VSI OpenVMS

TCP/IP Administrator's Guide: Volume I

Document Number: DO-DVTIA1-01B

Publication Date: January 2020

This document describes how to administrate VSI TCP/IP for OpenVMS.

Revision Update Information: This guide supercedes the VSI TCP/IP Administrator's Guide: Volume I, Version 10.5.

Operating System and Version: VSI OpenVMS Version 8.4-2L1 or higher

Software Version: VSI TCP/IP for OpenVMS Version 10.6

VMS Software, Inc., (VSI)
Bolton, Massachusetts, USA

TCP/IP Administrator's Guide: Volume I:

Legal Notice

The VSI OpenVMS documentation set is available on CD.

Preface

vms Software

1. About VSI

VMS Software, Inc. (VSI) is an independent software company licensed by Hewlett Packard Enterprise to develop and support the OpenVMS operating system.

VSI seeks to continue the legendary development prowess and customer-first priorities that are so closely associated with the OpenVMS operating system and its original author, Digital Equipment Corporation.

2. Intended Audience

This manual is intended for anyone who will be administering VSI TCP/IP. It provides an overview of VSI TCP/IP Version 10.5 and contains information about:

- Host tables and the Internet Domain Name System (DNS)
- Establishing IP connectivity and configuring services
- Network Time Protocol (NTP)
- Configuring electronic mail
- Printer configuration

The appendices in this document contain server configuration parameters and commands to configure network rooting.

3. Typographical Conventions

The following conventions are used in this manual:

Convention	Meaning
Ctrl/*X*	A sequence such as **Ctrl/*X*** indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
PF1 *X*	A sequence such as **PF1 *X*** indicates that you must first press and release the key labeled PF1 and then press and release another key (***X***) or a pointing device button.
Enter	In examples, a key name in bold indicates that you press that key.
...	A horizontal ellipsis in examples indicates one of the following possibilities:- Additional optional arguments in a statement have been omitted.- The preceding item or items can be repeated one or more times.- Additional parameters, values, or other information can be entered.
. . .	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.

Convention	Meaning
( )	In command format descriptions, parentheses indicate that you must enclose choices in parentheses if you specify more than one. In installation or upgrade examples, parentheses indicate the possible answers to a prompt, such as: `Is this correct? (Y/N) [Y]`
[ ]	In command format descriptions, brackets indicate optional choices. You can choose one or more items or no items. Do not type the brackets on the command line. However, you must include the brackets in the syntax for directory specifications and for a substring specification in an assignment statement. In installation or upgrade examples, brackets indicate the default answer to a prompt if you press **Enter** without entering a value, as in: `Is this correct? (Y/N) [Y]`
|	In command format descriptions, vertical bars separate choices within brackets or braces. Within brackets, the choices are optional; within braces, at least one choice is required. Do not type the vertical bars on the command line.
{ }	In command format descriptions, braces indicate required choices; you must choose at least one of the items listed. Do not type the braces on the command line.
bold type	Bold type represents the name of an argument, an attribute, or a reason. In command and script examples, bold indicates user input. Bold type also represents the introduction of a new term.
italic type	Italic type indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER=*name*), and in command parameters in text (where *dd* represents the predefined code for the device type).
UPPERCASE TYPE	Uppercase type indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
`Example`	This typeface indicates code examples, command examples, and interactive screen displays. In text, this type also identifies website addresses, UNIX command and pathnames, PC-based commands and folders, and certain elements of the C programming language.
-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes-binary, octal, or hexadecimal-are explicitly indicated.

4. VSI TCP/IP Support

VSI supports VSI TCP/IP running on VSI OpenVMS Integrity Version 8.4-2L1 (or higher) only. Please contact your support channel for help with this product.

5. VSI Encourages Your Comments

You may send comments or suggestions regarding this manual or any VSI document by sending electronic mail to the following Internet address: `<docinfo@vmssoftware.com>`. Users who have OpenVMS support contracts through VSI can contact support@vmssoftware.com [mailto:support@vmssoftware.com] for help with this product. Users who have OpenVMS support contracts through HPE should contact their HPE Support channel for assistance.

6. How to Order Additional Documentation

For information about how to order additional documentation, email the VSI OpenVMS information account: <info@vmssoftware.com>. We will be posting links to documentation on our corporate website soon.

