method documented in Ignite-UX FAQs. (See Section 11, Network/Tape Recovery, Question 23.) A copy of the Ignite-UX FAQs is installed with Ignite-UX in **/opt/ignite/share/doc/FAQ**, and the latest version of the Ignite-UX FAQ may be obtained by sending an email to **iux_faq@hpfcdn.fc.hp.com**. HP does not recommend running **make_net_recovery** in single-user mode.

Non-Responding NFS Servers

The **make_tape_recovery** and **make_net_recovery** tools can handle non-responding automounted file systems accessed via indirect maps or the -hosts map without hanging. However, the tools will hang if there are automounted file systems accessed via direct maps or directly mounted NFS file systems.

Unmounted File Systems in /etc/fstab file

If unmounted file system in **/etc/fstab** is detected, a WARNING message will be displayed and the **make_tape_recovery** and **make_net_recovery** will complete with return code 2.

Data Recovery from Raw Logical Volumes

Only file system data is included in recovery archives created by **make_tape_recovery**. Data included in raw logical volumes (like those that contain database data) is not included in recovery archives, and must be backed up as part of the overall backup stategy of a system.

Auto Port Aggregate (APA) Cloning Limitation

If a system has an auto port aggregate (APA) configured, Ignite-UX does not support cloning using that recovery archive. It is only supported to recover the archive to the system on which it was created. This assumes that the lan interfaces are still configured for APA, on the switch they are connected to, in exactly the same way as when the archive was created.

DEPENDENCIES

The Ignite-UX GUI must be run from the Ignite-UX Server, see *ignite*(5). **make_tape_recovery** depends on several other Ignite-UX tools. When running the Ignite-UX server GUI, Ignite-UX checks whether the client system that **make_tape_recovery** runs on has the same versions of Ignite-UX tools.

If running **make_tape_recovery** from the command line without ever interacting with the Ignite-UX GUI, commands will need to be installed using *swinstall*(1M) from the Ignite-UX server to the client system on which **make_tape_recovery** will be run.

make_tape_recovery requires the following filesets of the **Ignite-UX** product be installed on the system:

Ignite-UX.RECOVERY
Ignite-UX.BOOT-KERNEL
Ignite-UX.FILE-SRV-*release*

Ignite-UX.MGMT-TOOLS

AUTHOR

Ignite-UX and **make_tape_recovery** were developed by the Hewlett-Packard Company.

DIAGNOSTICS

All major steps within network recovery are logged on the server and displayed via the Ignite-UX Server GUI.

FILES

make_tape_recovery
always stores the archive on the local tape but may store its configuration, log, and status files either locally or on an Ignite-UX server specified by the **-s** *server option. If* **make_tape_recovery** stores its configuration files locally, the directory **/var/opt/ignite/recovery** is used to store the following files. If **make_tape_recovery** uses the server to store the files, the **/var/opt/ignite/clients/0x***LLA***/recovery** directory will be used. The following list is the configuration directory and files as found on the server. To have a full list of configuration, log, and status files on the local system, substitute the local directory for the server directory.

/opt/ignite/recovery/mnr_essentials
Lists the files and directories that are considered essential and are always included in the archive if they exist on the system.

/var/opt/ignite/recovery/mnr_essentials
Lists the files and directories that are essential, but acts as the user modifiable version so that the original mnr_essentials file may be maintained. When this file exists, its content is checked before the file **/opt/ignite/recovery/mnr_essentials**.

/var/opt/ignite/clients/0x*LLA***/recovery**
The per-clients recovery directory. It holds the client's recovery configuration, log, and status files as described below.

/var/opt/ignite/clients/0x*LLA***/recovery/archive_content**
Supplies files and directories to be included or excluded. Using the **-x** command line arguments will cause this file to be ignored.

/var/opt/ignite/clients/0x*LLA***/recovery/latest**
A symlink to the *date,time* directory containing the newest set of recovery files as described below.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***
Directory containing files pertaining to the **make_tape_recovery** command that was run with the **-s** option at the date and time indicated in the directory name. An example path looks like:

/var/opt/ignite/clients/0x*080009123456***/***2000-12-20,13:50*

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/system_cfg**
Configuration file which describes the file system and networking configuration of the system (generated by the *save_config*(1M) command).

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/archive_cfg**
Configuration file which supplies the location and access method to the archive containing the files to be restored.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/control_cfg**
Configuration file which supplies control parameters and the command scripts to import volume groups that will be preserved and not created during the recovery.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/flist**
"File" list file which supplies the list of files that are to be archived. This is a plain text file, but should not be modified. Each line is in a special format that, if altered, could cause problems with

the archive process.

/var/opt/ignite/clients/0x*LLA***/recovery/config.local**
or **/var/opt/ignite/recovery/config.local** An optional configuration file that the user may create to add configuration information to be used during the recovery of the client. For example, you may want to add to this file to a **post_config_cmd** to remirror disks that the recovery process unmirrored. See the document **/opt/ignite/share/doc/diskmirror.pdf** for an example. Once this file is created, **make_tape_recovery** will automatically add it to any new configurations that it adds to the **LIF** area.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/recovery.log**
Default log file location for **make_tape_recovery**.

/var/opt/ignite/clients/0x*LLA***/recovery/***date,time***/manifest**
Software and hardware manifest information installed and configured for the system at the time the archive was created. See *print_manifest*(1M).

/var/opt/ignite/clients/0x*LLA***/recovery/defaults**
Supplies the default options to **make_tape_recovery**. Created when run interactively using the **-i** option or by using the **ignite** GUI.

/var/opt/ignite/clients/0x*LLA***/recovery/client_status**
File used to communicate the status of the **make_tape_recovery** command back to the **ignite** GUI running on the Ignite-UX server. This file is created only when the **-s** option is used.

SEE ALSO

make_boot_tape(1M), make_medialif(1M), make_net_recovery(1M), manage_index(1M), pkg_rec_depot(1M), print_manifest(1M), save_config(1M), swinstall(1M), instl_adm(4), ignite(5).

NAME

manage_index − manage Ignite-UX INDEX files

SYNOPSIS

/opt/ignite/bin/manage_index -a -f *cfg_filename* [**-c** *cfg_clause_name* | **-r** *release*] [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -a -s *script_filename* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -d -c *cfg_clause_name* | **-r** *release* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -e -c *cfg_clause_name* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -l [**-c** *cfg_clause_name* | **-r** *release*] [**-o**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -l -o [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -m *old_clause_name* **-c** *new_clause_name* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -n *existing_clause_name* **-c** *new_clause_name* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -t -f *cfg_filename* [**-c** *cfg_clause_name* | **-r** *release*] [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -t -s *script_filename* [**-p**] [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -w -c *cfg_clause_name* [**-v**] [**-i** *index_filename*] [**-?**]

/opt/ignite/bin/manage_index -x -c *cfg_clause_name* [**-v**]

/opt/ignite/bin/manage_index -y *description* **-c** *cfg_clause_name* [**-v**] [**-i** *index_filename*] [**-?**]

DESCRIPTION

This utility is used to manipulate Ignite-UX **INDEX** files. By default it operates on the **/var/opt/ignite/INDEX** file, but an alternate **INDEX** filename may be given. **manage_index** is primarily called by other Ignite-UX tools but may also be called directly.

Options

manage_index recognizes the following options:

-a Add the configuration file specified in the **-f** option to the cfg clauses as specified by the other options:

With **-c**: add to a particular cfg clause.

With **-r**: add to all cfg clauses for that release

If neither **-c** nor **-r** are specified and the configuration file name begins with **/opt/ignite/data/Rel_***release* or **/var/opt/ignite/data/Rel_***release*, the part after the **Rel_** until the next **/** is the default *release*. In this case, the new configuration file is added to all cfg clauses for this default release.

Also, it may be used to add a specified script to the scripts clause. The **-a** option deals with scripts when the **-s** option is also specified.

When adding either a configuration file or a script file, the file must exist and be readable before **manage_index** will alter the **INDEX** file. If the file doesn't exist, an error message will be printed and a **1** returned.

-d Delete an entire cfg clause or clauses. Which cfg clause(s) gets deleted depends on the other options:

With **-c**: delete a particular cfg clause.

With **-r**: delete all cfg clauses for that release.

-e Establish the default cfg clause to be whatever clause is specified in the **-c** option. If any other cfg clause in **/var/opt/ignite/INDEX** is already the default, the old default is turned off. The only possible errors are if the cfg clause specified by **-c** does not exist or if there is more than one instance of it.

-l List the names of the cfg clauses as specified by the other options:

With **-c**: list the desired cfg clause named. This is a quick way to check whether a clause exists.

With **-r**: list the names of all cfg clauses for the specified release. Also see **-o**.

-w List the names of the configuration files in the cfg clause specified by the **-c** option.

-x Display the description of the cfg clause specified by the **-c** option.

-y Set the description of the cfg clause specified by the **-c** option.

-m Move (rename) an existing cfg clause. The old clause name is specified after the **-m** and the new name is specified via the **-c** option. For example:

manage_index -m *'old_name'* **-c** *'new_name'*

-n Create a new cfg clause from an existing clause (copy). The existing clause name is specified after the **-n** option and the new clause name is specified via the **-c** option. For example:

manage_index -n *'existing_name'* **-c** *'new_name'*

-t Trim a cfg clause or clauses; that is, remove a configuration file from them. This option requires that a configuration file be specified via the **-f** option. Which cfg clause(s) gets trimmed depends on the other options:

With **-c**: trim a particular cfg clause.

With **-r**: trim all cfg clauses for that release.

If neither **-c** nor **-r** are specified and the configuration file name begins with **/opt/ignite/data/Rel**_*release* or **/var/opt/ignite/data/Rel**_*release*, the part after the **Rel**_ until the next **/** is the default *release*. In this case, all cfg clauses for this default release are trimmed.

When the **-s** option is specified, trim a script clause from the **INDEX** file. That is, remove a script from it. The **-t** option deals with scripts when the **-s** option is also specified.

-? Display the help screen.

The remaining options are used to modify the preceding options:

-c *cfg_clause_name*
Only process the cfg clause with the specified name. The name is the label attached to a cfg clause; e.g., "HP-UX 11.23 default".

-f *cfg_filename*
Use the specified configuration file name.

-o When specified with **-l**, the name of the default cfg clause is returned. If there is no default clause, the return code will be non-zero.

-p Preview. Don't actually make any changes, just write messages to stdout explaining what would be done without the **-p** option.

-r *release*
Only process cfg clauses from a given release. An example release is B.11.23. A cfg clause is said to belong to a particular release if any of the configuration files specified in it are located in the release-specific directory for that release. For example, if a cfg clause contains the **/opt/ignite/data/Rel_B.11.23/core_cfg** file, this cfg clause belongs to the B.11.23 release.

-s *script_filename*
Use the specified script. When specified, this option makes the behavior of the **-a** and **-t** options operate on the scripts clause instead of the cfg clause.

-i *index_filename*
Use the specified **INDEX** file. An absolute path name must be given. This option may be useful in modifying the per-client **CINDEX** file used for network system recovery.

-v Verbose mode. By default, the **manage_index** tool does not write any messages when everything works. When verbose mode is specified, more detail is written to **stdout** about what has happened.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

List all of the cfg clauses defined for the B.11.23 release:

```
manage_index -l -r B.11.23
```

List all of the cfg files in the cfg clause for the B.11.23 release:

```
manage_index -w -c "HP-UX B.11.23 Default"
```

Add the configuration file called **/opt/ignite/data/Rel_B.11.23/my_cfg** to each of the cfg clauses in **/var/opt/ignite/data/INDEX** which are for the B.11.23 release. Notice that the **-r B.11.23** is redundant in this case:

```
manage_index -a -r B.11.23 -f /opt/ignite/data/Rel_B.11.23/my_cfg
```

Add the configuration file called **/opt/ignite/data/Rel_B.11.23/cfg1** to the cfg clause titled "HP-UX B.11.23 Default".

```
manage_index -a -c "HP-UX B.11.23 Default"
   -f /opt/ignite/data/Rel_B.11.23/cfg1
```

Add the configuration file called **/opt/ignite/data/Rel_B.11.23/me_cust_cfg** to the cfg clause called "Std ME Pack".

```
manage_index -a -c "Std ME Pack" -f
    /opt/ignite/data/Rel_B.11.23/me_cust_cfg
```

Add the script called **/var/opt/ignite/scripts/setup_server** to the scripts clause.

```
manage_index -a -s /var/opt/ignite/scripts/setup_server
```

Remove all cfg clauses for the B.11.23 release:

```
manage_index -d -r B.11.23
```

Remove the cfg clause called "HP-UX B.11.23 Default".

```
manage_index -d -c "HP-UX B.11.23 Default"
```

Remove the configuration file called **/opt/ignite/data/Rel_B.11.23/my_cfg** from any cfg clauses for the B.11.23 release:

```
manage_index -t -f /opt/ignite/data/Rel_B.11.23/my_cfg
```

Remove the script called **/var/opt/ignite/scripts/setup_server** from the scripts clause.

```
manage_index -t -s /var/opt/ignite/scripts/setup_server
```

Make the cfg clause called "HP-UX B.11.23 Default" the default:

```
manage_index -e -c "HP-UX B.11.23 Default"
```

Copy the existing cfg clause called "HP-UX B.11.23 Default" to a new clause called "HP-UX B.11.23 Default, DART75".

```
manage_index -n "HP-UX B.11.23 Default"
    -c "HP-UX B.11.23 Default, DART75"
```

Move (rename) the existing cfg clause called "HP-UX B.11.23 Default" to "HP-UX B.11.23 Default, AR".

```
manage_index -m "HP-UX B.11.23 Default"
    -c "HP-UX B.11.23 Default, AR"
```

Display the description for the cfg clause "HP-UX B.11.23 Default".

```
manage_index -x -c "HP-UX B.11.23 Default"
```

Set the description for the cfg clause "HP-UX B.11.23 Default, AR" to "This selection supplies 11.23 bits with additional AR content".

```
manage_index -c "HP-UX B.11.23 Default, AR"
  -y "This selection supplies 11.23 bits with additional AR content"
```

This is the three step process to support two versions of apps in a single release. In the first step, a new cfg clause is created. In the second step, the old **apps_cfg** configuration file is removed from the new clause. In the final step, the new **apps_cfg_DART75** configuration file is added to the new clause:

```
manage_index -n "HP-UX B.11.23 Default"
    -c "HP-UX B.11.23 Default, DART75"

manage_index -d -c "HP-UX B.11.23 Default, DART75"
    -f /opt/ignite/data/Rel_B.11.23/apps_cfg

manage_index -a -c "HP-UX B.11.23 Default, DART75"
    -f /opt/ignite/data/Rel_B.11.23/apps_cfg_DART75
```

Show the name of the current default cfg clause:

```
manage_index -l -o
```

RETURN VALUE

manage_index returns **0** upon successful completion of the task, and returns **1** if the operation cannot be completed.

AUTHOR

manage_index was developed by Hewlett-Packard Company.

FILES

/var/opt/ignite/INDEX

/var/opt/ignite/clients/0x*LLA***/CINDEX**

/opt/ignite/bin/manage_index

SEE ALSO

ignite(5), instl_adm(1M), instl_adm(4), instl_bootd(1M), make_config(1M), make_depots(1M), make_net_recovery(1M), swinstall(1M).

NAME

pkg_rec_depot − create a depot containing Ignite-UX recovery filesets

SYNOPSIS

/opt/ignite/lbin/pkg_rec_depot [**-f**] [**-?**]

DESCRIPTION

pkg_rec_depot is used to create a software depot on an Ignite-UX server for use in distributing Ignite-UX software to client systems that use the server for network system recovery (see *make_net_recovery*(1M)) and tape system recovery (see *make_tape_recovery*(1M)). The depot created is in the directory:

/var/opt/ignite/depots/recovery_cmds

The depot created by this command is a "packaged-in-place" depot, meaning the depot references the files as they exist on the system and does not make an extra copy of the files. This reduces the space consumed. It also means that if you modify any files distributed with Ignite-UX on the server, those modifications will be propagated to any systems that load from the depot that this command creates.

pkg_rec_depot will create a depot with a bundle entitled **IUX-Recovery** that contains the software filesets required to run the **make_net_recovery** command. It also creates a bundle entitled **Ignite-UX-XX-YY** for each **FILE-SRV-XX-YY** fileset found on the server.

Each of these bundles contains the software filesets required to run the **make_tape_recovery** command on a client with HP-UX release **XX.YY**.

Options

pkg_rec_depot recognizes the following option:

-f Forces **pkg_rec_depot** to recreate the depot even if the version of the software it currently contains matches the version loaded on the server. By default, if the depot appears to have the same fileset versions as that installed on the system, **pkg_rec_depot** will not take any actions.

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

WARNINGS

The **pkg_rec_depot** command will remove the depot **/var/opt/ignite/depots/recovery_cmds** prior to recreating it, so you will lose any previous modifications made to the depot by **pkg_rec_depot**.

The depot **/var/opt/ignite/depots/recovery_cmds** will be recreated (if it exists) each time Ignite-UX is installed on the server system in order to update it with the new version being loaded. Any prior manual modifications to the depot will be lost when Ignite-UX is updated.

SEE ALSO

make_net_recovery(1M), swpackage(1M), ignite(5), sd(5).

NAME

print_manifest − print a system manifest

SYNOPSIS

/opt/ignite/bin/print_manifest [**-s**] [**-o**] [**-e**] [**-i** *hw_path*][**-d**] [**-?**]

DESCRIPTION

This utility prints a formatted ASCII system manifest to **stdout**. The manifest includes information on hardware and software installed and configured on the system. It gathers information about the system every time it is run. A manifest is included with each Instant Ignition system shipped from HP.

You must be the privileged user to run **print_manifest**. See the "privileges" manpage for more information on the privileged user.

Options

print_manifest recognizes the following options:

-s Skip scanning the hardware and software on the system. Print the manifest based on the previous hardware/software scan. This is considerably faster than including the system scan, but does not include any information on hardware/software additions.

-o Print the contents of the original manifest that was created when the system was installed. This implies the **-s** option.

-e Causes the output file to be printed with HP-PCL3 control codes for an enhanced printer output.

-i *hw_path*
Do not print information related to the *hw_path* specified. Multiple **-i** options are allowed.

-d Causes disk capacity collection via **diskinfo** to be skipped. This can be used to improve performance.

-? Display the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

RETURN VALUE

print_manifest returns **0** if no errors are encountered.

AUTHOR

print_manifest was developed by Hewlett-Packard Company.

SEE ALSO

ignite(5).

NAME

save_config − create hardware configuration file

SYNOPSIS

/opt/ignite/bin/save_config [**-f** *file*] [*vg* ...] [*disk* ...] [**-?**]

DESCRIPTION

save_config extracts disk and file system structure information and certain system and networking parameters for the currently running system and writes them into a configuration file. When used in conjunction with an archive, it provides a quick means for restoring a system with Ignite-UX (see *ignite*(5)).

Errors are always logged to **stderr**.

Options

save_config recognizes the following options:

-f *file* Specifies an alternate path for the configuration file. The default path is **/var/opt/ignite/local/config.recovery**. If the file already exists and **save_config** is successful, the existing version is first copied to *file***.prev** before the new one is moved into place. If *file* is **-**, output is sent to **stdout**. This allows **save_config** to be used as the beginning of a pipeline.

vg Specifies the named volume group that should be included in the configuration file. None of the disks that comprise this volume group should be specified as well. Volume groups are represented by the basename of their volume group name. Multiple volume groups are allowed.

disk Specifies that the named whole disk should be included in the configuration file. The disk must not be part of a volume group. Disks are represented by their block device file name (/dev/dsk/... or /dev/disk/...). Multiple whole disks are allowed.

-? Displays the help screen.

If no volume group nor whole disk arguments are given, all volume groups and all whole disks that are either mounted or used for swap will be included.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

To create a configuration file in */var/tmp/myconfig* with all volume groups and all whole disks included:

```
save_config -f /var/tmp/myconfig
```

To create a configuration file using the default location containing just the volume group **/dev/vg00**:

```
save_config vg00
```

To display the configuration information to **stdout**:

```
save_config -f -
```

SEE ALSO

make_medialif(1M), instl_adm(4), ignite(5).

