JOIN

User's Guide

431106701A

Please contact RDI Computer at:

RDI Computer Corporation
2300 Faraday Avenue
Carlsbad, CA 92008
Phone (760) 929-0992
FAX (760) 929-9702
email: info@rdi.com
http://www.rdi.com

For RDI Customer Service:
2308 Faraday Avenue
Carlsbad, CA 92008
7:00 AM to 6:00 PM PST
Phone 1-800-734-7030
or (760) 929-0992
FAX (760) 931-5981
email: support@rdi.com

JOIN User's Guide

August 1997

RDI Part # 431106701, Revision A

Printed in the United States of America

JOIN and the JOIN User's Guide are provided to RDI Computer Users by Competitive Automation. Accuracy and updates are controlled by Competitive Automation and any comments, concerns or input to the JOIN software or JOIN User's Guide should be directed to Competitive Automation.

For technical assistance with the JOIN tool, please contact:

Competitive Automation
1050 University Drive
Menlo Park, California 94025
Phone(415) 321-4006
Fax(415) 321-4405 fax
E-Mailinfo@join.com

SPECIAL THANKS TO

Dr. Rob Stevens

Yul Inn

Eric Swildens

Akanna Peck

Notes

1 JOIN User's Guide

Until now, each time a systems administrator wanted to change the configurations of computers on a network, he or she had to spend hours tracking down the computers and entering changes in each machine manually. Entering information as simple as subnet masks, IP addresses and available printers created havoc because each machine was configured individually. As a result, the information entered was delayed and often inconsistent among computers.

Fortunately, with JOIN, administration of networks is much faster and easier.

JOIN allows:

- The automatic assignment of P addresses to clients on networks.
- The assignment of printers and shared file systems to hosts on networks.
- The consistent application of network parameters as subnet masks and default routers to all hosts on a network.
- Complete support of the Dynamic Host Configuration Protocol.
- All bootP parameters.
- Configuration of several computers at one time using templates.
- Automatic configuration of portable computers each time they attach to the network.

JOIN will give the administrator additional hours to perform useful work. Furthermore, because the administrator uses templates that configure several machines at once he or she ensures that configurations are consistent among computers. JOIN saves time and assures administration accuracy.

JOIN MAN PAGES

This manual describes the text of the JOIN man pages. Manual pages are available for the JOIN client configuration files. Man pages provide high-level reference information about JOIN commands. To access the man pages, enter the command man followed by the name of the command you wish to see the documentation on. The man pages can be accessed anywhere you may be in the file system.

For example: % man cd will display the manual page for the cd command.

NAME

joinc - server for client configuration

SYNOPSIS

joinc [-b] [-h] [-dn] [-In] [-exx:xx:xx:xx:xx:xx] [-n]

DESCRIPTION

joinc implements the client half of the DHCP protocol and JOIN extensions. Depending on the particular platform on which it is installed the code may be run from the boot prom, or at some later stage in the boot, after the kernel is already running. This man page describes the JOIN client for SunOS4.x as installed on hardware with local disk. In this mode it may be invoked as a user process, albeit one requiring root privileges. This will most conveniently be done by a command from within rc.local.

Joinc can serve its host client in two ways: it can configure everything monolithically when it is invoked, or it can write a local database which can be interrogated later. The latter is more flexible: it allows third party software access to the data through a published API, and allows system administrators more control over client configuration by customizing rc.local to permit various aspects of the client and its software to be customized in a specific order. Command line flags permit either mode of operation.

OPTIONS

-b Defer client configuration. The interfaces are configured with the proper addresses and subnet masks, but the bulk of the parameters are written to the .cf files housed in the configuration directory (default /etcl-join).

-dn Set debug level to n. If debug is turned on, log messages are also enabled.

-In Enable log messages.

-exx:xx:xx:xx:xx:xx
Pretend to be at Ethernet address xx:xx:xx:xx:xx:xx. This option may not be available on production

releases of joinc

-n Don't configure anything - just exchange packets with the server.

FILES

By default joinc expects to read its policy file and read and write its configuration databases in the directory /etc/join. The environment variable JOINCONFIG may be used to select a different directory.

client.pcy
Parameters governing the behavior of joinc, and general policies concerning network administration.

join.cf
Contains configuration data pertaining to the client host as a whole. Configuration pertaining to a specific interface is stored in files named *interface***.cf** (e.g. **leO.cf**).

leases
Stores a "lease" for each interface which JOIN configured. A lease consists of an IP address, its expiration, the server which granted it, and its commencement.

BUGS

Joinc can configure Clients with two or more interfaces giving each an IP address. However, each interface so configured must be on a different wire.