Chapter 1. Host Tables and DNS

This chapter provides an overview of VSI TCP/IP host tables and the Internet Domain Name System (DNS). Both host tables and DNS provide a way of associating host IP addresses with host names.

DNS load balancing helps to provide uninterrupted services if an individual server crashes or cannot handle the number of users trying to access it simultaneously.

1.1. Methods of Associating IP Addresses and Host Names

The two methods of associating IP addresses and host names are:

- Host tables, which offer a simplified method of translating a host name into an Internet address, can become unmanageable if there are many hosts on your network.
- DNS, which is more complicated to configure, offers the advantages of a distributed database.
- Multicast name resolution, which requires little configuration and is designed for small networks operating on a single logical LAN.

If you are connected to the Internet, DNS gives you access to the Internet DNS.

When you install VSI TCP/IP, you are asked if you want to use host tables or DNS. You can change your decision after installation using the instructions from this chapter.

DNS and the host table service are completely separate entities. If DNS is enabled, VSI TCP/IP only accesses host tables if a DNS query fails. DNS, however, never contacts the host table service, and no data is shared between these two services.

In addition to the administrative commands provided for configuring host tables and DNS, VSI TCP/IP provides C language library routines that can access either host tables or DNS. (See the *VSI TCP/IP Programmer's Reference* for more details.)

Note

The section on host tables in this chapter assumes that you are not connected to the Internet, and the section on DNS assumes that you are connected to the Internet.

1.1.1. Host Table

A host table is a file that describes network protocols, services, and host information accessed by utilities such as TELNET, RLOGIN, and TCPDUMP. To create a host table:

1. Build or retrieve the host table in one of the following ways:
   - Add site-specific information to the `HOSTS.LOCAL` file to describe protocol, service, network, and host characteristics. This method is described in Section 1.1.1.3.
   - If you have a UNIX-style /etc/hosts file, convert it as described in Section 1.1.1.5.
2. Compile the host table file as described in Section 1.1.1.6. The next time VSI TCP/IP starts, it installs the compiled host table as a global section.

3. To use the new, compiled host table, install it as a global section (see Section 1.1.1.7).

Once you install the compiled host table, VSI TCP/IP uses it to convert host names to IP addresses and vice versa. The resolution of host names and addresses is invisible to users.

1.1.1.1. Creating the Host Table Source Files

The first step in creating a host table is to gather the appropriate source files. There are three ways to obtain source files:

- Modify existing source files.
- Convert source files from other systems.
- Write completely new files.

1.1.1.2. Host Table Source Files

The compiled form of the VSI TCP/IP for OpenVMS host table is stored in the binary file `IP $:NETWORK_DATABASE`. The following list describes the source files from which the host table is compiled.

File name	Description
`IP$:HOSTS.LOCAL`	Contains any locally defined hosts, protocols, and services. Add custom host table entries to this file. For details on the source file format, see Section 1.1.1.3.
`IP$:HOSTS.SERVICES`	Contains standard information necessary for the operation of VSI TCP/IP. *Do not modify this file.*

All host table source files contain text in the form described by RFC-952, "DoD Internet Host Table Specification," with extensions provided to allow information not anticipated by the designers of RFC-952. For details, see Section 1.1.1.3.

1.1.1.3. Configuring the Host Table

The VSI TCP/IP host table contains more than just host name and address information. Table 1.1 shows the information stored in the VSI TCP/IP host tables and how to access the information.

The protocol and service definitions in the `HOSTS.LOCAL` file supplied with VSI TCP/IP are usually adequate. However, if you have an application that requires additional protocol or service information, you can add definitions for them. See Section 1.1.1.3.1 and Section 1.1.1.3.2 for more information.

You must add information about your network and hosts to the host table. See Section 1.1.1.3.3 and Section 1.1.1.3.4.