NAME

setup_server − perform some administration tasks for an Ignite-UX server

SYNOPSIS

/opt/ignite/lbin/setup_server [**-g** *gateway_ip_address*] [**-i** *ip_address* | *ip_address_1-ip_address_n* | *ip_address:MAC_address*]
[**-I** *ip_address* | *ip_address_1-ip_address_n*] [**-l** *lan_interface*]
[**-c** *class_id*] [**-t** *true* | *false*] [**-h** *true* | *false*] [**-n**]
[**-u** *none* | *target* | *server* | *show*] [**-?**]

DESCRIPTION

There are several tasks which need to be performed to administer an Ignite-UX server. The **setup_server** tool handles many of these tasks, and is is designed to provide a very simple interface. It has a command-line interface only with no interactive capabilities. The GUI in *ignite*(5) provides the same functionality interactively.

In general, the options are independent from one another. If an error is encountered while processing one option, any other options are generally processed. Also, options should only be specified once during a given invocation (the last specification is the one used).

Options

setup_server recognizes the following options:

-c *class_id* Sets the DHCP *class_id* which will be used to filter DHCP lease requests. This allows the DHCP server to respond only to Ignite-UX clients.

-g *gateway_ip_address*
Only valid when the **-I** option is also specified. This option lets you specify which gateway address should appear in the DHCP entry being created using the **-I** option. The default gateway will be the entry found in the **/etc/rc.config.d/netconf ROUTE_GATEWAY** entry.

-h *true*| *false* Modify the configuration file in the RAM FS specifying that all newly installed clients should halt when complete. The default is *false*, indicating the client will reboot.

-i *ip_address*|"*ip_address_1-ip_address_n*"|"*ip_address*":"*MAC_address*"
Sets up one or more temporary IP addresses for booting during the network installation process. Each simultaneous installation session requires its own IP address. Note that the IP addresses here are only used briefly (three (3) minutes). The addresses are recorded in the **/etc/instl_boottab** file. If only the IP (or a range of IPs) is specified, any install client may use the IP during the installation process. If you wish to assign an IP to a particular client, you may also include the client's MAC address (the MAC address must be specified as a hex number without a leading 0x). If this information is included, only the client you specify may use the IP. The IP addresses used during the majority of the installation are specified using the **-I** option.

-I *ip_address*|*ip_address_1-ip_address_n*
Sets up one or more temporary IP addresses for the remainder of the network installation process. Each simultaneous installation session requires its own IP address allocated using DHCP. It is preferable to allocate one (or more) range(s) rather than individual IP addresses.

-l *lan_interface*
Only valid when the **-I** option is also specified. This option is used to specify which lan interface should be used when setting the DHCP entry for this range of addresses. By default, the interface specified by **INTERFACE[0]** in the **/etc/rc.config.d/netconf** file is used. Examples of interfaces are LAN0 and LAN1.

-n Sets NFS mounts to allow client/server communication. This is an optional step. By default, no NFS exports from the server are required. However, if you wish to control the installation process from the server *or* if you wish to make sure that configuration files and manifest information are placed on the server, you must specify the **-n** option. This means that **/var/opt/ignite/clients** will be exported read/write.

-t *true*| *false* Mark networking information supplied to the client as temporary (i.e., valid for the installation session only). The default is *false*, which means the supplied networking information will be permanent.

-u *none*|*target*|*server*|*show*

These options specify where (and if) the Ignite-UX user interface should run by default. The **-u** *none option means no user interface is executed. The* **-u** *target option means that a terminal user interface (TUI) will be executed on the console connected to the target client. The* **-u** *server option means that a graphical user interface (GUI) will be run on the server. Specifying* **-u** *show describes the current state.*

-? Displays the help screen.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Set a few IP addresses for use during the initial booting of a target client:

setup_server -i *192.168.54.97-192.168.54.99*

Set a bank of IP addresses to use for the remainder of the network installation process. Instead of the default gateway, specify an alternate:

setup_server -I *192.168.54.105-192.168.54.114* **-g** *192.168.54.25*

Set a bank of IP addresses to use for the remainder of the network installation process, and specify a *class_id* to be used for filtering DHCP lease requests.

Note: Using the *class_id* provides a closed environment for the Ignite-UX server and clients by ensuring that DHCP leases in this address range will be utilized *only* by the Ignite-UX server and any clients booting from the Ignite-UX server.

setup_server -I *192.168.54.105-192.168.54.114* **-c** *MyIgniteEnv*

Set a bank of IP addresses to use for the remainder of the network installation, and set a flag instructing the client to halt at the completion of the installation. Finally, mark the networking information being supplied to the client for the installation process as temporary:

setup_server -I *192.168.54.105-192.168.54.114* **-h** *true* **-t** *true*

RETURN VALUES

setup_server exits with one of the following values:

0 Successful completion of the task.

1 Interrupted for signals 1 (SIGHUP), 2 (SIGINT), 3 (SIGQUIT) or 15 (SIGTERM).

2 An invalid command-line option was given.

3 **setup_server** was not invoked as user "root".

4 **setup_server** was invoked with the -? (help) option.

6 **setup_server** encountered 1 or more errors.

SEE ALSO

instl_adm(1M), instl_bootd(1M), make_config(1M), make_depots(1M), manage_index(1M), swinstall(1M), instl_adm(4), ignite(5).

NAME

auto_adm − manage LIF AUTO configuration files

DESCRIPTION

auto_adm files occur in two different formats, **auto_conf** (**CONF**) **and Initial System Loader** (**ISL**) **format. The** *auto_adm*(1m) utility is used to manipulate these files, and to convert between formats.

Information contained in an **auto_adm** file is used during the boot up process to offer the user a set of boot options, and to direct the **ISL** and Secondary System Loader (**SSL**) **to execute the desired boot sequence.**

CONF Format

The **CONF** format is intended to be human readable, and consists of two classes of information: global and image specific. The global information specifies the following *keyword=value* pairs:

timeout=NN

Wait *NN* seconds until executing the default boot action. 0 (Zero) means wait indefinitely.

message=string

A string containing the prompt displayed to to the user on the console during boot up, for example, "Choose Operating System to Install." The string need not be enclosed in quotes.

default=MM

The default image to boot after the timeout expires or user presses "Enter".

The image-specific portion of the **CONF** file consists of one or more sets of four *keyword=value* pairs describing a boot image action.

label=string

A string label that will be displayed to the user on the console describing what action will be performed if this boot image is loaded. For example, "Install 64-bit HP-UX B.11.11" The string need not be enclosed in quotes.

bootcmd=cmd

The SSL command to be executed if this image is selected. See *hpux*(1M) for more information.

boot=hwpath

The hardware path relative to the boot LIF containing the desired boot image. Typically **(;0)**. See *hpux*(1M) for more information.

image=path

The path of the image to be booted relative to the hardware path specified in the boot directive.

An example **CONF**-format **auto_adm** file is:

```
timeout = 0                          # wait indefinitely
message = Choose version to install  # prompt to user
default = 2                          # choose 64-bit by default

label   = B.11.11 32bit install
bootcmd = boot
boot    = (;0)
image   = boot/Rel_B.11.11/INSTALL   # path to 32-bit install kernel

label   = B.11.11 64bit install
bootcmd = boot
boot    = (;0)
image   = boot/Rel_B.11.11/WINSTALL  # path to 64-bit install kernel
```

ISL Format

The **ISL** format is the file format actually written into the boot LIF by **auto_adm**, assembled from one or more **CONF**-formatted files.

Like the **CONF** format, **ISL** formatted files consist of a global section and an image-specific section.

The global section consists of a single line:

```
hpux KernelPrompt message timeout default
reset
```

where the first two words are literally **hpux KernelPrompt**, and *message*, *timeout*, and *default* are as defined above under the **CONF** format section. This line is followed by a line containing only the word **reset**.

The image-specific section contains one or more lines of the format:

```
label cmd device image
```

with each of these terms being defined under **CONF**.

Finally, the image-specific section is followed by the line:

```
"Exit"  reboot
```

The **ISL**-formatted file corresponding to the above **CONF** file is:

```
hpux KernelPrompt "Choose target Operating System: " 15 2
reset
"B.11.11 32bit install"  boot (;0)boot/Rel_B.11.11/INSTALL
"B.11.11 64bit install"  boot (;0)boot/Rel_B.11.11/WINSTALL
"Exit"                   reboot
```

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

AUTHOR

auto_adm was written by Hewlett-Packard Company.

SEE ALSO

auto_adm(1M), boot(1M), hpux(1M), isl(1M), lif(4).

NAME

instl_adm − Ignite-UX configuration file syntax

DESCRIPTION

The Ignite-UX configuration files determine the default system configurations available during the process of installing the HP-UX operating system.

The Ignite-UX process uses configuration files to bring in default parameters, as well as, determine how the system will be configured. During the installation process, you are presented with a user interface (UI) that allows you to make selections that are implemented inside the configuration files as *variables*, *use models*, and *software selections*. The UI also allows the user to make modifications beyond what is specified by these constructs (such as adding new file systems or volumes).

Once you approve the configuration, a configuration file is created. This file contains only the changes made in UI. This configuration file is then passed on to another utility that actually performs the system configuration and software installation.

Organization and Indexing of Configuration Files

Although all configuration files use the same basic syntax, the information supplied by them is split into multiple files to allow them to be maintained individually. These configuration files fall into the basic classes of:

- Default disk and file system layout. Located in
  /opt/ignite/data/Rel_*release***/config**
- Software description of a single Software Distributor (SD) depot,
  or system archive. Located in **/var/opt/ignite/data/Rel_***release***/***file*
- Local configuration overrides that apply globally. Located in
  /var/opt/ignite/config.local
- Named configurations created by saving a configuration via the UI.
  Located in **/var/opt/ignite/saved_cfgs/***
- Client-specific configuration files. Located in
  /var/opt/ignite/clients/0x*LLA***/config**
- Boot control parameters and networking information. Located in the
  first 8KB of the file **/opt/ignite/boot/Rel_***release***/*INSTALLFS** managed by the **instl_adm** utility.

To completely describe all aspects of a system, it is necessary for Ignite-UX to read multiple configuration files as a combined set.

The grouping of configuration files into a set (known as a "configuration clause" or just "cfg clause") is accomplished with the use of the **INDEX** files: **/var/opt/ignite/INDEX** or **/var/opt/ignite/clients/0x***LLA***/CINDEX**

The **INDEX** file may contain multiple configuration clauses (groups of configuration files) that together describe a unique system configuration. The UI allows the user to choose from the configurations clauses that are defined in the **INDEX** file.

Similarly, the **CINDEX** file can contain configuration clauses, but these are only available for use by the specific client corresponding to the per-client directory it exists in. The **CINDEX** file is typically used by the **make_net_recovery** command to reference configuration files created for each recovery backup performed for that client. The paths to configuration files in the **CINDEX** file can be relative paths, in which case they will be accessed from within the per-client directory. The configuration clauses listed in both the **INDEX** and **CINDEX** files are available for use during an installation or recovery of the specific client.

The default **INDEX** configuration clauses provided with Ignite-UX are used to define the aspects of each HP-UX release that Ignite-UX supports. A separate configuration clause per HP-UX release is necessary because each HP-UX release has different defaults for file system layout and software bundling.

The ability to save a modified system configuration through the UI makes use of the **INDEX** file. When the UI is used to make modifications to the system configuration, Ignite-UX produces a "delta" configuration file that represents the changes. When this delta configuration file is appended to the list of default configuration files, it overrides the defaults with the values you specified. When a system configuration is saved with a new name, a new **INDEX** configuration clause is created that contains the original configuration files with the addition of the new delta configuration file.

Any information stored in the first 8KB of the ***INSTALLFS** file, as well as the client-specific configuration file, is always appended to the list of configuration files, although is not explicitly listed in the **INDEX** file set.

INDEX File Content and Syntax

The **INDEX** and **CINDEX** files contain configuration clauses, and may also contain a list of scripts that are available for scheduling in the "Advanced" tab of the UI.

The **INDEX** file syntax is of the form:

```
cfg "tag_string_1" {
     description "description_text"
     "config_file_path_1"
     "config_file_path_2..."
}
cfg "tag_string_2" {
     description "description_text"
     "config_file_path_1"
     "config_file_path_2..."
} = true | false
scripts {
     "script_file_path_1"
     "script_file_path_2"
}
```

In the above example syntax, the **cfg** keyword is used to specify a group of configuration files. The *tag_string* is displayed in the UI selection list, and must be 80 characters or less. The *description* keyword supplies additional information the UI may display to further describe a selection. After the closing brace, the **cfg** clause may be selected or unselected using the **= true**|**false** syntax. A setting of **= true** is used to select a particular configuration as the default.

Only one **cfg** clause may be selected as the default. The last clause that is selected will cause all other clauses to be automatically unselected. See the **EXAMPLES** sections on how to set up a **cfg** clause to be conditionally selected.

The **scripts** keyword is used to specify a list of file paths to scripts that may be selected and scheduled to run at the end of the client install. (See the description of the **post_config_script** keyword for more details.)

General Configuration File Structure

The configuration file is a free-structured file in which white space is optional but may be used to improve readability. The **#** character is used to designate a comment line that extends from the **#** character to the end of the line.

Some keywords require string arguments. Most keywords that take a string argument may also take a complex string which is a concatenation of strings, variables and system attributes. See the following descriptions for details.

All keywords in the configuration file may be either all lowercase or all uppercase, but not a combination. Variable and use-model identifiers are case sensitive.

Most keywords, and all variables, that use the **=** assignment operator will also accept the **+=** operator, which appends the given value to any value that the keyword/variable currently holds. The **+=** operator has the same effect as **=** when it is the first assignment. Using the **+=** operator on an integer variable

performs addition, whereas using it on string variables/keywords performs concatenation.

There are several constructs in the configuration-file syntax where a set of parameters are associated with a specific object. For example, the file-system parameters associated with a given file system. If the object being defined has already been defined, the new parameters are merged with any parameters previously assigned to the object. It is thus possible to override a small set of parameters while letting the rest be defined by the default configuration file.

Constants

`Integer`:

An integer may be a decimal value specified using digits **`0`** through **`9`**. No special significance is given to leading zeros.

An integer may be a hexadecimal value specified using the prefix **`0x`** followed by a value using the hexadecimal digits **`0`** through **`9`**, **`a`** through **`f`**, and **`A`** through **`F`**.

An integer may be a decimal value and may have a **`KiB`**, **`MiB`**, **`GiB`** or **`TiB`** suffix multiple. For compatibility with prior Ignite-UX releases a **`KB`**, **`MB`**, **`GB`** or **`TB`** suffix multiple may be used instead. These multiples are **`KiB`** or **`KB`** for 2^10, **`MiB`** or **`MB`** for 2^20, **`GiB`** or **`GB`** for 2^30, and **`TiB`** or **`TB`** for 2^40. According to International System of Units (SI) notation the preferred multiples are **`KiB`**, **`MiB`**, **`GiB`**, and **`TiB`**. Note that Ignite-UX does not support 10^3, 10^6, 10^9, and 10^12 decimal multiples. Further information about binary multiples is available from Commission Electrotechnique Internationale (IEC) and National Institute of Standards and Technology (NIST).

`String`:

A string is text surrounded by double-quote (") characters. If a double quote is needed inside the string, it must preceded with the backslash (\) character. No other characters, including newlines, need to be preceded with a backslash character.

Most keywords also allow for complex strings (*cplx_string*). Complex strings may be regular quoted strings or may be a combination of multiple strings, variable names, mathematical expressions, or system-attribute keywords enclosed with ${} combined using the "+" operator. For example: "The system contains " + ${num_disks} + " disk drives".

A complex string may also have a format string associated with it to allow specific formatting to be performed when converting the value into a string. The format string is passed directly to **`sprintf`**. The syntax to specify a format string is as follows:

```
mod_kernel += "maxdsiz " + ${"0X%X" _maxdsize_var}
```

where "0X%X" is the format string and causes the value of the variable to be converted to an uppercase hexadecimal value.

`Boolean`:

The boolean constants are: **`true/TRUE`** and **`false/FALSE`**.

`I/O Constant`:

Ignite-UX provides supports use of several different types of special constants related to I/O for use in conditional expressions and setting variables. A hardware path is a common example of one of these I/O types. More detailed information about these types is provided in the section on I/O Configuration.

Variables

The Ignite-UX configuration files support string and integer variables. Variable names must begin with an underscore (_) character followed by one or more characters from the set a-z, A-Z, 0-9, or underscore.

Integer variables and numeric values are stored internally in KB units when possible. This allows for larger than 4GB values to be operated on and still use a 32-bit value internally. However, the parser converts a value to regular units when addition or subtraction is performed using a non-KB value. The parser does not allow a combination of KB and non-KB values to be listed as potential values for a variable. When using a variable as part of *cplx_string*, you may need to add 0 (zero) to the value to force conversion to a non-KB

value. For example:

Swap size in bytes: +${_hp_pri_swap+0}.

Variables that the UI does not recognize as special are represented in the *Additional* button available in the UI on the *Basic* tab. From the *Additional* dialog, you may modify the values of any visible variables.

Variables do not need to be declared as their type (integer or string) is determined the first time they are assigned and cannot be assigned a different type later. Variables may be assigned using two different constructs:

init *_variable*=*value*
Preceding the assignment with the **init** keyword means the variable is to be initialized to the given value, but the UI is allowed to alter the value later.

_variable=*value*
When the **init** keyword is not used, the variable cannot be changed by the UI. This type of assignment is not recommended for "visible" variables.

Variables may be assigned a list of potential values the UI may choose from in generating a selection list for the user. This is done with the following syntax:

_variable=**{***value1*,*value2*,...**}**
Sets the list of potential values of the variable to those given and is for use by the UI only. There is no error checking to ensure the resulting value is in the given list.

_variable+=**{***value2*,*value3*,...**}**
Using the **+=** operator adds the given values to any existing values specified for that variable.

_variable=**{***number*..*number***}**
Specifies a numeric range of potential values using two numbers (or integer variables/keywords) separated by two dots.

_variable=**{disk[*], disk[*=***X***]}**
Creates a list of potential values made of the hardware paths of all the disks in the first case, and the first "X" number of disks in the second case. This is useful when you need to choose from the list of available disks on the system.

Variable values that should be restricted to one of the list of potential values may be specified as an enumeration variable using the **enum** keyword as follows:

enum *_variable*

Variables that should not be displayed in the UI should be set to invisible using the syntax:

_variable **visible_if false**

The **help_text** keyword may be used to specify a longer, more descriptive name for the variable that the UI may use in the *Additional* dialog. The syntax is:

_variable **help_text** *string*

Special Variables

These are variables that are treated specially by the UI or other parts of Ignite-UX and are described as follows:

_hp_addnl_fs_free_pct
Integer variable used to control the amount of additional free space (or "breathing room") allocated to volumes beyond the space required to load the software. If this variable is not set, it defaults to 10 (10%). See the definition for the file system **size** keyword attribute for more details.

_hp_cfg_detail_level
Internal string variable fundamental in the configuration file management done by Ignite-UX. It is used primarily in the client-specific configuration files. It contains a list of option characters that represent which aspects of the configuration file have been modified by the UI. This represents the areas of information the configuration file written by the UI contains.

The Ignite-UX process uses this information should it need to rewrite the configuration file. The configuration file is rewritten both by the UI and by the client installation process. In these cases, the process uses **_hp_cfg_detail_level** to determine how much information is represented by the configuration file and so may write the minimal amount of information, and then allow the rest be supplied by the other configuration files in the **INDEX** file set.

The option characters recognized in the **_hp_cfg_detail_level** string that may be found in a client-specific **config** file are:

'i' Index **cfg** clause selection. (File that containing the line indicating which **INDEX** file entry should be used.)

'v' Variable and use model settings.

's' Software selection settings.

'S' Modified software selection definitions.

'r' Modified software source definitions (depot information).

'f' Modified file system information.

'p' System identity information.

't' **post_config_script** selections settings.