SEE ALSO

shleases(1), client.pcy(5)

NAME

client.pcy - JOIN client policy

DESCRIPTION

client.pcy is a text database which governs the behavior of JOIN clients, which **joinc** expects to read on startup. If the variable JOINCONFIG is present in joinc's environment, it is taken to be the directory where client.pcy is housed; otherwise /etc/join is searched. Defaults exist for all parameters and switches, so it is not an error if a file does not exist.

FORMAT

Blank lines are ignored. The character '#' introduces a comment which continue to the next newline. Each new policy option must begin and end on a separate line. Policy options are introduced by a keyword, and may be Boolean, or may take a value separated from the keyword by white space (but not a new line). If an option is present more than once, only the value attached to the last occurrence will be take effect; earlier value(s) will be forgotten.

KEYWORDS AND VALUES

arp_timeout *seconds*

When the client receives an IP address from the server it will perform an ARP on the local network to verify than no other client is in fact using the address. If it receives no reply after *seconds* expires it assumes that it may use the address. Default: 2 seconds.

lease_duration *seconds*

The JOIN server grants the client permission to use a an IP address for a fixed period of time (which may be infinite). In the language of DHCP the client is granted a “lease” on the IP address. With this parameter, the client may request a lease of a particular duration, although servers are not bound to honor the request. If

the client doesn't care *seconds* should be set to zero; if an infinite lease is required to minus one, (-1). Otherwise, specify in seconds the lease duration required. Default: 0

retry_broadcast *integer*

This parameter is subtly different from the number of retries a client will make as part of an exponential broadcast retry backoff. Rather it is the number of separate attempts the client will make to contact a server, assuming that replies are received, but that the client, for one reason or another, rejected those replies. Default: 2

timeouts *value, value, value…*

Clients are required by the DHCP protocol to implement an exponential retransmission and backoff when broadcasting discover or request packets. The array of values specifies how long the client should wait for replies before timing out and retrying the broadcast. Default: 4,8,16,32

wait_before_broadcast *seconds*

The DHCP protocol requires clients to delay a random time interval on booting, and after each timeout, before broadcasting to the net. This is to prevent network "flooding" in the event of many clients all trying to configure simultaneously (say after a site-wide power-up). This parameter is the maximum delay that the client will tolerate. The actual delay is randomized from zero up to *seconds*. Note that on each timeout the client will also delay, and that the second and subsequent delays are also random, and need not be the same as the first. Default: 10 seconds.

request *parameter_name*

There may be many instances of the **request** keyword, each with a different *parameter_name*. Each parameter which is configurable through DHCP and the JOIN extensions is identified by a unique parameter. Limited size of DHCP packets dictates that a client should not request data which it cannot use. However, different DHCP servers, or different server policies may dictate that a server return more

configuration than a client requested. For a description of the meaning of the various parameters, consult RFC1542 and others to which it refers. Valid options follow. The first group is DHCP generic:

all_subnets_are_local
arp_cache_timeout
boot_file
boot_file_server
boot_size
broadcast_address
cookie_server
default_ip_time-to-live
dns_domain_name
dns_servers
ethernet_encapsulation
extensions_path
home_directory
host_name
impress_server
interface_mtu
ip_forwarding
keepalive_garbage
lease_time
log_server
lpr_server
mask_supplier
maximum_datagram_reassembly_size
merit_dump_file
name_server
netbios_datagram_distribution_server
netbios_name_server
netbios_node_type
netbios_scope
nis_domain_name
nis_server
non-local_source_routing
ntp_server
path_mtu_aging_timeout
path_mtu_plateau_table
perform_mask_discovery
perform router_discovery
policy_filter
rebinding_time_value
renewal_time_value
resource_location_server

root_path
router
router_solicitation_address
static_routes
subnet_mask
swap~server
tcp_default_time_to_live
tcp_keepalive_interval
time_offset
time_server_(rfc_868)
trailer_encapsulation
x_window_display_manager
x_windows_font_server

The next group is specific to JOIN:

nfs_mounted_file_systems
printers

SEE ALSO

joinc(1),

DARPA Internet Request For Comments RFC1541, RFC1542.

SOLARIS CLIENT MAN PAGES

The JOIN man pages applicable to the Solaris Client appear on the following pages.

NAME

joinc-daemon for client configuration

SYNOPSIS

joinc [-f] [-dn [-l*n*]

DESCRIPTION

joinc implements the client half of the of the DHCP protocol and JOIN extensions for hardware endowed with a local disk.

The DHCP protocol, among other things, permits a client to establish an end point for communication with a network by delivering an IP address for each of the client's network interfaces, and a “lease” on that address. The lease specifies the interval that the address remains valid: it may be infinite or of fixed duration. If it appears that the client wishes to continue using the IP address after its expiration the DHCP protocol must negotiate an extension. For this reason the DHCP client code must run as a daemon, only terminating when the client powers down.