Table 1.1. Information Stored in VSI TCP/IP Host Tables

Information	Keys	"C" Access Routine
IP Protocol Types	Name (for example, TCP, UDP) IP Protocol Number	getprotobyname( ) getprotobynumber( )
Services	Name (for example, TCP/TELNET) Port Number	getservbyname( ) getservbyport( )

Information	Keys	"C" Access Routine
Networks	Name (for example, ABC-NET) IP Network Number	getnetbyname( ) getnetbyaddr( )
Hosts	Name (for example, ABC.COM) IP Host Address	gethostbyname( ) gethostbyaddr( )

1.1.1.3.1. Adding Protocol Definitions

Protocol definitions in the VSI TCP/IP `HOSTS.LOCAL` file describe the protocols that can run on top of IP. A protocol definition contains the reserved word PROTOCOL, a protocol number used in the IP protocol field (for example, 6 for TCP), and the name of the protocol (for example, TCP). Enter each protocol definition as a single line without carriage returns or continuation characters.

The valid numbers and the protocol name values are defined in RFC-1060, "Assigned Numbers." Protocol numbers are 8-bit values ranging from 0 to 255. The protocol name can be up to 40 characters in length, and consists of uppercase letters, digits, and, optionally, a hyphen. Spaces and other special characters are not permitted in the protocol name. The format of a protocol definition is:

```
PROTOCOL : number : name :
```

For example:

```
PROTOCOL : 6 : TCP :
PROTOCOL : 17 : UDP :
```

1.1.1.3.2. Adding Service Definitions

Service definitions in the `HOSTS.LOCAL` file describe the various protocol services that may be invoked, and the protocol and service port to contact for these services. A service definition consists of the reserved word SERVICE, the protocol name (for example, TCP), the port number (for example, 23 for TCP/TELNET), and the service name (for example, TELNET).

When specifying the service name, you may use a comma-separated list of names. The first name in the list is the official name of the service; the other names are aliases. Enter each service definition as a single line without carriage returns or continuation characters. Valid port number and service names are defined in RFC-1060, "Assigned Numbers." The format of a service definition is:

```
SERVICE : protocol : port : names :
```

For example:

```
SERVICE : TCP : 23 : TELNET :
SERVICE : TCP : 25 : SMTP,MAIL :
```

1.1.1.3.3. Adding Network Definitions

Network definitions in the `HOSTS.LOCAL` file correlate network names to network numbers. A network definition consists of the reserved word "NET", the network number, and the network name. Network numbers are host IP addresses with the host part of the address set to zero. Enter each network definition as a single line without carriage returns or continuation characters. The format of the network definition is:

```
NET : network-number : name :
```

For example:

```
NET : 0.0.0.0 : DEFAULT-GATEWAY:
NET : 192.16.100.0 : LOOPBACK-NET:
NET : 192.41.228.0 : ABC-NET:
```

1.1.1.3.4. Adding Host Definitions

Host definitions in the `HOSTS.LOCAL` file map IP addresses to host names. A host definition consists of the following:

- The reserved word HOST.
- A comma-separated list of IP addresses by which the host may be contacted.
- A comma-separated list of host names. The first host name is the official host name; any other names are aliases.
- The computer CPU type (used only as an informative message for users).
- The operating system (used only as an informative message for users).
- A comma-separated list of services provided by the host (currently ignored by all programs).

Enter each host definition as a single line without carriage returns or continuation characters. The format of a host definition is:

```
HOST : addresses : host names : CPU type : operating system : services :
```

For example:

```
HOST : 192.0.0.9 : FLOWERS.COM,IRIS : HPE Integrity rx2800 i4 :
VMS : TCP/TELNET,
TCP/FTP,TCP/SMTP :
```

Do not embed spaces in the CPU type, operating system, and offered-services fields. For example, "IBM-PC" is a valid CPU type and "IBM PC" is not a valid CPU type. Spaces are permitted before or after the colon separator character, but not within the field value. You may also leave these fields blank, as in the following entry:

```
HOST : 192.116.0.1 : LOCALHOST : : : :
```

1.1.1.4. Host Name Conformance

Host names (A records) are now restricted to the following characters only:

Table 1.2. Host Name Conformance

Restricted	Excluded
A through Z (uppercase letters A through Z)	_ (underscore)
a through z (lowercase letters a through z)	/ (slash)
0 through 9 (the numbers zero through nine)	
. (dot or period)	
- (hyphen or dash)	

If there are any records in a zone file that do not meet these new guidelines, attempts for name resolution in that zone will fail. Other zones may begin to fail resolving your host names if your zone files are not in compliance with the relevant RFCs. RFC-952 (DoD Internet Host Table Specification, October 1985) and RFC-1123 (Requirements for Internet Hosts – Application and Support, October 1989) contain full descriptions of permitted host name specifications.