'h' Hardware control information (**hw_instance_num statements).**

'l' Other control information (for example, global **mod_kernel** statements).

For example, the line: **_hp_cfg_detail_level="ivsp"** in a client configuration file would indicate the file contains information about which **cfg INDEX** selection should be used, the variable settings, software selection settings, and system parameters.

_hp_console_verbosity

Integer variable used to control the verboseness of output to the console. If set to 0 (zero), messages to the console are turned off during the installation. If set to 6, messages will be printed to the console even when the **ignite** UI is controlling the installation (control_from_server=true). If not set, or set to 5, the messages normally are printed to the console except when **control_from_server** is set to **true** (which is the case when using the **ignite** UI command on the server).

_hp_custom_sys

String variable used in conditional statements surrounding network and system identity information when the UI writes a configuration file. This variable may be used in conditionals in configuration files to define multiple sets of network parameters that may be selected from in the *Additional* screen. You may see how this variable is used by performing a "Save As" operation during an installation after modifying the parameters on the "System" tab and viewing the resulting file in the **/var/opt/ignite/saved_cfgs** directory.

_hp_default_cur_lan_dev

String variable set to the LAN device that is enabled during the Ignite-UX process. It is used when a LAN device is omitted from keywords that may accept a LAN interface specifier. This defaults to the interface that you chose during the install, or in the non-interactive case, the LAN device from which the client booted. If this was not a network boot, the default is **lan0**.

_hp_default_final_lan_dev

String variable similar to **_hp_default_cur_lan_dev**, used when the network information is specified by using the **final** keyword. It defaults to **_hp_default_cur_lan_dev** if not set. The IP address associated with the LAN device specified by this variable is used for the entry in the **/etc/hosts** file for the client's host name.

_hp_disable_lvm_pvlink_all

Boolean-string variable that controls whether all alternate paths to disks should be automatically added to LVM volume groups. By default, all alternate paths (up to the maximum allowed) will be

added. Setting this variable to "true" will cause only the paths explicitly specified for the volume group in the configuration file to be added. This variable has no effect when installing B.11.31 and later. Starting at B.11.31, Ignite-UX uses the agile device files to configure LVM, which takes advantage of all paths to a disk.

_hp_disable_sas_disk_reorder
Boolean-string variable that controls whether SAS disk devices will be forced to have instance numbers assigned in order of the devices' physical location. The ordering will be forced by default, which results in more predictable hardware paths between installs. If set to "true", then the forced ordering is not done and the instance number assignment (and thus the hardware path) is not dependent on the physical location (and can be somewhat unpredictable). To take effect, this variable must be set in the ***INSTALLFS** file using **instl_adm**.

_hp_disk_layout
String variable set by the UI to indicate which disk layout (LVM, whole-disk, etc.) you selected. As configurations are saved, the list of possible values is increased to contain any modified layouts.

_hp_efi_partition_size
Integer variable set by the UI to indicate the size of the HP-UX EFI boot partition on Itanium®-based clients. This disk partition contains loader software and other utilities used to boot the HP-UX operating system. This value defaults to a size that allows some space for additional boot partition content in future HP-UX releases, and is adjusted to the minimum EFI boot partition size if set to zero or a value less than the minimum. This variable is supported beginning with HP-UX B.11.23, and is only supported on Itanium®-based systems. On HP-UX B.11.31 and later, the actual size will be slightly less than this value due to compliance with the UEFI standards in the **idisk** command.

_hp_service_partition_size
Integer variable set by the UI to indicate the size of the HP Service Partition on Itanium®-based systems. This disk partition contains diagnostics software, saved client and hardware state data, and other utilities used to verify and update hardware functionality. This value defaults to a size expected by the diagnostics software. A zero value indicates that no HP Service Partition should be created. This variable is adjusted to the minimum HP Service Partition size if set to a non-zero value less minimum. This variable is supported starting with HP-UX B.11.23 and only supported on Itanium®-based systems.

_hp_force_autoboot
String variable used to modify the behavior of Ignite-UX behavior with respect to stable store's autoboot flag. The Ignite-UX configuration process has two parts separated by a reboot. By default **_hp_force_autoboot** is set to **YES**, guaranteeing that the Ignite-UX autoboot flag is set during the installation process and then reset to its previous state (if necessary) at the end of the installation. If **_hp_force_autoboot=NO**, Ignite-UX will not touch the autoboot flag in stable storage. Note that this may mean that you will have to perform a manual boot from the primary path between the two parts of the Ignite-UX installation.

_hp_force_setboot_path
Boolean-string variable for installs of HPUX B.11.31 and later, when set to "true", the lunpath hardware path (for example 0/1/1/0.0x1.0x0) that is selected as the root disk will be supplied to the **setboot** command. When set to "false", the persistent device special file (for example /dev/rdisk/disk3) will be supplied to the **setboot** command. For more details on setboot behavior see *setboot*(1M). The default is "false" if not explicitly set.

_hp_loadfile_use_nfs
When set to "true", causes Ignite-UX to use NFS instead of TFTP to transfer files from the Ignite-UX server. The default is "false" if not explicitly set.

_hp_ht_enable
Boolean-string variable that indicates whether to enable HyperThreading on HyperThread-capable systems, as reported by the **is_ht_capable** variable. Available on HP-UX B.11.31 or later.

_hp_ignore_prior_config
Boolean-string variable that can be used to prevent the use of the **config** file in the client's /var/opt/ignite/clients/<client> directory. When set to "true" any pre-existing **config** file will be renamed to **config.ignore** and then be ignored. The user interface will not give any choice to use the prior **config** file when this variable is set to "true". The default is for the client's **config** file to be used. To take effect, this variable must be set in the ***INSTALLFS** file using **instl_adm**.

_hp_ignore_sw_impact
Integer variable that may be set to 1 to disable all effects the **impacts** statements declared in the software selections on the volume size calculations. This may be helpful to ensure Ignite-UX does not automatically modify the file system volume sizes.

_hp_keyboard
String variable the UI sets to the keyboard language/mapping desired. The **kbdlang** keyword used for this in past releases is equivalent to this variable. Setting this variable in the ***INSTALLFS** file using **instl_adm** prevents the system from prompting for a keyboard type during the installation. This information is stored in **/etc/kbdlang** on the final client.

_hp_lanadmin_args
If your network requires the default attributes of a lan interface be changed via the **lanadmin** command to operate correctly, you can specify the arguments to the **lanadmin** command via this variable. This setting must be performed in an installation file system using the **instl_adm** command. For more information on the options available for **lanadmin**, refer to the *lanadmin*(1M) manual page.

Any changes made with this variable are in effect during the installation only; they are not applied to the system permanently.

To set the speed and duplex values for a 100BT interface to 100 full duplex:

```
( lan[].driver ~ "btlan" ) {
    _hp_lanadmin_args="-X 100FD"
}
```

In the example configuration above, when Ignite-UX finds lan[].driver, it uses the variable **_hp_default_cur_lan_dev** as the index.

Note: All lan interfaces that support autonegotiation of the link speed and duplex values will do so when running the installation kernel. If you have 100BT lan interfaces and switches with ports hard set to 100 full duplex rather than autonegotiate, you will end up with a speed and duplex mismatch. You must use the above configuration in an installation file system.

1000BT interfaces can autonegotiate their speed and duplex values; however, if you have a switch set to operate only at 1000 full duplex, a 1000BT lan interface should automatically operate at that speed and duplex. In this situation, there is no need to use **_hp_lanadmin_args**. Gigabit ethernet over fibre can operate only at 1000 full duplex, so no configuration is necessary.

Note: Do not specify a PPA value in the **_hp_lanadmin_args** variable; Ignite-UX adds this automatically to the correct lan interface.

_hp_locale
String variable the UI sets to the language locale the user chose. If set to the string **ASK_AT_FIRST_BOOT** then the **geocustoms** application is invoked when the installation is complete to allow you to choose the desired language. Setting the value to **SET_NULL_LOCALE** leaves the system to default with no **LANG** variable set, in which case commands will use the internal messages that provide better performance. If **_hp_locale** is not set, it is set internally to the first value of the **locale** keyword for a selected **sw_sel**. If there are multiple selected

sw_sels which contain **locale** keywords, the **_hp_locale** variable is set to the first value of the **locale** of the first selected **sw_sel**.

_hp_min_swap
Integer variable used as the minimum size the primary swap volume may be reduced to if there is not enough disk space for all other volumes. This value defaults to an amount that should allow the system to boot and run HP-UX.

_hp_nfs_mount_opts
String variable set in the ***INSTALLFS** file and used to supply additional options to the NFS mounts that are performed during the installation. This is intended for use when the default options are not appropriate for your network. See *mount_nfs*(1M) for valid options.

Note: When multiple options are passed to *mount_nfs*(1M), they must be passed as a comma-separated list following the **-o** option.

To set the read and write NFS buffer size to 1KB:

```
init _hp_nfs_mount_opts="-orsize=1024,wsize=1024"
```

Note that this variable is not used during **make_net_recovery** operations. If you require these special options in order for **make_net_recovery** to function correctly, see *make_net_recovery*(1M).

_hp_nfs_mount_retries
Integer variable used to modify the default number of times that the NFS mounts are retried before failing. If this variable is not set, the mounts will be retried four times. If you need to change this default, this variable should be specified in the ***INSTALLFS** file. For example, to set the number of retries to 8:

```
init _hp_nfs_mount_retries=8
```

_hp_os_bitness
String variable used to set either "32" or "64" when an operating system is chosen. This is normally performed in a configuration file by keying off of the **sw_sel** for a 32 or 64-bit operating system. The **sw_sel** statements for certain applications rely on this variable to indicate which version of the application to installation based on the operating system.

_hp_pri_swap
Integer variable that may be set in the UI to indicate how much swap the user desires to have allocated on the root disk/volume-group. The default LVM primary swap volume is defined as a size range with **_hp_pri_swap** being the desired size and **_hp_min_swap** being the minimum size.

_hp_primary_path
String variable set during the initial startup of Ignite-UX on the client. This variable is set to the client's primary boot path as set in stable storage and read with **setboot**. If the client's primary boot path is incorrectly set to a non-existent device, **_hp_primary_path** is set to the empty string ("").

_hp_root_disk
String variable set by the UI to contain the hardware path or device specifier of the disk you have chosen to be the root disk. The value is initialized to the value of **_hp_primary_path** in **/opt/ignite/data/Rel_***release***/config** and may be overridden in other configuration files. If the initial value is not specified in the configuration file, it would default to the disk with the hardware path having the highest SCSI priority.

_hp_root_grp_disks
Integer variable to indicate how many disks to add to the root volume group.

_hp_root_grp_striped
String variable of possible values "YES" and "NO" to indicate if LVM data striping should be used on all disks in the root volume group if there are multiple disks in the root group.

_hp_saved_detail_level
Internal string variable used in configuration files that have been created by a "Save-As" operation. Like **_hp_cfg_detail_level**, it contains a list of option characters that represent which aspects of the configuration file have been modified. The format is identical to the **_hp_cfg_detail_level** variable.

_hp_sec_swap
Integer variable that may be set in the UI to indicate how much secondary swap the user desires to have allocated in the root volume-group. The default is zero (no secondary swap). If there is more than one disk in the root volume group, this volume is mapped to the second disk.

_hp_tftp_cmds
String variable that may be specified in the ***INSTALLFS** file to supply additional instructions to the **tftp** commands that are used to transfer data during an installation. The commands supplied with this variable are passed as input to the **tftp** command along with the usual commands supplied by Ignite-UX. The most likely use of this would be to modify the retransmission-timeout (**rexmt**) and overall timeout (**timeout**) values. The default values that Ignite-UX uses are: **rexmt** set to 2 and **timeout** set to 25. See *tftp*(1) for more details. The string assigned to this variable should contain one **tftp** command statement per line. For example:

```
init _hp_tftp_cmds="rexmt 5
timeout 40"
```

Use Model Variables

The configuration file parser also supports an older version of variables that existed in previous releases. These are referred to as *use models*. They are boolean variables in which the variable itself is a string and its value is either true or false. The variable name is presented in the *Additional* dialog that is available in the UI on the *Basic* tab. The user may toggle the boolean value.

Similar to variables, use models may be made invisible to the UI using the **visible_if** keyword. Ignite-UX does not support the **selectable_if** or **radio_sel** keywords that previous releases supported. The **radio_sel** functionality is replaced by the ability for regular variables to have a list of possible values associated with them. Additionally, the **help_text** keyword for use models has no effect.

Use models use the same **init** keyword as regular variables. When a use model is assigned using the **init** keyword, the UI is allowed to change that variable value. An example of how use models are used is shown below:

```
init  "use_model_name" = TRUE
init  "use_model_name" = num_disks > 2
"use_model_name" visible_if (memory > 64MB)
```

Logic Expressions

A block of configuration statements may be surrounded by a logic expression and a set of **{ }** brackets.

The logic expression may be made up of variables, use models or system attributes. The following logic operators are recognized (in order of precedence):

( ) Parentheses may be used to group expressions to force precedence.

! Logical not operator.

== != ~
Comparison operators: equal, not-equal and regular expression match, respectively. The **==** and **!=** operators compare strings in a case-insensitive manner. The **~** character is a regular expression comparison operator. The right side of **~** is treated as an extended regular expression string when compared to the left side string. (See *regexp*(5).)

> < >= <=
Comparison operators: greater than, less than, greater or equal, and less or equal, respectively.

& Logical *and* operator.

| Logical *or* operator.

Variables in logic expressions must use one of the comparison operators to result in a boolean value. Use model names may appear in logic expressions and are treated as a single boolean value.

Following the set of statements surrounded by **{ }** you may supply an **else** clause followed by a set of statements enclosed in **{ }** brackets. You may also specify conditional statements within conditional statements. For example:

```
(num_disks > 1 & disk[_hp_root_disk].size < 500MB) | "Use all the disks"
| hardware_model ~ "9000/8.*" | LLA[] == "080009150315" |
(_my_var > 20MB & _my_str_var ~ "Y.*")
{
  statements
  (num_disks > 2)
  {
   statements
  } else {
   statements
  }
}
```

Mathematical Operations

The configuration file supports the use of the *****, **/**, **+**, and **-** operators when dealing with integer constants, variables, or system attributes that evaluate to an integer value. For example:

```
init _hp_pri_swap = MEMORY * 2
```

System Attribute Keywords

The system attribute keywords that may be used in logic expression comparisons, mathematical operations (those with integer values), or in complex strings are:

disk[*hw_path*|*index*|*dev_spec***].size**

Evaluates to the size (in KB units) of the disk corresponding to the hardware path, index value or device specifier specified. A full hardware path, a single *index* integer (or integer variable), or a device specifier may be used to specify the disk. The *index* value may range from 0 to **num_disks-1**. If an *index* is specified greater than or equal to **num_disks**, the size value is zero. For example:

```
disk[2/0/1.6.0].size
disk[_hp_root_disk].size
disk[0].size
disk["HW_PATH='2/0/1.6.0'"].size
```

disk[*hw_path*|*index*|*dev_spec***].model**

Evaluates to the model string as reported by the disk specified by the hardware address path, index value or device specifier.

disk[*hw_path*|*index*|*dev_spec***].driver**

Evaluates to the device driver used to support the disk specified by the hardware address path, index value or device specifier.

disk[*hw_path*|*index*|*dev_spec***].wwid**

Evaluates to the world-wide identifier of the disk specified by the hardware address path, index value or device specifier. For disk devices that do not have a unique *wwid*, Ignite-UX or the install kernel will fabricate a *wwid*. See the I/O Configuration section for more information.

disk[*hw_path*|*index*|*dev_spec***].phys_loc**

Evaluates to the physical location of the disk specified by the hardware address path, index value or device specifier. Note that not all disks have a physical location, only SAS

disk drives do. See the I/O Configuration section for more information.

disk[*hw_path*|*index*|*dev_spec***].device_id**
Evaluates to the device identifier of the disk specified by the hardware address path, index value or device specifier. Note that this is an attribute set by system administrators and is not guaranteed to be unique. See the I/O Configuration section for more information.

disk[*hw_path*|*index*|*dev_spec***].dev_spec**
Evaluates to the device specifier of the disk specified by the hardware address path, index value or device specifier. See the I/O Configuration section for more information.

disk[*hw_path*|*index*|*dev_spec***]**
disk[*]
disk[*=*N**]**
Evaluates to a string that contains the hardware path(s) of the disk(s) referenced. In the first usage shown, the string contains the hardware path of the disk matching the specific *hw_path*, the *index* value or the device specifier. In the second usage, the string is a list of all hardware paths of disks on the system. Similarly, the third usage results in a string of hardware paths for the first *N* disks. The second and third usages may be useful when defining a list of potential values a variable may hold and display in the UI. For example:

```
enum _my_var
_my_var = { disk[*] }
```

vgdisk[*index***]**
Evaluates to a disk that has been already defined to be part of the volume group using the **physical_volume** keyword. The **vgdisk** keyword may be used in the same manner as the **disk** keyword, but within a **volume_group** definition only. The *index* parameter is used to index into the list of disks that make up the volume group. The first disk in the volume group may be referenced using an *index* value of 0. For example, in the root volume group, you may reference the root disk using **vgdisk[0]**. This keyword is useful in mapping logical volumes to disks within the volume group when you do not want to use a *hw_path*.

num_disks
Evaluates to the number of disks discovered on the system.

num_vgdisks
Evaluates to the number of disks specified to be part of the volume group being defined. This keyword should be used only within a **volume_group** definition after all **physical_volumes** for the volume group have been defined. It evaluates to zero if used outside a **volume_group** definition.

memory
Evaluates to the amount of memory installed on the system.

can_run_32bit
can_run_64bit
Boolean values that indicate if the system's hardware is capable of running a 32-bit or 64-bit HP-UX operating system respectively.

has_ps2
Boolean values that indicate if the system has a PS2 keyboard/mouse interface.

has_usb
Boolean values that indicate if the system has a USB (Universal Serial Bus) interface.

hardware_model
Evaluates to a string containing the equivalent of the **getconf _CS_MACHINE_MODEL** command. If that is unsuccessful, then the equivalent of the

uname -m command is used instead.

model Evaluates to a string containing the equivalent of the **model** command, which is typically more descriptive than **hardware_model**.

is_hppa
is_ia64
Boolean values that indicate if a system is PA-RISC or Itanium®-based architecture, respectively.

is_ht_capable
Boolean value that indicates if a system supports HyperThreading. Available on HP-UX B.11.31 or later.

num_cpus
Evaluates to the number of active processors discovered on the system.

num_lans
Evaluates to the number of LAN hardware devices discovered on the system.

lla[[*hw_path/interface*]**]**
Evaluates to a string containing the MAC address (or station address) of the client. The string will be in the form of a hex number (without a leading 0x). Evaluates to "unknown" if the system does not have a LAN interface.

The *hw_path/interface* specifier may be omitted; however, the square-brackets must still be used. It may be the full hardware path to the LAN device, or may be a LAN interface name. When the specifier is omitted, the value of the variable **_hp_default_cur_lan_dev** is used. See the special variables described previously. For example:

lla[] == "080009123456" | lla[2/0/2] == "080009654321" |
lla["lan0"] == "080009654321" {
statements
}

ip_addr[[*hw_path/interface*]**]**
Evaluates to a string containing the IP address of the system as set by the user, by DHCP, or by the configuration file. This is the value used during the installation, and is not the "final" value. If not set in the configuration file, this value is not set on initial boot, but is set by you during an interactive installation.

The *hw_path/interface* specifier follows the same rules as the **lla** keyword above.

lan[[*hw_path/interface*]**].driver**
Evaluates to the device driver string name used to support the specified LAN device (for example, "lan1", "btlan3", etc). The *hw_path/interface* specifier follows the same rules as the **lla** keyword previously described.

system_name
Evaluates to a string containing the host name of the system as set by the user, by DHCP, or by the configuration file. This is the value used during the installation, and is not the "final" value. If not set in the configuration file, this value is not set on initial boot, but is set by you during an interactive installation.

source_type
Evaluates to a string indicating the type of media being installed. This may be one of the following: **"DSK"** (if a CD/DVD), **"MT"** (if DDS or mag-tape), or **"NET"** (if using a network server).

release
A string corresponding to the HP-UX release for which the configuration file contains the system defaults (for example, "B.11.11", "B.11.23", etc.).

is_net_info_temporary
A boolean value that reflects the choice made by you in the network information screen or as set using the keyword of the same name in the configuration file.