Communication with **joinc daemon** is effected through the agency of an auxiliary program **dhcpconf**(q.v) in much the same way that the init daemon is controlled by telinit. Joinc may be invoked as a user process (albeit one requiring root privileges), but this is not necessary as **dhcpconf** will start it implicitly.

When started, joinc reads its startup file, **cclient.pcy** and either proceeds to act on the instruction(s) passed to it by dhcpconf or to enter a passive state while awaiting a new command. When a command is received to configure an interface the DHCP protocol is started. If successful the interface is configured and brought up. The configuration received is stored in a file named *interface***.dhc** located (by default) under the

directory /etc/join. The client daemon will then sleep until it needs to renew the lease, which will happen well before the lease expires. Upon wakeup, if the interface is found to be down or has a different IP address, joinc considers that the interface is no longer under its control and will drop it from future consideration, until an explicit request arrives from dhcpconf. If the lease cannot be renewed joinc will take the interface down when it expires as required by the DHCP protocol. The user should consult RFC154l for details.

The DHCP protocol also acts as a mechanism to configure other information needed by the client (e.g. name domain, addresses of routers, etc). Joinc does not configure this information but instead acts as a database which may be interrogated by other programs, and in particular by dhcpconf. This approach is more flexible: it allows third party software access to the data through a published API, and allows system administrators more control over client and its software to be customized in a specific order. On clients with a single interface this is quite straightforward. Clients with multiple interfaces may present difficulties, as there exists the possibility that some information arriving on different interfaces may need to be merged, or indeed that it may be inconsistent. Furthermore, the configuration of the interfaces is asynchronous, so requests may arrive while some or all of the interfaces are still unconfigured. Joinc resolves these problems as follows. When a request for a global parameter arrives, it searches all interfaces that it has successfully configured, and returns to the requester the name of the first one it finds which contains the pertinent data. The client program may then access the data by an API which reads the appropriate interface.dhc file. If no interfaces have successfully configured when the request is received, or if none of those which are configured have the data, the request fails. Dhcpconf allows this behavior to be overridden by insisting that the global data sought be associated with a particular interface. See the man page for details.

Joinc writes informational and error messages in four categories: errors, wamings, informatory and debug. Errors are severe, usually unrecoverable, events due to resource exhaustion and other unexpected failure of system call. An error is also generated if the client's lease on an IP address is in imminent danger of expiring. Warnings are less severe, and in most cases describe unusual or incorrect datagrams received from clients, or requests for service than cannot be provided. Informatory messages simply provide a human readable transcription of (correct) actions performed by the server on behalf of client hosts. Debug messages, if joinc was built to generate them, may be generated at various levels of verbosity from zero (not at all) through nine, as controlled by the -d option. They are chiefly of benefit to persons having access to source code.

The disposition of messages is (by default) as follows: warning, informatory and debug messages are discarded: errors are written to /dev/console and are sent to the system logger **syslog**(3) at priority LOG_ERR and with a facility identifier LOG_DAEMON. If warnings have been enabled they also are written to the system console and syslog with the same facility, but at priority LOG_WARNING. The creation and disposition of messages is controlled by the -f;-d and -I command line flags (q.v.), and the environment variable JOINLOG. When present, JOINLOG should name a file to which messages are sent in preference to the system console. Note that until the root file system is mounted read-write no ordinary file can be used for this purpose.

OPTIONS

-d Set debug level to n. If debug is turned on, log messages are also enabled.

-f Run in the foreground instead of as a daemon process.

-in Enable waming (n>0) and log (n>l) messages. If n is not explicitly given, the value one is assumed (i.e. wamings are turned on).

Receipt of SIGUSRi signals joine to dump the contents of its scheduling table and the status of each of the interfaces under its control.

FILES

By default joine expects to read its policy file and read and write its configuration databases in the directorylete/join. The environment variable JOINCONFIG may be used to select a different directory.

client.pcy

Parameters governing the behavior of joine, and general policies concerning network administration.

interface.**dhc**

Contains the configuration for inteiface. The mere existence of this file does not imply that the configuration is correct, since the lease may have expired.

BUGS

Joinc can configure clients with two or more interfaces giving each and IP address. However, each interface so configured must be on a different wire.

SEE ALSO

dbcpconf(l), shleases(l), client.pcy(5), RFCC1541

NAME

dhcpconf - daemon for client configuration

SYNOPSIS

dhcpconf [-d] [-fi [-s] [-a *server_ip***] [-w** *seconds***]**
interface **start|drop|release**

dhcpconf [*interface***]**
dns|domain|gateways|hostname|nis|routes

DESCRIPTION

dhcpconf and its companion joine implement the client side of the Dynamic Host Configuration Protocol, DHCP. The responsibilities of dhcponf are twofold: to control invocation and termination of DHCP on the client's hardware interface(s), and to provide a mechanism for rendezvous with the transactions of DHCP which are proceeding asynchronously with respect to the client boot.