As a security measure, the current release of the BIND name server (9.7.2-p3) enforces RFC-952 host name conformance (as modified by RFC-1123). As a result of this change, those host names that do not conform to the new rules will be unreachable from sites running the new name server. VSI TCP/IP properly responds to IGMP request messages because it is compliant with RFC 1112 “Host Extensions for IP Multicasting.”

1.1.1.5. Converting a UNIX /etc/hosts File

VSI TCP/IP provides a utility for converting a UNIX-format /etc/hosts host table file into a `HOSTS.TXT` file. Use the command procedure **IP$:CONVERT_UNIX_HOST_TABLE.COM** to make the conversion. This command has the following syntax:

```
$ @IP$:CONVERT_UNIX_HOST_TABLE infile outfile
```

- *`infile`* defaults to HOSTS.
- *`outfile`* defaults to the `HOSTS.TXT` file.

The outfile is converted to a file compliant with RFC-952, the DoD Internet Host Table Specification used by VSI TCP/IP. If outfile already exists, the command procedure replaces it with a new host table source file. During the conversion:

- Any comments in the input file that begin in the first column are preserved in the output file.
- Extra spaces are compressed.
- Blank lines are removed.
- Characters are converted to uppercase.
- If the host address or name cannot be found, an error is displayed and the output file is deleted.
- Host definitions are created with the host address, official name, and any aliases.

When the conversion completes, you may want to add the CPU type and operating system to each host entry, as described in Section 1.1.1.3.4.

1.1.1.6. Compiling the Host Table

After generating, modifying, or retrieving a VSI TCP/IP host table, compile it into binary form with the following command:

```
$ IP HOST_TABLE COMPILE
```

The command shown, the simplest form of the VSI TCP/IP host table compilation command, should suffice for most compilations. Additional qualifiers to this command are described in the *VSI TCP/IP Administrator's Reference*. The command qualifiers are:

Qualifier	Purpose	Default
/HOST_TABLE_FILE	Binary output file name	`IP$:NETWORK_DATABASE`

Qualifier	Purpose	Default
/SILENTLY	Suppress compilation messages	NOSILENTLY
/STARTING_HASH_VALUE	Initial hash size	Best value[1]
/TBLUK_FILE	Host-completion database	IP$:HOSTTBLUK.DAT
/UNIX_HOST_FILE	Produce UNIX-style hosts file	NOUNIX_HOST_FILE

[1]The "best value" default is computed from the size of the data as the utility attempts to create 512-byte units. When you run **HOST_TABLE COMPILE**, the hash value is listed in the displayed messages. If you only added hosts and want to select a number for this qualifier, use the value from the previous compilation as a starting point.

Note

If you are running VSI TCP/IP on an OpenVMScluster, you only need to run the **IP COMPILE** command on one node of the cluster.

1.1.1.7. Installing a Compiled Host Table as a Global Section

When VSI TCP/IP starts, it installs the compiled host table IP$:NETWORK_DATABASE as a global section. The compiled host table is organized as a "perfect hash" lookup system, allowing VSI TCP/IP to answer any query in one lookup. Because the host table is installed as an OpenVMS global section, access to host table information is extremely fast. To install the compiled host table as a global section without restarting VSI TCP/IP:

1. After recompiling a host table, reinstall it by rebooting or invoking **@IP$:INSTALL_DATABASES**.
2. If you want the new host table to be noticed by the servers that run as part of the IP$SERVER process, restart that process with **@IP$:START_SERVER**.
3. If you want the new host table to be noticed by the SMTP symbiont(s), restart them with the command **@IP$:START_SMTP**. For more information about configuring SMTP queues, see the *VSI TCP/IP Administrator's Guide: Volume II*.

Note

You must run the **@IP$:INSTALL_DATABASES**, **@IP$:START_SERVER**, and **@IP$:START_SMTP** commands on every OpenVMScluster node running VSI TCP/IP.