(sw_sel *string***)**
A boolean value that reflects whether the software selection tag given as *string* has been selected for loading.

Note: It is not advisable to initialize variables within a **sw_sel** conditional statement because the variables are not be re-initialized if a you make a change to the **sw_sel** in the UI.

(cfg *string***)**
A boolean value that reflects whether the INDEX file selection referenced by *string* is the one being used for this installation.

I/O Configuration

Some of the variables that control the I/O configuration process use special types of values. These value types may also be used by some conditional expressions and other variables. These special I/O configuration value types are:

Hardware Path (*hw_path***):**
For keywords that take a hardware path as an index parameter or a value, the hardware path may be a series of more than one decimal or hexadecimal numbers separated by the period (**.**) or the forward slash (**/**) characters. A complex string or string variable may also be used where a hardware path is expected. A hardware path refers to a particular node in the system I/O configuration (sometimes called the I/O tree). The hardware path value may be required to refer to specific I/O node type for some variables. For example, the hardware path may need to be an I/O adapter path or a disk device path. Note that hardware paths for mass storage devices (e.g. disks) in HP-UX 11.31 and later releases include: LUN hardware path, lunpath hardware path and legacy hardware path types.

Physical Location (*phys_loc***):**
A physical location may be a series of alphanumeric values separated by the colon (**:**) character. This value indicates the actual physical device location (e.g. based on an enclosure and bay number). Hardware paths for some I/O protocols vary from install to install. A physical location is a human-readable value which consistently refers to a device based on where the device is physically located in the system configuration. Physical location is not available for all I/O protocol types. Physical location may not be available for all devices of a protocol type even though it is available for some devices of that protocol type.

World-Wide Name / World-Wide Identifier (*wwid***):**
A globally unique value specific to an individual device. The format of the value varies depending on the protocol and the device. This value is often a standard IEEE value in hexadecimal format however this value may have some other format. The formats may vary widely but should not contain white space. Rarely, old devices may not support a unique identifier and a wwid value will be generated. Such a generated wwid value will not match from install to install and recovery to recovery.

Device ID (*device_id***):**
This value is typically set by system administrators using tools like **scsimgr** on HP-UX B.11.31 and later releases (see *scsimgr*(1M)). As such, it is not guaranteed to be unique. Ignite-UX will recognize this attribute and will use it to find a disk, but Ignite-UX

tools like **save_config** will not supply it. Use of this attribute is therefore discouraged.

Device Specifier (*dev_spec***):**
A method that combines the above four types into a single prioritized list of attributes that refer to a disk. The format is a String, which means the entire method is surrounded by double quotes. Tools that create this method determine the order of attributes, based upon which I/O Protocol applies and what **release** is involved. Attributes are separated by white space and the value part is surrounded by single quotes. Any embedded single or double quote characters must be escaped. The following attributes are supported:

WWID='*wwid***'**
A disk with the specified World-Wide Identifier *wwid* is searched for.

PHYS_LOC='*phys_loc***'**
A disk at the specified physical location *phys_loc* is searched for.

HW_PATH='*hw_path***'**
A disk at the specified hardware path *hw_path* is searched for. Pattern matching expressions are supported as the *hw_path*.

DEVICE_ID='*device_id***'**
A disk with the device ID *device_id* is searched for.

These attributes can then be listed in a prioritized list, as in:

disk["WWID='*wwid***' PHYS_LOC='***phys_loc***' HW_PATH='***hw_path***"]**

In this example, Ignite-UX would first look for a disk with a **WWID** of *wwid*, and if it could not be found, then it would look for a disk with a **PHYS_LOC** of *phys_loc*, and if that could not be found, then it would look for a disk with a **HW_PATH** of *hw_path* and return that if found. If all attributes are exhausted, then Ignite-UX will find a replacement disk if **allow_disk_remap** is set to true.

I/O Protocol (*protocol***):**
An I/O protocol type. Not all HP-UX releases support all protocol types listed here. Some protocol types listed here may not be supported by any HP-UX release and are included to indicate what the protocol value would be if the protocol is ever supported in the future. Protocols not supported by an HP-UX release will be ignored. These protocol type values may be used to construct a protocol list. That list order may be used by some parameters to indicate a specific prioritization of I/O adapters and devices for install or recovery processing. The valid protocol values are:

fibre_channel
I/O protocol type Fibre Channel (FC). The priority order of FC devices connected via the same I/O adapter is based on hardware path (lunpath hardware path for HP-UX 11.31 and later).

parallel_scsi
I/O protocol type Parallel Small Computer System Interface (SCSI). The priority order of Parallel SCSI devices connected via the same I/O adapter is based on SCSI bus priority order (e.g. SCSI address 7 to 0 followed by 15 to 8).

sas I/O protocol type Serial Attached SCSI (SAS). The order of SAS devices connected via the same I/O adapter is based on physical location. Note that hardware path values should not be used for device selection or recovery device matching since these values are not consistent for the same device from install to install.

usb I/O protocol type Universal Serial Bus (USB). The order of USB devices is not yet defined. There is currently no support for USB devices as a target for install or recovery.

This section contains variables related to advanced features used to control the behavior of I/O configuration during installation and recovery.

allow_disk_remap = *boolean*

Setting this to **true** allows Ignite-UX to substitute a disk that was specified in the configuration files but that does not exist on the system with a disk that does exist and that was not specified to be used, hidden or blocked. This allows for a configuration file generated for one client to be more easily used for a different system that has disks at different hardware paths. This should be used with caution since the remapping algorithm may end up using disks you did not originally wish to use. The remapping algorithm attempts to find substitute disks with the same SCSI ID, and gives priority to finding a substitute for the root disk. If a substitute disk cannot be found, the volume group is either created without that disk or the file system is not created.

The default for this value is **false** for a non-interactive install, and **true** for an interactive install. During an interactive install, you are shown which disks are being substituted and may make changes, or cancel the install.

hide_boot_disk = *boolean*

Setting this to **true** prevents the installation process from allowing the boot disk to be configured and/or "cleaned". This is useful only when the Ignite-UX kernel is booted from a dedicated hard disk that you wish to protect from being modified.

_hp_hide_other_disks

String variable that may be set to one or more space-separated hardware paths or device specifier attributes of disks that should be "hidden" from being configured or otherwise modified during the installation. It allows for more disks to be hidden than that provided by the **hide_boot_disk** keyword.

hw_instance_num = *hw_path class_string [driver] instance_num*
hw_instance_num += *hw_path class_string [driver] instance_num*

Setting this keyword forces a specific instance number assignment for the specified hardware device. This keyword may be used when specific device special file names are desired to produce a client consistent with others regardless of variations in hardware configurations. The following examples assign the instance number 10 to the external bus (SCSI bus) 2/0/1, and instance number 1 to the LAN device 2/0/2. Note that legacy disk device special file names contain the instance number of the parent "ext_bus" device.

```
hw_instance_num = 2/0/1 "ext_bus" "c720" 10
hw_instance_num += 2/0/2 "lan" "btlan" 1
```

It may be very confusing when a configuration file defines the instance number for a LAN device, and also specifies networking information by using the interface name, which has the instance number in the string (for example, **ip_addr**[*lan0*]). In this case, the original/default instance numbers are used because the instance number assignment specified with **hw_instance_num** does not take place until late in the installation process. This confusion may be avoided by using the device's hardware path instead of the interface name (for example, **ip_addr**[*2/0/2*]).

inventory_block_path = *hw_path*
inventory_block_path += *hw_path*

This keyword is used to control Ignite-UX inventory functionality. The hardware paths must refer to an I/O adapter or a device. Ignite-UX will not collect inventory information for devices listed or for devices connected to adapters listed. All adapters used for devices must be listed when devices are connected using multiple paths. Ignite-UX will inventory devices connected to any adapters not listed. Restricting the Ignite-UX inventory process reduces the time required to collect I/O configuration information but will result in some I/O information not being available for install and recovery processing. Blocking inventory at the device level may not provide a significant performance benefit.

Because Ignite-UX does not control kernel boot inventory, some inventory information for blocked devices may be collected. All blocked devices will be hidden and not available for selection during install. To take effect, this variable must be set in the ***INSTALLFS** file using **instl_adm**.

inventory_block_protocols = { "*protocol***" [,** "*protocol*" **]... }**
This keyword is used to control Ignite-UX inventory functionality. Ignite-UX will not collect inventory information for devices connected to I/O adapters of listed I/O protocol types. Restricting the Ignite-UX inventory process reduces the time required to collect I/O configuration information but will result in some I/O information not being available for install and recovery processing. Because Ignite-UX does not control kernel boot inventory, some inventory information for blocked devices may be collected. All blocked devices will be hidden and not available for selection during install. To take effect, this variable must be set in the ***INSTALLFS** file using **instl_adm**.

io_info = *io.info_entry*
io_info += *io.info_entry*
This keyword is used to record details about hardware configuration for use during recovery. Each value corresponds to a complete io.info entry and together they form a complete saved io.info configuration file. These values are intended for Ignite-UX internal use and should not be modified.

File System Configuration

The default disk and file system configuration may be specified as a **partitioned_disk** or a **volume_group**. In a **partitioned_disk** configuration, each disk may have a maximum of two partitions. One partition is a file system and the other is swap space. This configuration is also referred to as the "whole disk" layout. The **volume_group** configuration allows one or more disks to be configured into a Logical Volume Manager (LVM) volume group or VxVM disk group. That volume group may then be split into one or more logical volumes. Each logical volume may be used for either file system, swap or dump space. An LVM volume group may also be split into one or more physical volume groups, with at least one disk in each physical volume group.

The general syntax of the disk and file system configuration is:

```
partitioned_disk
{
    physical_volume disk[hw_path|index|dev_spec]
    {
        physical_volume-attributes...
    }
    fs_partition
    {
        file-system-attributes...
    }
    swap_partition
    {
        swap-attributes...
    }
}
volume_group group_name
{
  volume_group-attributes...
    physical_volume group "pvgname"
    {
        physical_volume disk[hw_path|index|dev_spec]
        {
            physical_volume-attributes...
```

```
            }
          ...
        }
      ...
        physical_volume disk[hw_path|index|dev_spec]
        {
            physical_volume-attributes...
        }
      ...
        logical_volume
        {
            file-system/swap-attributes...
        }
        logical_volume "lvname"
        {
            file-system/swap-attributes...
        }
        ...
    }
```

The keywords and attributes used for disk configuration are described below:

partitioned_disk
Begins the specification of a disk that may be partitioned into one or two simple partitions. The disk may be used for file system, swap, or file system and swap. This is also referred to as the "whole disk" layout.

volume_group *group_name*
Begins the specification in which one or more disks may be put into Physical Volume Groups, a Logical Volume Group, and then split up into one or more logical volumes. The *group_name* must be a quoted string typically of the form **"vg***NN***"** where *NN* is a two-digit number. However, the name may be any string representing a valid directory name.

physical_volume group *pvgname*
Specifies a physical volume group as part of the current **volume_group**. *pvgname* must be a quoted string. All disks listed inside the data structure become a member of the physical volume group; at least one disk must be a member.

physical_volume disk[*hw_path*|*index*|*dev_spec***]**
Specifies the disk to use for the current configuration. *hw_path* is the hardware path of the disk (as reported by **ioscan**). *hw_path* may also be a string variable such as the special variable **_hp_root_disk**, which corresponds to the disk the user chose from the UI. The **_hp_root_disk** variable may also be set in the configuration file, but by default is the **hw_path** of the disk with the highest SCSI priority.

hw_path may also be the ***** character, in which case all unconfigured disks are included in the **volume_group**. In the case of a partitioned_disk, one unconfigured disk is used. Another form of *hw_path* may be **= N*, where *N* is the number of disks that should be put in the **volume_group**. An alternative to using an explicit hardware path is to use an integer (or integer variable) *index* value. If an *index* value is used, it should be in the range of 0 to **num_disks-1**.

If a *dev_spec* type is used, refer to the discussion regarding Device Specifier in the I/O Configuration section.

Following **physical_volume**, may be a block of physical_volume-attributes surrounded by **{ }** brackets. These attributes are described in the following sections.

fs_partition
In a **partitioned_disk** configuration, this keyword starts the specification for the file system partition of the disk. The block following this keyword is surrounded by **{ }** brackets and

consists of file system attributes. File system attributes are described in the following sections.

swap_partition
In a **partitioned_disk** configuration, this keyword starts the specification for the swap partition of the disk. The block following this keyword is surrounded by **{ }** brackets and consists of subset of the file system attributes, namely the **size**, **usage**, and optionally the **mount_point**. These attributes are described in the following sections.

logical_volume
logical_volume *cplx_string*
Defines a logical volume that should be part of the current **volume_group**. A volume group may contain any number of logical volumes. The logical volume may optionally be given a name by specifying a *cplx_string* after the **logical_volume** keyword. If a name is not supplied, the default of "lvolX" is used where X is the creation number of the logical volume. Following the **logical_volume** statement is a block of file system and swap attributes.

Physical Volume Attributes
Following the specification of a disk using the **physical_volume** keyword, a block of attributes surrounded by **{ }** brackets may be specified. See *mkfs*(1M) for details. These attribute keywords are:

nsect = *number*
Specifies the number of sectors per track as *number*.

ntrack = *number*
Specifies the number of tracks per cylinder as *number*.

rpm = *number*
Specifies the revolutions per minute of the disk as *number*, where *number* is converted to RPS before passing it to the **newfs** command.

interleave_factor = *number*
This attribute used to control whether **mediainit** should be run on the disk. It is no longer supported to run **mediainit** on the disk, and the command has been removed from Ignite-UX. Therefore this attribute has no effect.

File System/Swap Attributes
The following attributes may be specified inside the blocks following the **fs_partition**, **swap_partition**, and **logical_volume** keywords.

usage = HFS
usage = VxFS
usage = SWAP
usage = SWAP_DUMP
usage = DUMP
usage = Unused
Specifies the volume should be used as one of: **HFS** (hierarchical file system), **VxFS** (journaled file system), virtual memory **SWAP**, both virtual memory **SWAP** and **DUMP** (for system core dump storage), **DUMP** alone, or **Unused**.

Note that the **/stand** file system must be **HFS** on PA-RISC systems. On Itanium®-based systems, **/stand** may also be VxFS. Also note that **usage** is restricted to the following values for each of these disk and file system configurations: **fs_partition** may be either **HFS** or **VxFS**; **swap_partition** must be **SWAP**; and **logical_volume** may be either **HFS**, **VxFS**, **SWAP**, **SWAP_DUMP**, **DUMP**, or **Unused**.

size = *number*
size = remaining
size = remaining | *maxsize*

size = *minsize* | **remaining**
size = *minsize* | **remaining** | *maxsize*
Specifies the size of the volume/file system to be *number*. In the case the size is set to **remaining**, the volume is assigned the disk space that is left after allocating all other volumes. If *maxsize* is specified in conjunction with **remaining**, the remaining disk space, up to *maxsize*, is assigned to the volume. If *minsize* is specified, the size of the volume is at least that size if there is sufficient space, and is larger, up to the *maxsize* value, if possible. *number*, *minsize*, and *maxsize* may be in units of bytes, or may be in MB or KB (for example: 100MB, 102400KB).

Ignite-UX uses the disk-space impact information supplied with the software selections in determining the final size requirements for volumes. Ignite-UX does not let a volume size be lower than the space required to load the selected software. If a volume size is specified smaller than the software requires, the size will automatically be increased. Ignite-UX attempts to ensure that some amount of free space is left on each volume after the installation is complete.

The algorithm used when calculating the software impact to volumes is as follows:

• The minimum size that a volume is allowed to be is the software impact, plus 12% (or the file system **minfree** percentage if set).

• In addition to this minimum size, a percentage is added to the volumes to ensure sufficient "breathing room" after the system is installed. This percentage is by default 10%, but may be controlled by the configuration file variable **_hp_addnl_fs_free_pct**. If the disk capacity is insufficient to accommodate this additional free space, this additional percentage value is continually reduced by 1/2 until a value is found that allows all the volumes to fit. If the volumes do not fit with 0% additional space, the install is not allowed to proceed.

size = *percent* **%free**
size = *amount* **free**
size = *minsize* | **remaining** | *percent* **%free**
size = *minsize* | **remaining** | *amount* **free**
Specifies the size of the volume/file system to be relative to the expected amount of data for that file system. The size may be specified as a the desired percentage of free space, or the number of megabytes desired to be free after the installation.

When the size is specified as a range, the size of the volume is allowed to be adjusted down to the *minsize* value in order to fit any other volumes into the amount of disk space available. Some examples follow:

size = 100MB
size = 150MB | remaining | 20% free
size = 30MB free
size = remaining

mount_point = *cplx_string*
Specifies the directory the file system is mounted under as *cplx_string*. Specifying a **mount_point** for a swap volume is not necessary, but may be performed so that the swap volume may be uniquely identified if it is to be overridden later in the configuration file.

minfree = *number*
Sets the percentage of the disk to reserve for root-only use to *number*.

file_length = SHORT
file_length = LONG
Sets the file system filename length; the default is **LONG**. This keyword applies to **HFS** file systems only; **VxFS** file systems are always capable of long filenames.

blksize = *number*
Sets the file system block size to *number*. This keyword applies to both **HFS** and **VxFS** file systems. Acceptable values of *number* for **HFS** file systems are 4096, 8192, 16384, 32768, and 65536; the default is 8192. Acceptable values of *number* for **VxFS** file systems are 1024, 2048,

4096, and 8192; the default is 1024.

fragsize = *number*
Sets the file system fragment size to *number*.

ncpg = *number*
Sets the number of cylinders per group to *number*.

nbpi = *number*
Sets the number of bytes per inode (inode density) to *number*.

rotational_delay = *number*
Sets the rotational delay factor for the file system to *number.* (See *tunefs*(1M).)

largefiles = *boolean*
Specifies that largefiles are to be enabled or disabled for this volume. This is supported for both **HFS** and **VxFS** file systems.

version = *number*
For HP-UX B.11.23 and later clients, this keyword specifies the disk layout version number of the **VxFS** file system to create for a mount point. If this keyword is not specified for a **VxFS** mount point, then the default value for the release is used. For B.11.23, this value is **5**; for B.11.31 this value is **6**. This is only supported for **VxFS** file systems.

The following attributes are recognized when the volume is an LVM volume (see *lvcreate*(1M)). Those that apply to VxVM volumes are noted as such, otherwise they are ignored if the disk group is VxVM.

bad_block_relocate = *boolean* | **none**
Sets the bad block relocation policy to 'y' if *boolean* is **true** or 'n' if *boolean* is **false**. If **none** is given, the bad block relocation policy is set to 'N'.

contiguous_allocation = *boolean*
Sets the contiguous allocation policy to 'y' if *boolean* is **true** or 'n' if *boolean* is **false**. Also applies to VxVM volumes.

stripes = *number*
stripes = *
Specifies the data for this logical volume should be striped across multiple physical disks in the volume group. The number of disks used to stripe the data may be provided by *number*, or if the data should be striped across all the disks in the group, ***** may be specified. Volumes that must be contiguous (**/**, **/stand**, and primary **swap**/**dump**) cannot be striped. There must be two or more disks configured in the volume group in order to specify a striped volume in that group. Also applies to VxVM volumes.

stripe_size = *number*
Specifies the data size of each stripe. This is only used when the logical volume is specified to be striped via the **stripes** keyword. The default **stripe_size** is equal to the file system block size (which is 8KB by default). The allowed values for *number* depend on the **usage**. If the **usage** is LVM then the allowed values for *number* must be a power of 2 and be in the range of 4KB to 32MB. If the **usage** is VxVM the allowed values can be any size.

disk[*hw_path*|*dev_spec***]**
Specifies one or more references to a disk that may be listed inside a logical volume definition. Doing so will force that volume to have data allocated only from the disk or disks listed. The disks are listed inside the logical volume must have been listed as **physical_volumes** in the same volume group in which the logical volume exists. Unless the desired disks are listed within the logical volume definition, the logical volume could have data allocated from any of the disks in the volume group. The **vgdisk[***index***]** keyword may be used for this purpose as well. Also applies to VxVM volumes.

minor_number = *number*
Determines the logical volume number that is assigned to the associated logical volume which in turn determines the minor number value for the device file. *number* may range from 0 to 255. If the **minor_number** is not specified and the logical volume name is specified and is the form **lvol***X*, then the value *X* is used to determine the minor number. Otherwise, the value is assigned in the order the volumes are created.