All invocations of **dhcpconf** send instructions or requests to joine which is listening at a well known port number on the Internet Protocol loopback address. Unless the -w flag is given, dhcpconf will wait for the requested operation either to complete, fail, or for the number of seconds specified in the following argument. When the timer expires, dhcpconf exits with a failure code, but the operation requested will continue.

The commands to **dhcpconf** are divided into two groups: start, release, and drop initiate the terminate DHCP control of an interface. The remainder request dhcpconf to configure the host-wide parameters or service specified, according to DHCP supplied data. The latter don't in general need an interface to be specified, except in the circumstance that different interfaces receive different configurations (See NOTES).

OPTIONS

-a *server_ip*
Direct all DHCP protocol messages to the given IP address. Currently not implemented.

-d Start DHCP only if the interface is down.

-w *seconds*

Instructs **dhcpconf** to wait for the time specified (if positive) or forever (if negative), or until the operation completes or fails. This option is only relevant on operations which cannot complete immediately. If the timer expires while the operation is still in progress, dhcpconf will exit with a failure code, but the operation will still continue. If the user specifies a finite wait interval it should, for consistency, be at least equal to the sum of the timeout values for exponential backoff in the startup file client.pcy.

-f This option is only relevant on the **start** command. When an interface is started joinc will send DHCP discover packets using the exponential backoff and retransmission intervals given in client.pcy. If no reply has been received at the end of this cycle, the client will reply to the controller with failure. When this option is in effect, joinc will continue trying to contact a DHCP server forever, either by retrying the whole backoff cycle or using the last timeout value in the array. See client.pcy for details.

start

Puts the interface specified under control of DHCP. Joinc commences the DHCP on the interface. Fine tuning of this process is provided by parameters in the startup file client.pcy.

release

Makes joinc take the interface down and transmit a DHCP release message to the DHCP server that the IP address assigned to the interface is no longer needed. The server will be permitted to re-assign the IP address to another client.

drop

Tells the client daemon that it should relinquish control of the interface. The options to drop and release the interface are subtly different. Release is part of the DHCP protocol; drop is not. Drop tells DHCP that it services for the interface in question are no longer required DHCP will not try to renew the lease on the IP address and if the lease should expire no action will be taken. This violates the protocol and is not recom-

mended, except for testing.

NOTES

When two or more interfaces are configured by DHCP, the possibility exists that the configurations received may differ. This will be the norm for interface specific parameters, but for parameters that pertain to the host as a whole questions of interpretation arise. In particular list items may differ such as the default gateways. When configuring services, dbcpconf will not merge data from different interfaces: rather only a single interface is consulted, which, unless given on the command line, will be the first one in dhcpconf 5 internal array which it has configured at the time the request is made.

DIAGNOSTICS

Exit codes are as follows:

0 Success.
1 DHCP was successful, but the parameters were not found.
2 DHCP was not successful. The DHCP client daemon may not be running, the interface might have failed to configure, or no satisfactory DHCP responses were received.
3 Bad arguments.
4 A timer was set (with a -w) and the interface had not been configured before it expired.
5 Can only be run as root.
6 Some system error (should never occur).

SEE ALSO

dhcpparm(1), joinc(l), sbowdhc(1), shleases(1), client.pcy(5)

NAME

shleases - Display a client's IP address leases.

SYNOPSIS

shleases *filenames...*

DESCRIPTION

shleases reads the file housing information about a client's leases (default /etc/join/leases) and displays its contents on stdout. There will be as many leases as there are interfaces which JOIN has configured. A lease is identified by the name of the interface (e.g., **1eO**) and the IP address assigned to it. It is characterized by a start time, an expiration, and the IP address of the server which granted it.

FILES

/etc/join/*.dhc

SEE ALSO

joinc(1)

NAME

showdhc - Display the contents of configuration files

SYNOPSIS

showdhc *filenames…*

DESCRIPTION

showdhc reads the files specified on the command line and, assuming they are configuration files, displays the data contained in them. The format of the output is a single line in the format of **dhcpcap**(5).

If any of the files named on the command line is not of the correct type, diagnostic output is sent to stderr. Configuration files are identified by an internal magic number, and the major and minor release numbers of JOIN. Configuration files are usually written to directory /etc/join and are named *interface***.dbc**.

FILES

/etc/join/*interface*.dhc

SEE ALSO

joinc(l)

Notes