1.1.2. Using the Domain Name System (DNS)

DNS (Domain Name System) is the preferred method of maintaining host name and address information. DNS provides a fully distributed database of host names, Internet addresses, host information, and mail forwarding information.

When using DNS, a host has access to the full database, yet local information can be maintained locally and exported to the rest of the Domain Name System. DNS is fully documented in several RFCs published by the Defense Data Network Network Information Center (DDN NIC). The following RFCs describe DNS:

RFC-1032	Domain Administrators Operations Guide
RFC-1033	Domain Administrators Guide
RFC-1034	Domain Names — Concepts and Facilities

RFC-1035	Domain Names — Implementation and Specification

The Defense Data Network Information Center (DDN NIC) publishes a softbound manual containing all RFCs pertaining to DNS. This chapter cannot describe all of the intricacies of the Domain Name System.

1.1.2.1. DNS Resolvers and Servers

VSI TCP/IP provides both DNS server and resolver (client) support. The server communicates with the rest of the Internet DNS and participates in distributing DNS data. The resolver receives requests from applications and queries a DNS server for the information.

The DNS Resolver (Client)

The DNS resolver (client) is used by applications to access the DNS database. The DNS resolver is accessed via the `IP$SOCKET_LIBRARY`.

The VSI TCP/IP DNS resolver is enabled when you use the NET-CONFIG **SET DOMAIN-NAMESERVERS** command. This command defines the domain name servers that your system queries to satisfy host-name-to-address translation, and is normally set to the local host. VSI TCP/IP is initially configured to provide a caching-only name server, which works with the resolver. When the resolver fetches a mapping, it is "cached" by the name server and stored for other applications that need the information.

The resolver is almost always invisible to an applications programmer. When using the VSI TCP/IP socket library, the normal host table access routines, such as **gethostbyname( )** and **gethostbyaddr( )**, automatically call the DNS resolver routines and only use host table access when a DNS resolver fails. For the VSI TCP/IP OpenVMS ULTRIX Connection (UCX) $QIO Driver, the $QIO functions for translating host names and addresses are referred to as the VSI TCP/IP IP$SERVER process, which then uses the DNS resolver routines to satisfy the UCX name translation query.

The VSI TCP/IP SMTP symbiont queries the Internet DNS for mail forwarding information and calls the DNS resolver routines directly to query the DNS for Mail Exchanger (MX) resource records.

The DNS Server

The VSI TCP/IP DNS server is based on the ISC BIND 9.7.2-p3 name server. The IP$SERVER process starts the IP$NAME_SERVER process if this process is not running. The DNS server processes queries from resolvers, then responds to the queries, or queries other DNS servers to obtain information from other parts of the DNS database.

VSI TCP/IP is shipped with DNS configured to operate as a caching-only server in the Internet environment.

The VSI TCP/IP DNS server uses the `IP$:NAMED.CONF` file for configuration information. The configuration file is the equivalent of the `/etc/named.conf` file used by UNIX-based BIND implementations.

The configuration file typically contains references to other files that contain definitions for the server's contribution to the DNS database. These other files contain text in standard "resource record" format, as described in RFC-1035, "Domain Names — Implementation and Specification."

The ISC BIND 9.7.2-p3 distributed HTML documents are on the CD-ROM in the directory named [BIND9-DOC]. Refer to the file named `INDEX.HTML` for a list of BIND documents.

1.1.2.2. Remote Name Daemon Control (RNDC)

Remote Name Daemon Control (RNDC) remotely controls the operation of a name server. It communicates with the NAMED server over a TCP connection authenticated with digital signatures. In the current versions of RNDC and NAMED, the only supported authentication algorithm is HMAC-MD5, which uses a shared secret on each end of the connection. This provides TSIG-style authentication for the command request and the name server's response. All commands sent over the channel must be signed by a key_id known to the server.

RNDC reads a configuration file to determine how to contact the name server and decide what algorithm and key it should use. The DNS/NAMED server also provides for an RNDC configuration in named.conf. These 2 files will work in tandem to allow remote control of the NAMED server.

Note

RNDC is disabled by default.

Configuring RNDC

To configure RNDC, you must run the VSI provided RNDC_CONFGEN Utility. This utility will generate configuration information for both the named.conf and rndc.conf files.

Note

A system manager should generate a different key for every server that they want to remotely manage.