Volume Group Attributes

Volume groups may be either LVM or VxVM. Support for these two types of volume groups is release-dependent.

usage = LVM
usage = VxVM
Determines type of volume group; the default is LVM.

Within the **volume_group** block, the following LVM attributes may be set. See *vgcreate*(1M) for details on meanings and defaults. These attributes are ignored for VxVM volume groups.

max_physical_extents = *number*
Sets the maximum number of physical extents that may be allocated from any of the physical volumes in the volume group.

max_logical_vols = *number*
Sets the maximum number of logical volumes that the volume group is allowed to contain.

max_physical_vols = *number*
Sets the maximum number of physical volumes that the volume group is allowed to contain.

physical_extent_size = *number*
Sets the number of megabytes in each physical extent, where *number* may range from 1 through 256.

minor_number = *number*
Sets the minor number of the volume group control file. Use of this option is cautioned if you do not fully understand the consequences. The value of *number* may range from 0 to 255. If this attribute is set, that *number* is used. Otherwise if the volume group name is of the form **"vg***NN***"** and *NN* is in range and not already declared by a **minor_number** attribute, the value of *NN* is used. Otherwise the first unused value in the acceptable range is used.

System Identity and Network Configuration

The configuration file may set attributes that control the default values for system identity and network configuration. Most of these keywords may have values that are used during the install, and may have **final** values that are applied to the system after the installation. This is to support situations when the client is not being installed on the network of its final destination.

When keywords are not proceeded with the **final** keyword, the information is used during the installation, and may also be applied to the final client depending on the boolean value of the **is_net_info_temporary** keyword. If the **final** keyword is specified, the information is applied to the final system, and overrides any temporary information that was used during the installation. Note that **instl_adm** may be used to set some of these temporary values with command line options. The keywords recognized for this are:

[**final**] **system_name =** *cplx_string*
Sets the client's **hostname** and **uname** value to *cplx_string*.

[**final**] **ip_addr =** *cplx_string*
[**final**] **ip_addr**[*lan_dev*] **=** *cplx_string*
Sets the client's default IP address to *cplx_string*. If the IP address is to be used on a specific LAN device other than the default, *lan_dev* may be specified within square brackets. *lan_dev* may be either a string representing the interface name, "lan0" for example, or it may be the hardware path to the LAN device, 2/0/2 for example.

[**final**] **netmask =** *cplx_string*
[**final**] **netmask**[*lan_dev*] **=** *cplx_string*
Sets the client's default netmask, as used by *ifconfig*(1M), to *cplx_string*. *lan_dev* may be used to specify to which LAN interface is applicable.

[**final**] **broadcast_addr =** *cplx_string*
[**final**] **broadcast_addr** [*lan_dev*] **=** *cplx_string*
Sets the broadcast address for the specified LAN device.

dhcp_class_id = *cplx_string*
dhcp_class_id [*lan_dev*] **=** *cplx_string*
Sets the class ID that is used when trying to obtain information for the specified LAN interface from a DHCP server. Specifying a **dhcp_class_id** limits the responses from DHCP servers to those that have information for the same class that is specified. This is used to isolate clients into different classes. Using the **dhcp_class_id** is especially useful when temporary IP addresses to be used for installs come from a different server than addresses used by other systems on the network. The **dhcp_server** keyword may also be used for this purpose.

dhcp_server = *cplx_string*
dhcp_server [*lan_dev*] **=** *cplx_string*
Specifies the DHCP server from which the internet address lease should be obtained. Specifying the server prevents the client from accepting offers from any other server on the network. Normally this is not set and the client is allowed to accept any offer.

dhcp_misc_opts = *cplx_string*
dhcp_misc_opts [*lan_dev*] **=** *cplx_string*
Can be used to specify miscellaneous options to the **dhcpclient** command when it is used to obtain the IP address lease. Common options specified with this keyword are **-G** and **-l.** The **-G** option may be used in conjunction with the **dhcp_class_id**, which causes the **dhcpclient** command to refuse any offers from servers that do not supply the matching class ID string in their reply offer. Normally servers that do not have the same class ID do not respond; however, some vendor's implementations do respond, thus the **-G** option may be used to force filtering of these replies on the client side. The **-G** option only works with DHCP servers that supply the class ID in their response. Servers known to supply the class ID are HP-UX B.11.11 and later **bootpd**. The **-l** option may be used to specify debug output from the **dhcpclient** command. Other options are documented in the *dhcpclient*(1M) man page on HP-UX releases B.11.23 and later. For example, to set the debug output level:

```
dhcp_misc_opts = "-l 6"
```

[**final**] **route_gateway** [*index*] **=** *cplx_string*
[**final**] **route_destination** [*index*] **=** *cplx_string*
[**final**] **route_count** [*index*] **=** *number*
These keywords are use to configure the client's network routing information. If no routing information is given, the default is to set up a "default" route though the local host. If one or more static routes are to be configured, the **route_gateway** must be set. If **route_destination** is not set, it defaults to the string "default". If **route_count** is not set, it defaults to 1 if the **route_gateway** is not the same as the client's IP address. It defaults to 0 if it is the same as the client's IP address.

index number should be different for each set of route parameters to be configured; it may be thought of as an array index. The first route specified should have an *index* of 0.

[**final**] **dns_domain =** *cplx_string*
[**final**] **dns_nameserver** [*index*] **=** *cplx_string*
[**final**] **dns_search** [*index*] **=** *cplx_string*
Sets the domain name services (DNS) domain, one or more name servers, and optionally, one or more search domains. The *index* parameter to the **dns_nameserver** and to the **dns_search** keywords allows more than one name server and search domain to be configured. The first server

and search domain should be assigned using an *index* of 0, and the second 1, etc. See *resolver*(4) for details.

[**final**] **nis_domain =** *cplx_string*
[**final**] **nis_server =** *cplx_string*
[**final**] **wait_for_nis_server =** *boolean*
Sets NIS domain and NIS server information. The **nis_server** keyword is optional when configuring NIS. The default for **wait_for_nis_server** is TRUE, meaning that the client waits at boot time until the server can be contacted.

[**final**] **ntpdate_server =** *cplx_string*
Sets the network time protocol server that the client uses to keep the clock synchronized. See *ntpdate*(1M) for details. This is currently not used during the installation, rather after the client is finished installing.

is_net_info_temporary = *boolean*
Set to **true** indicating the network information supplied during the install is not assumed to be correct for the system after it is installed. When set to **true** and no **final** parameters have been set, the **set_parms** and **auto_parms** utilities try to obtain the final information using the **dhcpclient** command using the dynamic host configuration protocol (DHCP), or from you if DHCP is not available. In addition, if the information used during the installation was originally obtained from a DHCP server, the IP address lease is returned at the end of the installation.

If set to **false**, which is the default, the network information used during the install process is assumed to be default after the installation is complete. This means that if the information was obtained from a DHCP server, the lease information obtained is kept and used when the client boots. If it was not obtained via DHCP, the ENABLE_DHCP variable in **/etc/rc.config.d/netconf** is set to '0' for the specific LAN interface. This prevents **auto_parms** from overriding the supplied network information with any possible information from a DHCP server.

run_setparms = *boolean*
Set to **true** indicating the host name is removed from the final **/etc/rc.conf.d/netconf** file, which will cause the **set_parms** application to run upon first boot querying you for networking information.

server = *cplx_string*
Used to specify the default IP address of the install server, when the installation source is the network. This is an Ignite-UX server from which the installation process reads the configuration files.

sd_server = *cplx_string*
Specifies the IP address of the system to use as the source for the Software Distributor (SD) **swinstall**, or **swm** utilities. See *swinstall*(1M), or *swm*(1M) (for HP-UX B.11.31 or later). This information is used only if the **sw_source** does not specify the **sd_server** information.

sd_depot_dir = *cplx_string*
Sets the default depot directory to be used by the **swinstall**, or **swm** (on HP-UX B.11.31 or later) utilities. This depot directory is the location the HP-UX software has been stored on the **sd_server** by the **swcopy** utility. This information is used only if the **sw_source** does not specify the **sd_depot_dir** information.

root_password = *cplx_string*
Specifies *cplx_string* as being the encrypted password to be stored in the final **/etc/passwd** or **/etc/shadow** file for the root user, or in the protected password file **/tcb/files/auth/r/root** if an archive of a trusted system has been loaded.

timezone = *cplx_string*
Specifies *cplx_string* as the time zone to be stored in the **/etc/TIMEZONE** file on the final client. For HP-UX releases beginning with B.11.11, this also specifies the time zone data for the

/etc/default/tz file.

serial_number = *cplx_string*
customer = *cplx_string*
order_number = *cplx_string*
These three keywords are used to supply information that is displayed on the client's manifest. They are informational only.

Control Parameters
The following parameters may be used to control the behavior of the installation process:

run_ui = *boolean*
Specifies whether the installation process should be interactive. This may be set to **false** if you also set **control_from_server = true**, or if the configuration file specifies all the necessary information needed to perform the installation. If any information is missing or if any problems are found with the configuration, the installation switches to interactive mode.

In the case of a non-interactive installation, you are given a default of ten (10) seconds to stop the process. See **env_vars** to adjust this timeout value and to specify whether warnings conditions are allowed during non-interactive install.

Setting **run_ui = false** and not setting **control_from_server = true** should be done with extreme caution since the potential for accidental reinstallation of system is possible. This would cause all data on one or more disks to be lost. Be aware that a network boot server that is not restricting access may be accidentally booted from. Therefore it is best to either restrict access to the boot server (see *instl_bootd*(1M)), or surround the **run_ui = false** statement with a condition that makes it specific to the client's **LLA[]**. The client's LLA[] may be retrieved from the boot ROMs using the **lan_addr** command, or by using the **lanscan** command.

control_from_server = *boolean*
Setting this value to **true** causes the client to enter a mode where it will wait for the install server to instruct it how to proceed with the installation (via the **/opt/ignite/bin/ignite** application). The default is **false**, so the UI runs on the client without interaction on the server. This keyword only sets the default; you have the opportunity to use either mode during the install on the client.

When used in combination with the **run_ui=false** keyword, the client may enter a mode to wait for server instructions immediately after booting, assuming that it may obtain all necessary network information.

halt_when_done = *boolean*
Setting this value to **true** halts the client installation when it is complete instead of rebooting from the disk that was just installed; the default is **false**.

clean_all_disks = *boolean*
Setting this to **true** causes the installation process to write over all disks, even those that are not configured for a file system. Its purpose is to remove prior file system information and make all the unused disks appear as "new".

This keyword should almost always be left as the default of **false**.

recovery_mode = *boolean*
Setting this to **true** indicates to Ignite-UX that the contents of the archive being used to install the client has files that are specific for the client being installed. When **true**, many files that are specific to a client are preserved as-is from the archive. When set to **false**, those same files are copied from the **/usr/newconfig** directory and modified for the specific client being installed.

In addition, when set to **true**, the device files and kernel from the archive are preserved and not removed and recreated. Most of the effects of this keyword are visible and modifiable in the script: **/opt/ignite/data/scripts/os_arch_post_1.** Search for the **if** statement

regarding **RECOVERY_MODE** in this script.

This keyword defaults to **false**, and is set to **true** in configuration files created by **make_net_recovery**.

disable_dhcp = *boolean*

Setting this to **true** in the ***INSTALLFS** file prevents the process from attempting to contact a DHCP server for networking information. This may be useful if you do not use DHCP and want to avoid the delay of searching for an Ignite-UX server. Setting this to **true** in any configuration file other than ***INSTALLFS** allows the initial DHCP request to be performed, but the resulting client has DHCP disabled. This may be useful if you are using **BOOTP** to supply the IP address and host name of the clients, but do not require ongoing maintenance of a lease. The **BOOTP** protocol is a subset of DHCP, and may be used to satisfy a DHCP query without the maintenance of a lease.

error_if_bad_sw = *boolean*

Setting this to **true** turns a non-existent **sw_source WARNING** into an **ERROR**. The default is **false**.

use_expert_ui = *boolean*

Setting this to **false** used to select the deprecated guided/wizard interface mode. Now the only mode is advanced. Ignite-UX will force this keyword to **true**, and if found **false** will issue a warning that the setting will be ignored.

sd_command_line = *cplx_string*
sd_command_line += *cplx_string*

Used to specify options to all **swinstall**, or **swm** (on HP-UX B.11.31 or later) operations performed during an install. The options specified are added to the default options that are required to perform the software installation. It is advised to include space characters at each end of the *cplx_string* to avoid options running into each other and resulting in syntax errors. For example:

```
sd_command_line += " -xpatch_match_target=true "
```

sd_secrets = *cplx_string*

Setting this should only occur when the file **/var/adm/sw/security/secrets** on the source depot server(s) has been modified to provide additional security. When this is performed, all clients that have access to that server must have an identical **secrets** file. The **sd_secrets** keyword allows you to specify the contents of the **secrets** file that are used by the Ignite-UX server performing the installation.

sysadm_message = *cplx_string*

Sets the message displayed to you when the client first boots the cold install process. This message is only recognized when stored in the configuration file that exists in the first 8KB of the ***INSTALLFS** file. The **instl_adm -a** option to make this easy to change.

error = *cplx_string*
error += *cplx_string*

Used to indicate an error condition such that the *cplx_string* is displayed to you and the installation process is not allowed to proceed. The use of the **+=** operator adds a new message separate from any others. The **=** operator clears out any other messages previously added so that *cplx_string* is the only message seen. If the **=** operator is used with an empty *cplx_string* (""), all error messages using this mechanism are cleared.

warning = *cplx_string*
warning += *cplx_string*

Sets the warning message to **cplx_string**, similar to the **error** keyword described above; however, **cplx_string** is considered a warning message and does not prevent the installation from proceeding.

note = *cplx_string*
note += *cplx_string*

Sets the note message to **cplx_string**, similar to the **error** and **warning** keywords described above; however, **cplx_string** is considered a notice message and will not prevent the installation from proceeding.

mod_kernel = *cplx-string*
mod_kernel += *cplx-string*

Used to add drivers or tunable parameters to the client's kernel that is built during the Ignite-UX process. The format of *cplx-string* may be either "driver" or "tunable value".

The largest of any tunable parameter that exists in either the **/stand/system** file or that is specified by you is used. Beginning with the B.11.23 release, if a tunable is not found in **/stand/system**, it is compared with the default value as reported by **kctune**. Ignite-UX does not compare the values of formulas to discrete numeric values, two formulas, or hexadecimal values, in order to determine which is larger. It issues a note message stating that it assumes the last **mod_kernel** keyword parsed is larger (regardless of whether it is a formula or discrete numeric value) and applies it. There is no bounds checking done on tunable parameters.

Decimal values should be in the range from 0 to 2147483647 (2^31-1). The shell that is used to evaluate these values uses signed 32-bit arithmetic for decimal values, and problems occur when values exceed this limit. If larger values are needed, the values must be converted to hexadecimal because the shell does not evaluate them.

Note that any allowable syntax supported by the **config** command may be used in **mod_kernel** keywords. This includes formulas. For example, it is possible to use:

```
mod_kernel += "ninode (20+8*MAXUSERS+NPTY+" + ${"%d" _inc} + ")"
```

This example assumes **_inc** has been initialized to some value in the configuration file.

The **=** operator overrides any prior global **mod_kernel** assignments. The **+=** operator adds to any prior settings. Notice that **mod_kernel** statements may also be associated with a **sw_sel** definition. The **=** operator does not have any effect on **mod_kernel** assignments made in a **sw_sel**.

Beginning with the B.11.23 release, the format for *cplx_string* arguments is enhanced. There are additional formats:

mod_kernel += "tunable *name value***"**
mod_kernel += "module *name [state]***"**

The first performs the same as it did previously with the addition of the keyword **tunable**. The second describes how kernel modules are added to the client. The optional *state* argument has one of four values: **static**, **auto**, **loaded** and **best**. No syntax checking is performed for this value. See *kcmodule*(1M) for more information.

Actions involving **mod_kernel** are performed before those for both **set_kernel** and **rm_kernel**.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

set_kernel = *cplx-string*
set_kernel += *cplx-string*

Setting this keyword is the same for drivers as the **mod_kernel** keyword in that it adds the driver to the kernel. For tunables it differs from **mod_kernel** in that it sets the tunable to an arbitrary value as defined by *cplx-string*. No comparisons or checks are performed with prior or default values. Actions involving **set_kernel** are performed after those for **mod_kernel**, but before those for **rm_kernel**.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

rm_kernel = *cplx-string*
rm_kernel += *cplx-string*

Setting this keyword removes the driver or tunable from the **/stand/system** file. For drivers, this implies that the driver is removed from the kernel. For tunables, this implies the tunable reverts back to its default value. Actions involving **rm_kernel** are performed after those for both **mod_kernel** and **set_kernel**.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

vxvm_hostid = *cplx_string*

Setting this identifier writes it to all VxVM disks on a client when they are first added to a disk group. When a client starts, it looks at all the disks and compares its VxVM identifier to the identifier stored in **/etc/vx/volboot**. For each disk that matches that identifier, the client enables access.

vxvm_version = *cplx_string*

This keyword causes all VxVM disk groups on a client to be created with the on-disk data structures for the VxVM version specified. The allowable values as well as the defaults in cases where this keyword is not specified are defined as follows:

HP-UX release	Default	Allowed
B.11.11	3.5	3.5
B.11.23	3.5	3.5, 4.1, 5.0
B.11.31	4.1	4.1, 5.0

This keyword is applied globally and cannot be set on a per-disk group basis. Valid values for **vxvm_version** currently are: "3.5", "4.1", and "5.0".

If **vxvm_version** is encountered multiple times in a configuration during a cold install, the lowest value of the keyword will be used as the global value.

The behaviour of this keyword is slightly different during a recovery. The **save_config** command creates the configuration file that describes any VxVM disk groups that will be recreated during a recovery. This command sets **vxvm_version** to the highest VxVM version used to create a disk group presented to the system it is running on.

env_vars = *cplx_string*
env_vars += *cplx_string*

Sets a string of variables of the form *variable=value* that is sourced into the environment for the duration of the installation process. The format of *cplx_string* should be one variable assignment per line. The value of the assignment does not need to be quoted (unless quotes need to be part of the value). These variable assignments are stored in the file */tmp/install.vars* and is left on the client after the installation.

The use of the **+=** operator is recommended to ensure any prior settings are kept and not discarded as is the case when the **=** operator is used.

Some special environment variables that may be set inside **env_vars** and are recognized by the cold installation process are:

INST_NET_RESPONSE_TIMEOUT=*seconds*

Booting an installation client from the network sets the amount of time the client waits for you to respond before it reboots assuming the client booted from the installation server accidentally. Setting *seconds* to 0 (zero) disables the timeout and the client does not prompt for a response.

INST_ALLOW_WARNINGS=*seconds*
Setting this environment variable is useful for non-interactive installation sessions when warnings about disks containing data cause the installation to switch to interactive mode. Setting *seconds* to **1** causes all warnings to be ignored and the installation to proceed. Setting *seconds* to greater than **1** allows you that number of seconds to read the warning and stop the installation by pressing **< return >**.

INST_BATCH_MODE_TIMEOUT=*seconds*
Sets the amount of time you have to interrupt a non-interactive installation; the default is 10 seconds.

INST_ENABLE_NETWORK=*1*
Set in the case where an automated install is to be performed from physical media and the network interface needs to be enabled. If the host name and IP address are not set, the install process will attempt to contact a **DHCP**/**bootpd** server to get that information.

LOADFILE_RETRY_COUNT=*number*
Used to change the default number of times the internal **loadfile** command retries a failed attempt to retrieve data from the server or media. Usually this retry mechanism is used to overcome **tftp** transfer problems; the default value is 5.

Software Source and Selections

Ignite-UX supports a sophisticated mechanism for specifying the software that is available for loading during installation. It supports loading from multiple sources of various types, primarily SD depots, and **tar**/**cpio**/**pax** archives. The configuration files contain the mapping from what software selections you see to the source that contain them. This allows you to operate on the full list of software selections without needing to know the location of the source of the selections.

The three constructs for handling software are: **sw_source**, **sw_sel**, and **sw_category**. A **sw_source** specifies an SD depot, or an access method to an Ignite-UX server containing archives. The **sw_sel** specifies the software contained in the SD depot, or specifies the path to an archive on the server or media. There is typically one **sw_sel** definition per software bundle or archive. The **sw_category** is a mechanism for grouping the multitude of **sw_sel**s available. The UI treats some categories specially by putting them on the basic screen or a under a special selector.

When setting up an SD depot that uses software *bundles*, as all software shipped by HP should, you may simply run the **make_config** command may be run against that depot to generate the configuration file describing the software available in terms of those constructs. Setting up a configuration file that allows access to an archive requires manual editing, and the use of the **archive_impact** command.

The general structure for specifying software is as follows:

```
sw_source src-tag-string
{
    source-attributes...
}
sw_category cat-tag-string
{
    description = string
}
sw_sel sel-tag-string
{
    sw_source = src-tag-string
    sw_category = cat-tag-string
    selection-attributes...
}
init sw_sel sel-tag-string=boolean
```

sw_source *src-tag-string* **{** *source-attributes* **}**

The software source specifies an SD depot or an access method to an archive. The *src-tag-string* is used by **sw_sel** to identify which **sw_source** is applicable. The attributes that may be within a **sw_source** clause are:

description = *cplx_string*

Provides a description, typically one line, of the source contents. This is used to provide feedback to you.

source_type = "NET" | **"MT"** | **"DSK"**

Used to override the default source type, which corresponds to the type of media from which the client was booted. The source type strings are: network (**"NET"**), magnetic-tape (**"MT"**), and CD/DVD (**"DSK"**). Regardless of the boot sources, the default **source_type** is **"NET"** if an Ignite-UX server is being used, which may be overridden by this keyword.

source_format = SD | **ARCHIVE** | **CMD**

Specifies the general format of the source. This determines which information in the **sw_source** clauses used when accessing the source. The default is **CMD**.

When set to **SD**, and **source_type** is **NET**, the **swinstall**, or **swm** (on HP-UX B.11.31 or later) command is used to load from the specified **sd_server** and **sd_depot_dir**. If **source_type** is **MT**, and the tape contains a LIF volume a the beginning, the **swinstall** command expects to find the depot as the third file on the tape.

As of the HP-UX B.11.31 release, the **swm** command is used to load software, and it does not support installing from a tape containing a LIF boot volume. Because bootable tapes for Itanium®-based systems do not contain a LIF boot volume, **swinstall** will not expect to find a depot on the tape. For Itanium®-based systems, and for B.11.31 or later releases, a second tape containing the depot will be needed. When using a second tape for the depot the **change_media** keyword should be set to **true**.

When set to **ARCHIVE**, and **source_type** is **NET**, either the **ftp_source**, **remsh_source**, or **nfs_source** keywords must be supplied and any **sw_sel**s that refer to this source must specify an **archive_path**.

When set to **CMD**, only the specified ***_cmd**, ***_script**, and ***_kernel** keywords are used and executed at the appropriate time. This allows for a customized method of loading the represented software or for setting kernel parameters.

source_path = *cplx_string*

Sets this keyword to the default device file path to access the tape or CD for non-network sources. If not specified, it will default to the device from which it was booted.

change_media = *boolean*

Setting this keyword indicates whether Ignite-UX should prompt you to change media before loading this source for non-network sources. The **description** information is used in the prompt so you know which media to insert.

sd_server = *cplx_string*

Sets the IP address of the SD (**swinstall**, or **swm**) server to use when **source_format=SD**. This value overrides the global **sd_server** keyword when loading from this source.

sd_depot_dir = *cplx_string*

Sets the depot directory to be used in combination with the **sd_server** value when running the **swinstall**, or **swm** commands to load the associated software when **source_type** is **"NET".** When **source_type** is **"DSK", sd_depot_dir** sets the depot directory to be used on the CD when running **swinstall**, or **swm** to load the associated software. (If not specified, "/" is assumed.)

sd_use_ui = *boolean*
Indicates if the **swinstall**, or **swm** (on HP-UX B.11.31 or later) command should be run interactively, on the client console when loading from this source. The default is false indicating it runs non-interactively.

sd_command_line = *cplx_string*
sd_command_line += *cplx_string*
Use to specify additional options to be passed to the **swinstall**, or **swm** commands. This keyword does not normally need to be specified. These options are appended to those supplied by any global **sd_command_line** keyword, and can therefore override those set there on a per-**sw_source** basis.

ftp_source = *cplx_string*
Use when **source_format=ARCHIVE** and the archive that is to be loaded may be obtained via the **ftp** command. The format of *cplx_string* is **"***user***@***system***:***passwd***"**. For example: "anonymous@11.23.34.45:Ignite-UX". Use of an anonymous **ftp** login is recommended so that unencrypted passwords are not available in the configuration file.

remsh_source = *cplx_string*
Use when **source_format=ARCHIVE** and the archive that is to be loaded may be obtained via the **remsh** command. The format of *cplx_string* is **"***user***@***system***"**. This requires the server to have **.rhosts** access enabled for the "root" user on all installation clients for the *user* account specified, or that the string "+ root" be placed in the *user* account's **.rhosts** file. Because of the need to have open access to the server, this method only works in a very relaxed security environment and should be used with caution.

nfs_source = *cplx_string*
Use when **source_format=ARCHIVE** and the archive that is to be loaded may be obtained by NFS mounting the specified directory. The format of *cplx_string* is **"***Ignite-UX server***:/***mount-point***"**. **The** *Ignite-UX server* must export the specified *mount-point* to all installation clients. The **/etc/exports** file may specify read-only permissions to aid in security. The path to the archive relative to *mount-point* is provided in the associated **sw_sel**'s **archive_path**.

load_order = *number*
Use to specify the order in which this **sw_source** should be accessed relative to any other sources. The default load order is 5. For sources that provide the core HP-UX operating system, regardless of whether it is an SD or **ARCHIVE** source, the **load_order** must be set to 0 (zero) to ensure it is loaded first, to invoke special handling within Ignite-UX.

pre_load_cmd = *cplx_string*
post_load_cmd = *cplx_string*
post_config_cmd = *cplx_string*
final_cmd = *cplx_string*
Use to specify a shell command string that should be executed at the prescribed points during the installation. See the **Command and Script Execution Hooks** section for details.

pre_load_script = *cplx_string*
post_load_script = *cplx_string*
post_config_script = *cplx_string*
final_script = *cplx_string*
Use these keywords specify a shell script to be loaded from the Ignite-UX server using **tftp** and executed at the prescribed points during the installation. See the **Command and Script Execution Hooks** section for details.

sw_category *cat-tag-string* **{ description =** *string* **}**

Use the **sw_category** definition to provide a grouping mechanism for **sw_sel** definitions to reference. The UI uses the **sw_category** to help you browse the software more easily. There are six values of *cat-tag-string* that the UI treats specially. Software in these groups is represented in special locations and by special methods in the UI. These values are: **"HPUXBaseOS"**, **"HPUXEnvironments"**, **"Languages"**, **"LanguagesUI"**, **"OpEnvironments"**, and **"UserLicenses"**. The only attribute associated with a **sw_category** is a **description**.

sw_sel *sel-tag-string* **{** *selection-attributes* **}**
init sw_sel *sel-tag-string* **=** *boolean*
init sw_sel *sel-tag-string* **=** *PARTIAL*

Use the **sw_sel** keyword to define a software selection for installation. A software selection typically refers to an SD bundle or a single **tar**/**cpio**/**pax** archive. Each **sw_sel** must reference a **sw_source**, which provides the information needed to access the actual software. A **sw_sel** may be selected or unselected in the configuration file by using the **init** keyword and setting it to a boolean value (see EXAMPLES). If the **init** keyword is not used in an assignment, the UI is not able to change the value. A **sw_sel** may be assigned the value "PARTIAL" to indicate that it is to be selected but that no **also_select** references within the **sw_sel** definition should be selected or unselected. When a **sw_sel** that has **also_select** references is set to a boolean value, then all **sw_sel** items it references with an **also_select** statement are also selected (when the **sw_sel** is true) or unselected (when the **sw_sel** is false).

The *sel-tag-string* is used to reference the **sw_sel** in subsequent assignments. It is possible to assign multiple tag strings to a single **sw_sel** in order to provide aliases for a single selection. In this case, any tag may be used when referencing it within a configuration file. The syntax for specifying multiple tags is:

sw_sel *tag1* | *tag2* | ...**{***selection-attributes***}**

The *selection-attributes* that may be specified for a **sw_sel** are described below as follows:

description = *cplx-string*

Provides a one-line description of the software to be displayed along with the tag string or the **display_name**.

display_name = *cplx-string*

Specifies a more descriptive name than *sel-tag-string* for the UI to display as the product name for a non-**HPUXEnvironments** category **sw_sel**. For an **HPUXEnvironments** category **sw_sel**, it is displayed as the environment description.

sw_source = *src-tag-string*

Specifies the **sw_source** from which this software selection is to be installed. The *src-tag-string* must match the tag string used when defining the source. This is a required attribute of a **sw_sel**.

sw_category [+]= *cat-tag-string*

Specifies the software category with which this software selection is to be grouped. Using the "+=" operator allows for placing the sw_sel into multiple categories.

archive_path = *cplx-string*

Specifies the file path to the associated **tar**/**cpio**/**pax** archive. This is only used when the associated **sw_source** sets **source_format=ARCHIVE**. If the **sw_source** uses the **nfs_server** access method, this path must be relative to the NFS mount point. For **ftp_source** or **remsh_source** access methods, the path may be relative to the given account's home directory, or an absolute path if desired.

If **source_type="MT"** then the **archive_path** may be a quoted number string, which is used to position the tape device prior to extracting the data. This would provide the *number* argument to the command: **mt fsf** *number*, where *number* is the number

of files to skip from the beginning of the tape. Typically **archive_path** is **"1"** for PA-RISC tapes and **"22"** for Itanium® tapes. Note that for Itanium® tapes **archive_path** is actually ignored for archive extraction and the archive is located using tape label information (see *ansitape*(1M)).

If **source_type="DSK"** then **archive_path** is used to access an archive on the CD when mounted as a file system. In this case, **archive_path** must be a path relative to the top directory of the CD.

When using an archive to contain the core HP-UX operating system (a full system archive), the archive must be packaged using relative path names. This allows Ignite-UX to extract it to the mounted disk. In addition, the load_order of the **sw_source** must be set to 0 (zero). Non-core operating system archives may be packaged using either relative or absolute path names.

archive_type = [{**compressed**|**gzip**}] {**tar**|**cpio**|**pax**}

Specifies the type of archive that **archive_path** references. The archive may be an uncompressed **tar**, **cpio**, or **pax** archive, or may be compressed using either **compress** or **gzip**. The compression type, if specified, may be either of the keywords **compressed** or **gzip**. The second keyword must be **tar**, **cpio**, or **pax**. The **pax** option is only supported for the B.11.23 release and later. For example: **archive_type = gzip tar**.

If **source_type="MT"** the archive will be extracted with a block size dependent on the archive type. If the **archive_type** is **tar**, the block size will be 10240 bytes. If the **archive_type** is either **cpio** or **pax**, the block size will be 5120 bytes.

sd_software_list = *cplx-string*

Specifies a list of SD software objects (bundles, products, filesets, etc.) that are to be installed using the **swinstall**, or **swm** (on HP-UX B.11.31 or later) command. One or more software objects may be specified.

impacts = *directory number*
impacts += *directory number*

Use to specify the amount of disk space impact the associated software has on the given directory. Multiple **impacts** statements may be specified. Normally there is one statement for each top-level directory into which the software is installed. However, the directory level may be more specific if desired. If the **=** operator is used, any previous *number* assigned for *directory* specified for this **sw_sel** is replaced. If the **+=** operator is used, the *number* specified is added to any prior *number* specified for the given *directory* associated with this **sw_sel**.

This information is critical to the way Ignite-UX performs volume size calculations that are based relative on the amount of disk space used by each software selection.

The **make_config** command automatically generates **sw_sel** statements with this information. The **archive_impact** command may be used to generate the **impacts** attributes when manually creating **sw_sel** statements that represent a **tar**/**cpio**/**pax** archive.

mod_kernel [+]= *cplx-string*

Allows the addition of kernel drivers or tunable parameters when this **sw_sel** has been selected for loading. Similar to the global **mod_kernel** keyword.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

set_kernel [+]= *cplx-string*

Allows addition of kernel drivers or setting tunable parameters when this **sw_sel** has been selected for loading. Similar to the global **set_kernel** keyword.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

rm_kernel [+]= *cplx-string*
Allows removal of kernel drivers or tunable parameters when this **sw_sel** has been selected for loading. Similar to the global **rm_kernel** keyword.

Note: Spaces should only be used within strings associated with *kernel keywords. Tabs are not acceptable characters.

locale = {*string1***,** *string2***, ... }**
Sets a list of language locales that are supplied to the system when this **sw_sel** is loaded. This list is used by the UI to give you a selection of languages to choose from based upon what software has been selected. The format of the strings listed is: *locale***:***description*. Where *locale* is something to which the LANG environment variable could be set, and *description* is a string that briefly describes the locale.

visible_if = *boolean*
Allows the **sw_sel** to be hidden from the UI when set to false. The default is true.

manifest_info = *cplx-string*
Specifies that the text in the client manifest template tagged with the given string should be printed as part of the client's manifest. See *print_manifest*(1M) for details.

corequisite [**+**]**=** *tag_string*
Indicates that the **sw_sel** referred to by *tag_string* should be loaded with this **sw_sel**. Multiple corequisites may be listed using one **corequisite** statement per line with the **+=** operator.

exrequisite [**+**]**=** *tag_string*
Defines an exclusive relationship between the current **sw_sel** and the one referenced by *tag_string*. This prevents the referenced **sw_sel** from being selected any time that this **sw_sel** is selected and vice-versa. The **+=** operator may be used to define multiple exrequisites.

exrequisite = sw_category
Specifies that this **sw_sel** is to be exclusive with all other software in the same category (or categories). This attribute may be used in addition to other exrequisites and does not override them even though the **+=** operator is not used.

load_with_any = *tag_string* [[| *tag_string*]...]
Specifies that when any of the **sw_sel**s listed are selected, this **sw_sel** should be selected for installing as well. Multiple tags may be listed separated by the | character.

load_with_any ~ *tag-regexp* [[| *tag-regexp*]...]
load_with_any ~ *category***.***tag-regexp ...*
Specifies that when the **~** operator is used rather than **=**, the **sw_sel** tags listed are treated as *fnmatch*(3C) regular expressions that may be used to match any selection fitting the given pattern. Note this operator uses *fnmatch*(3C) pattern matching expressions. Other uses of the **~** operator in the configuration file are with extended regular expressions. This difference is for compatibility with some existing data. The *category* string may be specified, with a "." separator, to limit matches to those selections in the specified software category.

load_with_all = *tag_string* [[**&** *tag_string*]...]
Specifies that when all the **sw_sel**s listed are selected, this **sw_sel** is selected as well. Multiple tags may be listed separated by the **&** character.

If multiple **load_with_*** keywords are specified, the list of tag strings is added to any already listed for the respective keyword.

also_select = *tag_string* [[| *tag_string*]...]
also_select ~ *tag-regexp* [[| *tag-regexp*]...]
also_select ~ *category*.*tag-regexp* ...

The **also_select** keyword allows for one **sw_sel** to effect the selection of other **sw_sel**s. It is different than a **corequisite** in that software selected (or unselected) by this mechanism can be individually unselected (or selected) without effecting the selection status of the referencing **sw_sel**. In contrast, a **corequisite** cannot be unselected if the item declaring the **corequisite** is selected.

When selecting (setting to true) a **sw_sel** ("A") that declares an **also_select** on another **sw_sel** ("B"), the **sw_sel** "B" will also be selected. Similarly, when **sw_sel** "A" is unselected, then **sw_sel** "B" will also be unselected.

The argument syntax of this keyword is the same as the **load_with_any** keyword.

pre_load_cmd = *cplx_string*
post_load_cmd = *cplx_string*
post_config_cmd = *cplx_string*
final_cmd = *cplx_string*

Specify a shell command string that should be executed at the prescribed points during the installation when this **sw_sel** has been selected. See the **Command and Script Execution Hooks** section for details.

pre_load_script = *cplx_string*
post_load_script = *cplx_string*
post_config_script = *cplx_string*
final_script = *cplx_string*

Specify a shell script that should be loaded from the Ignite-UX server using **tftp** and executed at the prescribed points during the installation if this **sw_sel** is selected. See the **Command and Script Execution Hooks** section for details.

Command and Script Execution Hooks

Ignite-UX supports mechanisms for the commands and scripts you supply to be inserted into various stages in the installation process. There are two types of keywords that the configuration file supports. The keywords that end in **_cmd** may be used to embed scripts directly in the configuration file. This is useful when the command desired to be executed is short and may be inserted directly into the configuration file without compromising the maintainability of the configuration file. The keywords that end in **_script** are used to reference files to be loaded from the Ignite-UX server using the *tftp*(1) service and executed as a stand-alone command.

With the exception of the **pre_config_cmd** keyword, the execution hooks may be specified in three different areas of the configuration file. These areas are:

- Within a sw_source definition.
- Within a sw_sel definition.
- At the global level.

Although only the **=** assignment operator is shown with these keywords, the **+=** operator can also be used to add a script or command fragment to any existing assignments made to the same keyword in the same context. The **=** operator overwrites any prior assignments made for that keyword. A separate context for each keyword is kept for the global level, each unique **sw_source** and each unique **sw_sel**.

The exit code returned by a script results in different handling by the Ignite-UX installation process:

0 Ignite-UX does not display any additional messages and proceeds

1 Ignite-UX displays an additional ERROR message and proceeds

2 Ignite-UX displays an additional WARNING message and proceeds

4 Ignite-UX displays an additional ERROR message and terminates

other
The results are undefined

The point in time that the associated commands or scripts are executed is described below.

pre_load_cmd = *cplx_string*
pre_load_script = *cplx_string*
The global **pre_load** commands and scripts are executed prior to any software installation, and after the file systems have been created. Those within a **sw_source** or a **sw_sel** definition are executed prior to loading any software from the respective source. Those associated with a **sw_source** are executed before of any associated with a **sw_sel**.

post_load_cmd = *cplx_string*
post_load_script = *cplx_string*
The global **post_load** commands and scripts are executed after all software has been installed, but prior to the final client reboot, and software configuration step. Those within a **sw_source** or **sw_sel** definition are executed after all software from the associated source has been installed and prior to the global **post_load** commands and scripts. Those associated with a **sw_sel** are executed ahead of any associated with a **sw_source** (opposite order of **pre_load**). This is the correct hook to use if customizations to the kernel via the **/stand/system** file are to be made beyond what is capable through the **mod_kernel** keyword.

post_config_cmd = *cplx_string*
post_config_script = *cplx_string*
The **post_config** commands and scripts are executed after all software has been configured and the final client kernel has been built and booted. It is safe to execute commands in this environment that may have dependencies on the version or modifications of the kernel that was put into place prior to rebooting. The **post_config** commands and scripts associated with a **sw_source** or **sw_sel** are executed first, and then the global versions last.