Example 1.1 shows the example of RNDC_CONFGEN session.

Example 1.1. RNDC_CONFGEN Session Example

```
$ @sys$manager:ip$commands
$ rndc_confgen
# Start of rndc.conf
key "rndc-key" {
 algorithm hmac-md5;
 secret "xQXMQrK4bLonYatzs3hLcg==";
};

options {
 default-key "rndc-key";
 default-server 127.0.0.1;
 default-port 953;
};
# End of rndc.conf

# Use with the following in named.conf, adjusting the allow list as needed:
# key "rndc-key" {
#  algorithm hmac-md5;
#  secret "xQXMQrK4bLonYatzs3hLcg==";
# };
#
# controls {
#  inet 127.0.0.1 port 953
#   allow { 127.0.0.1; } keys { "rndc-key"; };
# };
```

```
# End of named.conf
```

First, copy the rndc.conf section from the previous example output to the rndc.conf file as shown in the following example.

```
$ edit ip$config:rndc.conf
# Start of rndc.conf
key "rndc-key" {
        algorithm hmac-md5;
        secret "xQXMQrK4bLonYatzs3hLcg==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf
```

The VSI provided named.conf file has a standard configuration for RNDC that is already included in the file.

To configure RNDC, edit the named.conf file, and, immediately after the already included section, add the lines that the rndc_confgen utility provided.

The following example shows the correctly edited named.conf file.

```
/* The rndc-key can be kept in a separate rndc.conf file */
/* rndc provides remote nameserver control in Bind 9 */
/* default is disabled */
/key "rndc-key" {
/        algorithm hmac-md5;
/        secret "qUyMmEHlQMKn8g0I9W1yTw==";
/};
/* disable the default rndc control socket */
//controls {
//inet * allow { none; } keys { "rndc-key"; };
//};
/controls {};
key "rndc-key" {
        algorithm hmac-md5;
        secret "xQXMQrK4bLonYatzs3hLcg==";
 };
#
 controls {
        inet 127.0.0.1 port 953
                allow { 127.0.0.1; } keys { "rndc-key"; };
 };
```

Note

The above configuration will allow connections from 127.0.0.1 only. A list of addresses that will be used to control the DNS server may be used if other systems are allowed RNDC access.

Using RNDC

To use RNDC, you need to reload the DNS server.

```
$ ip netcontrol domainname restart
Connected to NETCONTROL server on "LOCALHOST"
< ia50.eng.vmssoftware.com Network Control  at Tue  2-APR-2019 09:15 EDT
< Starting Nameserver
< Nameserver Started, process id 24201C2D
```

Then, test your RNDC configuration by issuing an RNDC command.

```
$ rndc reload
server reload successful
```