Note that if **is_net_info_temporary** is **true**, then certain commands are not usable at this time. If it is **true**, then certain networking files (e.g. **etc/hosts**) are gone, and so commands like **swconfig** do not work.

final_cmd = *cplx_string*
final_script = *cplx_string*
The **final** commands and scripts are executed as the very last step before the final client reboot or halt. In fact, these scripts and commands are executed after the log file has been closed and the installation's overall status has been determined. Thus, the exit code from a **final_script** does not influence the outcome of the installation. Also, any information logged from these commands and scripts is not written into the log file, but is written to the client console. It is safe to execute commands in this environment that may have dependencies on the version or modifications of the kernel that was put into place prior to rebooting. The **final** commands and scripts associated with a **sw_source** or **sw_sel** are executed first and the global versions last.

Note that if a temporary host name/IP address is used during the installation, all the commands and scripts, including the **final_cmd**/**script**, are executed while that temporary information is in effect. If you require the final networking information to be in effect, then you may need to have it run at the first client boot by loading the script in the **/sbin/rc*.d** directory and having it remove itself when completed.

pre_config_cmd = *cplx_string*
A list of semicolon separated commands can be set to execute prior to the creation and mounting of file systems. It also runs prior to the Ignite-UX UI. There are almost no commands available at this stage; not even a shell. Any commands needed have to be loaded by using the **loadfile** command that exists in this environment. For command usage, type "loadfile -?" from a recovery shell.

It is inadvisable to do any sort of software loading via invoking **swinstall** directly from any command or script hook. This can lead to various problems such as missing disk space impacts, ensuring the software is fully configured, and getting kernel software loaded before the kernel is built. It is much better to use the standard software mechanisms provided with the **sw_source** and **sw_sel** attributes as required. Furthermore Ignite-UX exports environment variables that control **swinstall** behavior, and this can have unexpected results.

In addition to the hooks above that may be specified in an Ignite-UX configuration file, Ignite-UX provides a way for applications that are installed using an archive to supply scripts to be run. After installing any software from an archive, Ignite-UX recursively searches the **/var/adm/sw/products** directory for any scripts entitled **iux_postload** and **iux_postconfig**. Any **iux_postload** scripts provided by the archive(s) are run after all software sources are installed and before the client kernel is built. Any **iux_postconfig** scripts found are run after all software is installed and the system is running on the final kernel. The order is similar to the **post_load_cmd** and **post_config_cmd** keywords described previously . These scripts are not executed if they were installed from an SD depot, because the regular SD control scripts are run.

When Ignite-UX is provided with a final system host name, it searches the **/sbin/ch_hostname.d** directory, if it exists, for any executable files/scripts. It then runs the executables in that directory with a single argument of the new host name. Note that this host name argument may be different than the one currently in use, and takes effect after the pending reboot. The scripts are run in the order in which the **ls** command sorts them. This hook may be useful for products that need to modify configuration files based on the host name and would normally do this in an SD configure script. A recommendation for software product developers is that any host name specific tasks be done using this hook, and that the SD control script call the script in this directory as well. Note that SD configure scripts are not run when using Ignite-UX to install a client from a client archive. The scripts in **/sbin/ch_hostname.d** must be re-runnable any time the host name changes. In the HP-UX B.11.11 release, the **set_parms** command will also implement this mechanism to be used on initial system setup, as well as, host name changes.

When a **tar**/**cpio**/**pax** archive is used to supply the core HP-UX operating system (a full system archive), you should to use the two scripts that are supplied with Ignite-UX. These scripts should be associated with the **sw_sel** that references the system archive. See the example configuration file, **/opt/ignite/data/examples/core.cfg**. There is one **post_load_script**, **/opt/ignite/data/scripts/os_arch_post_l**, and one **post_config_script**, **/opt/ignite/data/scripts/os_arch_post_c**. These shell scripts may require customizations to meet your needs. See the comments provided in the files for more details.

Configuration File Parsing Order and Precedence

In general, the configuration files are parsed in the order that they are listed in the INDEX file **cfg** statement selected. In addition, the information stored in the first 8KB of the ***INSTALLFS** file and the file entitled **config** in the client-specific directory are parsed after those listed in the **INDEX** file. Configuration files that are parsed later have precedence over any earlier files and may override values set in earlier files. Thus, the client-specific configuration file has highest precedence.

Tape and CD media are in *lif*(4) format and contain an **INDEX** file. Typically the only file referenced by the **INDEX** file is entitled **CONFIG**. It is this **CONFIG** file, in addition to the first 8KB of the ***INSTALLFS** file, that make up the configuration information for these media types.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXAMPLES

Setting the Default Configuration Using Logic Expressions

When configurations are saved as a named configuration, the UI creates a new **INDEX** file **cfg** clause for that saved configuration. It may be desirable to create a logic expression to determine which saved

configuration is to be applied to a new client. This may be performed by setting the **cfg** clause to true/false inside the ***INSTALLFS** file using the **-f** option with the **instl_adm** command. The ***INSTALLFS** file is the proper place to retain this information. If put in an **INDEX** file, the UI does not preserve this type of information the next time a save operation has completed.

```
# Select the "Two disk cfg" if the system has two or more disks.
cfg "Two disk cfg" = (num_disks >= 2)

# For a system with only one disk and a small amount of memory,
# select the "small system configuration".
num_disks == 1 & memory < 64MB
{
   cfg "small system configuration" = true
}
```

Using a Software Archive

If a set of files exists that is packaged in a **tar**/**cpio**/**pax** archive and you want it installed during the installation process, you should create a new configuration file similar to the following example. This creates a new software selection representing this source. It also conditionally selects this software for loading if the root disk size is more than 1800MB.

```
sw_source "Per-Discipline Packs" {
   description = "Software Apps for Individual Disciplines"
   source_format = archive
   nfs_source = "14.12.99.113:/var/opt/ignite/archives"
}
sw_category "Disciplines" {
   description = "Software for Individual Disciplines"
}
sw_sel "EE pack" {
   description = "Software Apps for Electrical Engineering"
   sw_source = "Per-Discipline Packs"
   archive_type = gzip tar
   archive_path = "11.11_700_ee.gz"
   sw_category = "Disciplines"
   sw_category += "EE-Software"
   # Use the /opt/ignite/lbin/archive_impacts tool to create the
   # impacts size statements below:
   impacts = "/var" 12568Kb
   impacts = "/usr" 23468Kb
   impacts = "/" 2Kb
}
init sw_sel "EE pack" = (disk[_hp_root_disk].size > 1800Mb)
####
```

The contents above would be stored in a file such as **/var/opt/ignite/data/Rel_B.11.11/ee_apps_cfg**, and then added to the **INDEX** file using the command:

```
/opt/ignite/bin/manage_index -a \
   -f /var/opt/ignite/data/Rel_B.11.11/ee_apps_cfg
```

Scheduling a post_config_script To Be Run

The **/var/opt/ignite/config.local** file is a convenient place to set the **post_config_script** keyword to run the scripts that should always be run, or that may be run based on a logic expression. Also, this is a convenient place to put kernel modifications that you want performed on all clients installed. The following example schedules one script to always be run, one to be run if the system is an S890, and increases the **maxuprc** kernel tunable for all clients:

```
post_config_script += "/var/opt/ignite/scripts/do_our_setup"
HARDWARE_MODEL == "9000/890" {
  post_config_script += "/var/opt/ignite/scripts/S890_setup"
}
mod_kernel = "maxuprc 100"
```

AUTHOR

Ignite-UX and the configuration file syntax were developed by Hewlett-Packard Company.

FILES

/var/opt/ignite/INDEX

/var/opt/ignite/clients/*LLA***/CINDEX**

The **INDEX** and **CINDEX** files are used on the Ignite-UX server to define the available configurations, and the configuration files that comprise each configuration.

/opt/ignite/data/Rel_*release*

/var/opt/ignite/data/Rel_*release*

For each supported HP-UX release, Ignite-UX creates a directory to store the configuration files defining the defaults for each release. In addition, a **gzip-compressed tar** archive entitled **SYSCMDS** is stored in the **/opt/ignite/data/Rel_***release* directory, and is the mini system that is laid down on the disk to supply the commands needed to install the rest of the software. The **/var/opt/ignite/data/Rel_***release* directories are the recommended location to contain the configuration files created locally that are not part of the standard Ignite-UX product.

/var/opt/ignite/config.local

This configuration file is a place holder for local modifications to the HP-UX defaults. It is typically included in every **INDEX** file **cfg** clause after the default configuration files.

/var/opt/ignite/saved_cfgs

This is the directory the UI uses to store configuration files that you request to be saved as standard configurations. The **/var/opt/ignite/INDEX** file is modified to reference the files in this directory.

/opt/ignite/data/scripts

/var/opt/ignite/scripts

These two script directories are the recommended locations for scripts that are referenced by configuration files. The first path is used by scripts delivered as part of Ignite-UX. The second path is the recommended location for scripts you create.

/opt/ignite/data/examples

Contains example configuration files. See the comments in the files for details.

SEE ALSO

lifinit(1), lifls(1), archive_impact(1M), bootpd(1M), instl_adm(1M), instl_bootd(1M), lvmmigrate(1M), make_config(1M), manage_index(1M). swcopy(1M), swinstall(1M), swm(1M) (HP-UX B.11.31 or later), lif(4), ignite(5).

NAME

ansitape - ANSI standard magtape labels

DESCRIPTION

An ANSI-labelled tape starts with a volume header. This header specifies the volume name and protection, the owner of the volume, and the ANSI label standard level that the tape conforms to.

Every file on the tape has a header, some data blocks, and a trailer. A tape mark follows each of these elements. At the end of the tape, two tape marks follow the trailer, to indicate logical end-of-tape.

If a file is too large to be copied onto one tape, it may be continued on another tape by modifying the trailer section.

VOLUME HEADER

Field	Width	Example	Use
VOL1	4	VOL1	Indicates this is a volume header.
Label	6	VAX1	The name of the volume.
Access	1	space	Volume protection. Space means unprotected.
IGN1	20		<< ignored >>
IGN2	6		<< ignored >>
Owner	14	Joe User	The name of the user.
IGN3	28		<< ignored >>
Level	1	3	ANSI standard level.

Owner The owner field is 14 characters in ANSI labels. IBM labels cut the owner field to 10 characters. The IGN2 field is 10 characters on IBM-format tapes.

FILE HEADERS

Field	Width	Example	Use
HDR1	4	HDR1	Identifies first file header.
Name	17	FILE.DAT	Leftmost 17 characters of filename.
Set	6	VAX1	Name of volume set this file is part of.
Vol Num	4	0001	Number of this volume within volume set.
File Num	4	0001	Number of file on this tape.
Generation	4	0001	Like a major release number.
Gen Version	2	00	Version of a file within a release.
Created	6	b86001	The date of file creation.
Expires	6	b86365	Date file expires.
Access	1	space	File protection. Space means unprotected.
Blockcount	6	000000	Number of blocks in the file.
System	13	OS360	The name of the software system that created the tape.
IGN	7		<< ignored >>

Name — The filename may be up to 17 characters in IBM labels, and ANSI labels before standard level 3. On ANSI level 3 and after, the HDR4 record provides overflow storage for up to 63 more characters of filename.

Set Name — On multi-reel tape sets, a name identifying the set as a whole. Normally, this is just the volume name of the first reel in the set.

Generation — Like a major release number. The version field is a version within a generation. On VAX/VMS systems, these two fields are mathematically related to the (single) version number of disk files.

Created — The date the file was created. This is a six character field, where the first character is always a space. The next two are the year. The final 3 are the day within the year, counting January 1st as day 1.

Blocks — The number of blocks in the file. In HDR1 records, this is always zero. The corresponding EOF1 or EOV1 contains the number of tape blocks written in the file on the current reel.

Field	Width	Example	Use
HDR2	4	HDR2	Second file header.
Rec Format	1	D	Record format.
Blk Length	5	02048	Tape block size.
Rec Length	5	00080	Record size.
Density	1	3	Recording density code.
Vol Switch	1	0	1 if this is a continuation of a file from a previous reel.
Job	17	user/program	See following notes.
Recording	2	space	Unused in 9-track tapes.
Car Control	1	space	See following notes.
Blocking	1	B	See following notes.
IGN	11		<< ignored >>
Offset	2	00	Bytes to skip at front of each block.

Rec Format

A single character indicating what type of records are provided. The codes are

Code	Meaning
F	Fixed-length
D	Variable up to rec length
V	IBM code for variable
U	Unknown

Job The name of the job (username in Unix) right-padded to 8 characters, a slash (/), and the job step (program name in Unix) right-padded to 8 characters. This identifies where the JCL was when this file was created.

Carriage Control

Normally a space, indicating that the records do not contain carriage control information. When printed, each record is placed on a separate line. If an 'A' is used, the first character of each record is presumed to be a Fortran carriage-control character. VAX/VMS also uses 'M' to indicate that carriage-control is embedded as part of the data. This is usually used in the case of binary files.

Blocking

The B indicates that as many records as will fit are placed in a physical tape block. Records do not cross block boundaries. A space indicates only one record per physical tape block.

The HDR3 and HDR4 labels are not written on IBM tapes. ANSI allows, but does not require, these labels.

Field	Width	Example	Use
HDR3	4	HDR3	Third file header.
OS	76		Operating-system dependent.

OS This field is reserved for the use of the operating system that created the file. Other operating systems are supposed to disregard HDR3 records. On VAX/VMS, this record contains the RMS file description.

Field	Width	Example	Use
HDR4	4	HDR4	Fourth file header.
Name 2	63		Name continuation from HDR1.
Unknown	2	00	Unknown, fill with 00.
IGN	11		<< ignored >>

Name 2

On ANSI tapes, if the filename is longer than 17 characters, the first 17 are placed in the HDR1 record. The next 63 are put in HDR4. Filenames longer than 80 characters are truncated. Note that it is not required to have a HDR3 record in order to have a HDR4.

FILE TRAILING LABELS

These labels are written after a tape file. For every label written at the head of the file, there will be a corresponding label at the tail. Except for the *block count* field in HDR1, the only difference is in the name of the label. If we have reached the logical end of the file, the characters *HDR* in the headers are replaced by the characters *EOF* in the trailing labels. If we are not at the logical end of the file, but are merely pausing at the physical end of tape before continuing on another reel, the *HDR* characters are replaced by *EOV* (end-of-volume).

The *block count* field of HDR1 was initially recorded as 000000. When the trailers are written, the block count is changed to indicate the number of tape data blocks written. A file that is continued over several volumes maintains separate counts for each reel.

RECORD FORMATS

The two basic record formats are fixed and variable.

Fixed format uses records that are all constant length. This is the case with VAX/VMS executable images (record length = 512). It is also used by IBM systems for text files, with a record length of 80 (card images). The record size field of HDR2 tells how long each record is.

With fixed-length records, the blocksize is usually selected to be some multiple of the recordsize. As many records as will fit are placed in each block. Since records do not (normally) span physical tape blocks, extra space at the end of a block is wasted.

Variable-length records are used by VAX/VMS for text files. The Unix program *ansitape(1)* also turns Unix text files into variable-length tape files. With this format, the record length specified in HDR2 is an upper limit.

Each record is preceeded by a 4-digit (zero-filled) byte count. The count included the digits themselves, so the minimum valid number is 0004. These four digits specify how long the record is. The data follows the digits, and is in turn followed by the digits for the next record.

When writing, *ansitape* checks to make sure that there is enough room in the tape block for the next record. If the record (including its length digits) won't fit, the current block is sent to the tape, and a new block is started. Unused space at the end of the tape block is filled with circumflex (^) characters.

NAME

ignite − HP-UX configuration, installation, and recovery manager

SYNOPSIS

/opt/ignite/bin/ignite [**-recovery**] [**-install**] [**-S**] [*XToolkit-Options*] [*client1, client2, client3, ...*]

DESCRIPTION

Introduction

ignite is part of the Ignite-UX product, a client-server application that provides the ability to configure, install, and recover HP-UX clients.

The current version of Ignite-UX is the "C" version. The various HP-UX releases that the "C" version is capable of installing include:

C.7.3-C.7.?	11.31, 11.23, and 11.11
C.7.0-C.7.2	11.31, 11.23, 11.11, and 11.00
C.6.9-C.6.10	11.23, 11.11, and 11.00

The **ignite** command is the graphical user interface (GUI) of this client-server application. It provides the ability to build software configurations and use these configurations to install HP-UX clients. In addition, **ignite** provides the ability to create network recovery archives and use these archives for system recovery. A "Tutorial and Demo" is available from the Actions menu of the GUI to introduce key features of the product and how each works.

Options

The **ignite** command recognizes the following options:

-recovery If one or more clients are specified on the command line, the "Create System Recovery Archive" action will automatically be invoked using the *first client* in the list. Once the action has completed, the user is returned to primary Ignite-UX console and may select the next available client on which to perform a subsequent action.

-install If one or more clients are specified at the command line, the "Install Client/New Install" action will automatically be invoked using the *first client* in the list.

-S Directs **ignite** to use **ssh** based protocols instead of **remsh** based protocols when communicating with remote systems for initiating recovery sessions or client reboots.

XToolkit Options

The **ignite** command supports a subset of the standard X Toolkit options to control the appearance of the GUI. The supported options are: **-bg**, **-background**, **-fg**, **-foreground**, **-display**, **-name**, **-xrm**, and **-synchronous**. See the *X*(1) manual page for a definition of these options.

client1,client2,client3,..

A list of one or more clients which will be used to filter the set of clients displayed by the user interface. If a client is specified and its corresponding client directory (in **/var/opt/ignite/clients**) does not exist, a note for each client that is missing is displayed as the user interface is started.

The Ignite-UX product enables users to:

- Control multiple simultaneous install sessions from a single server.
- Perform a single standalone installation of a target client from a terminal user interface running on the target machine.
- Initiate non-interactive installations from the client or from the server.
- Install multiple applications at the same time as the base operating system.

- Load software from non-Software Distributor (SD) sources (**tar, cpio,** or **pax**) as well as from SD sources in one session.
- Customize a client so that it is "ready to go" as soon as the install session completes.
- Specify user-supplied scripts to be executed at pre-defined points both during the installation process and after.
- Create a "golden system image" that may be unpacked directly to a target machine over the network.
- Save customized configurations and quickly apply those configurations to multiple target clients.
- Capture a snapshot of the currently loaded software along with a complete hardware inventory.
- Create client-specific recovery archives that may be stored on the network and loaded by Ignite-UX in the event of root disk failure or corruption.
- Create bootable system recovery tapes.

Hardware/Software Requirements

For performing installations, the following hardware is required to set up an Ignite-UX server. The server requires either a graphics display or a display redirected to another *X*(1) windows system. The redirection is accomplished by setting the **DISPLAY** environment variable. For example, at the Korn Shell or Posix Shell type:

```
export DISPLAY=<server>:0.0
```

Ignite-UX server requirements:

- An HP-UX system running HP-UX 11.11, 11.23 or 11.31.
- An X11 display server (workstation, X-terminal, PC running an X server, etc.). This may be the same system as above.
- Sufficient disk space to load Ignite-UX and any software depots and/or archives to be used during the install.
- Access to the Ignite-UX tool set. The tool set may be loaded onto any of the above clients.
- Tape/CD-ROM to load Ignite-UX and any software depots planned for distribution on the server.
- Network access to any clients to be installed. Client and server must be on the same subnet if the initial boot of the client is planned to be over the network.

Ignite-UX Server Setup: Overview

1. Install HP-UX.
2. Install the Ignite-UX software.
3. Set up the core HP-UX software.
4. Add additional applications (optional).
5. Run **ignite** to complete the configuration and to start the process.

   Note: All operations are executed as "privileged user" on the Ignite-UX server. Except where noted, all commands referenced here are located in **/opt/ignite/bin**. (See the "privileges" manpage for more information on the privileged user.)

Ignite-UX Server Setup: Details

1. Install HP-UX

Refer to the *HP-UX 11i Installation and Update Guide* and *Read Before Installing or Updating to HP-UX 11i* for the revision of HP-UX 11i that you wish to install or update for instructions.

2. **Install Ignite-UX software.**

   The Ignite-UX tool-set is contained on the HP-UX application set of CD-ROMs. The software bundles are named as follows: **Ignite-UX-11-31**, **Ignite-UX-11-23**, **Ignite-UX-11-11**, and **Ignite-UX-11-00**. Each software bundle contains the Ignite-UX tools in addition to the data files required for support of the particular HP-UX release indicated by the bundle name. You may load one or more of the Ignite-UX bundles onto your server depending on which releases of HP-UX you plan on installing onto clients. The entire set of releases is supported with the **IGNITE** bundle.

   Once the application CD-ROM containing Ignite-UX has been mounted, you may use the **swinstall** command to load the desired Ignite-UX bundles. For example, the command below would load the support needed for installing HP-UX 11.23 onto clients assuming the CD-ROM is mounted at **/cdrom**:

   ```
   swinstall -s /cdrom Ignite-UX-11-23
   ```

3. **Set up core HP-UX software.**

   Before Ignite-UX may be used, you must configure the software for it to load onto the clients. Since both SD sources and archive (non-SD) sources (**tar, cpio,** or **pax**) may be used, both cases will be considered separately below.

   For SD operating system software:

   For Ignite-UX to use an SD source, you must have a registered SD depot available and an Ignite-UX configuration file generated from that depot.

   If you already have an SD depot containing the core HP-UX software, you may enable Ignite-UX's access it by using the **make_config** and **manage_index** commands.

   You may also use the Server Setup wizard available from the Welcome screen of the Ignite-UX server and from the Actions menu. The wizard is self-explanatory.

   Enabling (or updating) an existing depot

   If you already have an SD depot available, or if you have made changes to a depot about which Ignite-UX knows, you may use the **make_config** and **manage_index** commands to generate a configuration file that Ignite-UX will use to access the depot. For example:

   ```
   make_config -s server:/depot_1123 \
     -c /var/opt/ignite/data/Rel_B.11.23/core_cfg

   manage_index -a -f \
     /var/opt/ignite/data/Rel_B.11.23/core_cfg
   ```

   If at a later time, you modify the contents of a depot (for example, using swcopy to add software), the same **make_config** step will need to be rerun in order for Ignite-UX to be aware of the modifications.

   Note: The **make_config** command only operates on software that is contained in a *bundle*. If you have a depot that has products not in a bundle, you may run the **make_bundles** command on the depot prior to running **make_config**.

   For non-SD (archive) operating system software:

   Ignite-UX has the capability of loading a client from an archive image taken from a client that represents a standard configuration. This method gives significantly faster install times, but may not be as flexible as using an SD source.

   You will first need to generate the archive image of a client in the desired state. It is recommended that the **/opt/ignite/data/scripts/make_sys_image** script be used to accomplish this task.

Once an archive image is created, a configuration file that represents the location and attributes of that image must be created before Ignite-UX may use it.

A sample of a config file that may be used with a core archive may be found at:

/opt/ignite/data/examples/B.11.23.archives.cfg

The comments in this example file describe where to copy the file and what to change in the file to make it reference your archive and work in your environment.

4. **Add additional applications (optional).**

If you have other software you would like to have loaded during the client installation, you may create configuration files similar to what was done for the core operating system software by using **make_config** and **manage_index** or by using an example configuration file and modifying it.

For SD application software:

Run the following commands for each depot you plan to load SD software from during the installation. The **make_config** command only handles SD software that is packaged in bundle form. If the SD depot you want to use has software not contained in a bundle (for example, a collection of patches), the **make_bundles** command may be used to create bundles in the depot.

make_bundles */depots/11.23_patches*

```
make_config -s server:/depots/11.23_patches \
  -c /var/opt/ignite/data/Rel_B.11.23/patches_cfg

manage_index -a -f /var/opt/ignite/data/Rel_B.11.23/patches_cfg
```

To make a depot containing compilers available which are already in bundles (no need to use **make_bundles**):

```
make_config -s server:/depots/compiler \
  -c /opt/ignite/data/Rel_B.11.23/compilers_cfg

manage_index -a -f /opt/ignite/data/Rel_B.11.23/compilers_cfg
```

The depot server (in this example <*server*>) should be replaced with the server on which you have the SD software. The **make_bundles** command must be run on the same client the depot exists. If the depot is not on the Ignite-UX server, you may need to copy the **make_bundles** command to the depot server and run it there.

Note: The **make_config** command will need to be rerun each time new software is added or modified in the depots.

make_config constructs Ignite-UX config files which correspond to SD depots. When an SD depot is used as part of the Ignite-UX process, it must have a config file which describes the contents of the depot to Ignite-UX. This command may automatically construct such a config file given the name of an SD depot on which to operate. This command should be run when adding or changing a depot which will be used by Ignite-UX.

manage_index is used to manipulate the **/var/opt/ignite/INDEX** file. This utility is primarily called by other Ignite-UX tools but may also be called directly.

For non-SD application software:

If the source is not an SD depot, the **make_config** command is not applicable. You will need to create a unique config file that includes the non-SD software. A sample of a config file that performs a non-core archive may be found at:

/opt/ignite/data/examples/noncore.cfg

The comments in this example file describe where to copy the file and what to change in the file to make it reference your archive and work in your environment.

5. **Run ignite to complete the configuration and start the process.**
   On the server execute **`/opt/ignite/bin/ignite`**. This will start the Ignite-UX server program.

   Complete the Configuration:
   When the GUI display titled "Ignite-UX (host name)" appears after the Welcome screen is dismissed, perform the following:

   a) Choose Options: Server Configuration...

   b) Look over both the Server Options and the Session Options to verify they are suitable.

About the Screen: 'Configure Booting IP Addresses'

Booting Clients: xx.xx.xx.xx to xx.xx.xx.xx
These IP addresses are used to initially boot the target clients. They are used until the client is assigned one of the DHCP-assigned addresses. One address is required for each simultaneous boot. Typically one to three are needed, depending on the usage.

DHCP Addresses: xx.xx.xx.xx to xx.xx.xx.xx
These IP addresses are used during the operating system download and application loading. These addresses are in use for most of the Ignite-UX download to a target machine.

One address is required for each simultaneous download. You should set more, if the addresses are to be assigned permanently.

DHCP Class ID
The unique name for the DHCP server that serves these DHCP Addresses. This is not necessarily the install server.

Do not apply the class ID unless you are configuring the install server to be a DHCP server.

DHCP Addresses are Temporary
If these DHCP Addresses are used only for performing installs, and the clients will get reassigned new addresses when deployed, keep this field set.

If you want to set up the Ignite-UX server as a departmental DHCP server, in which case the IP address leases are permanent and isolated to the department's DHCP server, set this field to false.

Boot a Client that Supports Network Boot

If the client you plan to install is running HP-UX, you may use the **`bootsys`** command from the server to remotely reboot the client to run Ignite-UX.

If the client is new or disabled such that **`bootsys`** cannot be used, then you can boot it over the network. To perform a network boot, open the console for that client and enter the appropriate command. You will find the exact boot ROM commands for your client in the *Installation and Update Guide HP-UX 11i* for your specific operating system release.

PA-RISC Clients:

If you need further help with the boot process, enter:

Main Menu: Enter command > **`help boot`**

`1)` Obtain the IP address of the Ignite-UX server you intend to use.

2) Cycle the power (perform a cold reset) on the client to bring it to a known state.

To stop selection process, press and hold the **`< escape >`** key.

During the boot sequence, status messages are displayed on the client console. Depending on what type of machine, server or workstation model, a boot administration menu and/or firmware prompt may appear.

3) Boot the client using your Ignite-UX server's IP address by entering this command at the client console:

Main Menu: Enter command > **boot lan.n.n.n.n install**

where: **n.n.n.n** is the IP address of the Ignite-UX server.

The client then begins to load the install kernel (ignite the client) from the network server.

Note: To search for Ignite-UX servers, type the following at the client console (workstations only):

Main Menu: Enter command > **search lan install**

The list of servers that you can boot the client from is displayed with the corresponding IP addresses and is similar to:

```
Searching for potential boot devices(s)...  on Path LAN
This may take several minutes.

To discontinue search, press any key (termination may not be
immediate).

Path Number    Device Path                      Device Type
-----------    -----------                      -----------
P0             LAN.15.1.46.117.3.254    lp2     100/Full Dx
P1             LAN.15.1.41.70.3.254     lp4     100/Full D
```

You may need to run the **nslookup** command on another running system to determine which address corresponds to your Ignite-UX server.

Itanium-Based Clients:

1) Cycle the power (perform a cold reset) on the client to bring it to a known state.

During the boot sequence, status messages are displayed on the client console. Depending on what type of machine, server or workstation model, the EFI Boot Manager menu appears and looks similar to:

```
EFI Boot Manager ver 1.10 [14.60]

Please select a boot option

HP-UX Primary Boot: 0/2/2/0.0.0.0
EFI Shell [Built-in]
Boot option maintenance menu
Security/Password Menu

Use ^ and v to change option(s). Use Enter to select an option
```

To stop selection process, press and hold the **< space >** key.

2) Select **Boot option maintenance menu** using the up and down-arrows, which advances you to the **EFI Boot Maintenance Manager** Main Menu, which is similar to:

```
EFI Boot Maintenance Manager ver 1.10 [14.60]

Main Menu. Select an Operation
```

```
    Boot from a File
    Add a Boot Option
    Delete Boot Option(s)
    Change Boot Order

    Manage BootNext setting
    Set Auto Boot TimeOut

    Select Active Console Output Devices
    Select Active Console Input Devices
    Select Active Standard Error Devices

    Cold Reset
    Exit
```

3) Select **Add a Boot Option.**

```
EFI Boot Maintenance Manager ver 1.10 [14.60]

Add a Boot Option.  Select a Volume

    Removable Media Boot[Acpi(HWP0002,0)/Pci(2|0)/Ata(Primary,..
    Load File [EFI Shell [Built-in]]
    Load File [Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E1E4ED4)]
    Load File [Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E1E4ED4)]
    Exit
```

4) Select the appropriate network interface so that this network boot loads the appropriate file. For example, look for entries with a **Media Access Control (MAC) address** or **link-level address (LLA)** followed by the **MAC/LLA** address of the LAN card.

```
Device Path Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E4A134B)
Boot0001: Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E4A134B)
Edit Existing Boot Option or make a new entry [E-Edit N-New]:
```

5) Enter **N** to add the new boot option.

6) Enter a brief, descriptive boot option name at the message prompt. In this example, the new boot option is named **LAN1**.

```
Enter New Description:
```

7) Enter a brief description for this boot option.

```
New BootOption Data. ASCII/Unicode strings only, with max of
   240 characters
Enter BootOption Data Type [A-Ascii U-Unicode N-No BootOption]:
```

8) Enter the data type of this boot option.

```
Save changes to NVRAM [Y-Yes N-No]:
```

9) Enter **Y** to save the new boot option.

```
LAN1 [Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E1E4ED4)]
Removable Media Boot[Acpi(HWP0002,0)/Pci(2|0)/Ata(Primary,...
Load File [EFI Shell [Built-in]]
Load File [Acpi(HWP0002,0)/Pci(3|0)/Mac(00306E1E4ED4)]
Load File [Acpi(HWP0002,100)/Pci(2|0)/Mac(00306E1E3ED6)]
```

```
Exit
```

10) Exit to the EFI Boot Manager menu taking care not to select a boot option, as you will be forced to re-enter the information for the selected option.

    The new boot option should appear in the **EFI Boot Manager** main menu.

```
EFI Boot Manager ver 1.10 [14.60]

Please select a boot option

   HP-UX Primary Boot: 0/2/2/0.0.0.0
   LAN1
   EFI Shell [Built-in]
   Boot option maintenance menu
   Security/Password Menu

Use ^ and v to change option(s). Use Enter to select an option
```

11) Select the new boot option, **LAN1** in this case, from the list, and then press **Enter**.

    The following is an example of a successful network boot using the new **LAN1** boot option, which ignites the client:

```
Loading.: LAN1
Running LoadFile()

CLIENT IP: 10.1.52.128  MASK: 255.255.248.  DHCP IP: 10.1.53.37
GATEWAY IP: 10.1.48.1
Running LoadFile()

Starting: LAN1

@(#) HP-UX IA64 Network Bootstrap Program Revision 1.0
Downloading HPUX bootloader
Starting HPUX bootloader
Downloading file fpswa.efi  (371200 bytes)

(c) Copyright 1990-2001, Hewlett Packard Company.
All rights reserved

HP-UX Boot Loader for IA64  Revision 1.671

Booting from Lan
Downloading file AUTO   (528 bytes)
Press Any Key to interrupt Autoboot
AUTO ==> boot IINSTALL
Seconds left till autoboot -   0
AUTOBOOTING...
```

Boot Remotely Using bootsys

If a client you need to reinstall is up and running HP-UX 11.X and is available on the network, then you may use the **bootsys** command from the server to cause the client to boot Ignite-UX. Using the **bootsys** command has the advantages of:

- Being able to be done remotely from the Ignite-UX server; that is, it doesn't require access to the client console.

- Being able to boot S800s that are not capable of a network boot.
- Allowing the booting of clients that are on different subnets (since it is not really performing a true network boot).
- The **bootsys** command may also be used to schedule installations to occur at a later time by being called from **at** or **cron**.

The **bootsys** command has options to initiate an automated installation (**-a**), or an installation controlled from the **ignite** user interface (**-w**). **For examples using bootsys**, see *bootsys*(1M).

Boot from Customer-Created Install Media

An install image used to boot a client may be created using **make_medialif**. When transferred to physical media, it may be used to:

- Automatically load an archive image also stored on the media without intervention.
- Initiate an install from an Ignite-UX server. The install may be either automated or interactive.

This is a way for shipping user-customized install media to remote sites. See *make_medialif* (1M) for more information.

Start the Installation from the Server

After the client is booted, its icon should appear on the **ignite** interface. If the server set up is not complete, or if the client could not obtain enough networking parameters via DHCP, the client may require interaction on the client console.

After the client icon appears on the server screen, select it by clicking on the icon for that client. Use the Actions menu to select a task for the selected client. The first task would be to choose "Install Client". Then choose "New Install".

During the installation, choose a configuration file for this installation. Clients will be installed per the description given in the configuration file.
If you want to reuse this configuration, save the file.

Once installation is proceeding, check the client's status on the server.

When the installation completes, you may print a manifest and either save the client's data in a history directory or remove the client and its data from the server.

Refer to the online "Tutorial and Demo" for more information.

Start the Installation from the Client

The standalone type of installation is invoked by booting the client from the network as described in the previous section, or by booting from the Ignite-UX media. After choosing "Install HP-UX", select "Local interaction at console, installing from network server" on the next screen.

Or, if you are installing from media, select "Media installation, with user interaction at local console".

A basic interface, "guided mode" may be chosen at this point. This will direct you though the required client set up steps. This mode is for the novice user, and proceeds though a more limited set of configuration steps and options while giving recommended choices based on the client's hardware.

Additionally, a more sophisticated interface may be selected if the user needs to modify the file system configuration or would like to set the client host name and networking parameters prior to completing the installation.

If you find that you have selected the wrong interface to accomplish your task, you may use the "Cancel" button. This will allow you to switch to the other interface mode.

The standalone installation uses an ASCII (TUI) interface, and requires a keyboard for navigation. For help navigating the TUI, see the *Ignite-UX Administration Guide* sections "Terminal Keyboard Shortcuts" and "Advanced Keyboard Navigation".

Manifest Generation

Included in the Ignite-UX tool set is a command: *print_manifest*(1M). This utility prints a formatted ASCII system manifest to standard output. The manifest includes information on hardware and software installed and configured on the client. It gathers information about the client every time it is run.

ignite may display and/or print the manifest of a newly installed client with the action "View/Print Manifest". If the client's data is moved to history, that data includes both the manifest for the client and the config file. Both these files may be recalled at a later time.

Recovery Archive Creation

Ignite-UX provides the ability to create archives for system recovery and store those archives on the network or on the client being archived. To create a system recovery archive for a particular client running HP-UX:

- Select its icon from the **ignite** display.
- If there is no icon for the client you want to archive, choose "Add New Client for Recovery..." from the Actions menu to create an icon.
- Once the icon is available and selected, choose "Create Network Recovery Archive" from the Actions menu for the client to store the archive on the server. Choose "Create Tape Recovery Archive" from the Actions menu for the client to store the archive on the tape of the client.
- You will be guided through the necessary steps.

The actual command run on the client to create the archive is **make_net_recovery** or **make_tape_recovery**. The command makes calls to **save_config**, **make_sys_image**, **make_arch_config**, and **manage_index** to create a complete recovery solution. No files need to be hand-edited to make the recovery archive usable by Ignite-UX. While the system recovery archive is being created, you may monitor its progress from the Ignite-UX server in the same way that you may monitor the progress of an installation.

In the event that you need to use the archive for recovery of a client:

- Replace or repair the failed hardware.
- Go to the client console and perform a network boot from the server on which the archive resides, or perform a boot from the tape.
- If the client cannot boot over the network or is on a different subnet than the Ignite-UX server, see *make_boot_tape*(1M).
- After the client is booted, select your recovery archive from the list of configuration choices displayed for installation on your client.
- Ignite-UX detects from the configuration file for the archive that this is a recovery situation and uses the archive for recovery rather than for install.

Note that recovery archive creation using the Ignite-UX user interface requires a graphical display. A recovery archive cannot be created by a server running in ASCII mode. Once created, however, the recovery archive may be installed on the client using the ASCII interface on either the client or the server.

Note that although permissible within HP-UX, using characters that do not have a printable graphic that are likely to confuse the terminal on the hardware you commonly use is not recommended. Filenames with these characters cause warning messages to be displayed by **make_tape_recovery** or **make_net_recovery**. In addition, files that contain these non-printable characters are not included in the archive.

International Code Set Support

Ignite-UX uses a variety of system commands to accomplish its functionality. Because the output of many of these commands is parsed, Ignite-UX ensures that the POSIX locale is normally used by modifying environment variables. Help text and some command output not parsed by Ignite-UX will be left in the user's specified locale.

EXTERNAL INPUTS AND INFLUENCES

Default Options

The server maintains a defaults file, **ignite.defs**, located at **/var/opt/ignite/server/ignite.defs**. This file contains a subset of the values that are entered on the Server Configuration screen.

The following values and their defaults are shipped in the Ignite-UX product:

client_timeout: 30
Time (in mins) until the client is declared hung.

halt_when_done: false
Halt the client after installation rather than reboot to invoke **set_parms**.

ignite_welcome: true
Show the server's welcome screen.

itool_welcome: true
Ask for customer information during client installation.

new_client_notification: true
Asks whether the user should be notified when new clients boot.

print_enhanced_manifest: false
Print the manifest using HP-PCL3 control codes for an enhanced output.

full_recovery_info: true
Show all the information screens about recovery archive creation.

Locking

In order to allow multiple install sessions to run concurrently with the currently installing process, **ignite** will lock a client during a New Install or a Repeat Install. This lock is tested in the actions: "New Install", "Repeat Installation", "Stop Client" and "Remove Client".

The lock is removed when the client is stopped or COMPLETE. For stopped clients, it is possible for someone, other than the installer, to remove locks. For COMPLETE clients, it is possible for someone other than the installer to either remove locks or move them to history.

RETURN VALUES

ignite returns the following values:

0 **ignite** completed successfully.

1 **ignite** failed.

DIAGNOSTICS

Logging

All major events are logged to the server log file located at **/var/opt/ignite/logs/server**.

FILES

/opt/ignite/bin
Contains Ignite-UX commands.

/opt/ignite/lbin
Contains Ignite-UX commands used by other commands.

/var/opt/ignite/depots
Contains the software depots used by Ignite-UX.

/var/opt/ignite/logs
Contains log files for each command.

/var/opt/ignite/clients
Contains the per-client directories.

/var/opt/ignite/server
Contains the file ignite.defs (server defaults).

AUTHOR

Ignite-UX was developed by the Hewlett-Packard Company.

SEE ALSO

add_new_client(1M), archive_impact(1M), bootsys(1M), instl_adm(1M), instl_bootd(1M), make_boot_tape(1M), make_bundles(1M), make_config(1M), make_depots(1M), make_medialif(1M), make_net_recovery(1M), make_tape_recovery(1M), manage_index(1M), print_manifest(1M), save_config(1M), setup_server(1M), instl_adm(4), sd(5).