To view the RNDC commands available in HELP, issue RNDC at the DCL prompt.

```
$ rndc
Usage: rndc [-b address] [-c config] [-s server] [-p port]
 [-k key-file ] [-y key] [-V] command

command is one of the following:

  addzone zone [class [view]] { zone-options }
    Add zone to given view. Requires new-zone-file option.
  delzone zone [class [view]]
    Removes zone from given view. Requires new-zone-file option.
  dumpdb [-all|-cache|-zones|-adb|-bad|-fail] [view ...]
    Dump cache(s) to the dump file (named_dump.db).
  flush   Flushes all of the server's caches.
  flush [view] Flushes the server's cache for a view.
  flushname name [view]
  Flush the given name from the server's cache(s)
  flushtree name [view]
  Flush all names under the given name from the server's cache(s)
  freeze Suspend updates to all dynamic zones.
  freeze zone [class [view]]
  Suspend updates to a dynamic zone.
  halt  Stop the server without saving pending updates.
  halt -p Stop the server without saving pending updates reporting
  process id.
  loadkeys zone [class [view]]
  Update keys without signing immediately.
  notify zone [class [view]]
  Resend NOTIFY messages for the zone.
  notrace Set debugging level to 0.
  querylog newstate
  Enable / disable query logging.
  reconfig Reload configuration file and new zones only.
  recursing Dump the queries that are currently recursing (named.recursing)
  refresh zone [class [view]]
  Schedule immediate maintenance for a zone.
  reload Reload configuration file and zones.
  reload zone [class [view]]
  Reload a single zone.
  retransfer zone [class [view]]
  Retransfer a single zone without checking serial number.
  secroots [view ...]
  Write security roots to the secroots file.
  sign zone [class [view]]
  Update zone keys, and sign as needed.
  signing -clear all zone [class [view]]
```

```
  Remove the private records for all keys that have
  finished signing the given zone.
  signing -clear <keyid>/<algorithm> zone [class [view]]
  Remove the private record that indicating the given key
  has finished signing the given zone.
  signing -list zone [class [view]]
  List the private records showing the state of DNSSEC
  signing in the given zone.
  signing -nsec3param hash flags iterations salt zone [class [view]]
  Add NSEC3 chain to zone if already signed.
  Prime zone with NSEC3 chain if not yet signed.
  signing -nsec3param none zone [class [view]]
  Remove NSEC3 chains from zone.
  stats  Write server statistics to the statistics file.
  status Display status of the server.
  stop  Save pending updates to master files and stop the server.
  stop -p Save pending updates to master files and stop the server
  reporting process id.
  sync [-clean] Dump changes to all dynamic zones to disk, and optionally
  remove their journal files.
  sync [-clean] zone [class [view]]
  Dump a single zone's changes to disk, and optionally
  remove its journal file.
  thaw  Enable updates to all dynamic zones and reload them.
  thaw zone [class [view]]
  Enable updates to a frozen dynamic zone and reload it.
  trace  Increment debugging level by one.
  trace level Change the debugging level.
  tsig-delete keyname [view]
  Delete a TKEY-negotiated TSIG key.
  tsig-list List all currently active TSIG keys, including both statically
  configured and TKEY-negotiated keys.
  validation newstate [view]
  Enable / disable DNSSEC validation.

Version: 9.9.9-P8
```

1.1.3. Enabling a Caching-Only Name Server

A caching-only name server queries other name servers to resolve host names to IP addresses. The answers received from the inquiry are retained and used in subsequent name resolver requests without querying the remote name server. The default DNS configuration files are shipped as a caching-only server; you only need to enable DNS.

Note

Master or authoritative name servers also cache responses to their queries.

Use a caching-only name server if your OpenVMS system is not the authoritative name server for any domain.

To determine whether DNS is enabled on your system, check the `IP$NAMESERVERS` logical name:

```
$ SHOW LOGICAL IP$NAMESERVERS
```

If DNS is enabled, your system is already set up to use DNS to resolve host names and addresses. If the logical is set to 127.0.0.1, your system also acts as its own name server.

If DNS is not enabled, you can enable it to take effect when the system is rebooted or to take effect immediately.

To enable the caching-only name server to take effect when the system reboots:

```
$ IP CONFIGURE
VSI TCP/IP for OpenVMS Network Configuration Utility 10.5(nnn)
[Reading in MAXIMUM configuration from IP$:IP.EXE]
[Reading in configuration from IP$:NETWORK_DEVICES.CONFIGURATION]
NET-CONFIG>SET DOMAIN-NAMESERVERS 127.0.0.1
NET-CONFIG>SET HOST-NAME fully-qualified-domain-name
```

Also ensure the fully qualified domain name (FQDN) is included in your host table. The easiest way to do this is to modify the HOST line that describes your system in `IP$:HOSTS.LOCAL` to be of this form:

`HOST : address(s) : FQDN,short_name : [CPU] : [OS] : [services] :`

See Section 1.1.1.3.4 for more information. If you change the `IP$:HOSTS.LOCAL` file to include your FQDN, you must also recompile the table and install it as a global section, as shown in this example:

```
$ IP HOST_TABLE COMPILE
$ @IP$:INSTALL_DATABASES
```

To make the change take effect without rebooting:

1. Define the logical name `IP$NAMESERVERS` as 127.0.0.1 to take advantage of your local name server's cache:

   ```
   $ DEFINE/SYSTEM/EXECUTIVE IP$NAMESERVERS "127.0.0.1"
   ```

2. Define the official, fully qualified domain name:

   ```
   $ DEFINE/SYSTEM/EXECUTIVE_MODE IP$HOST_NAME"fqdn"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE ARPANET_HOST_NAME "fqdn"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE UCX$INET_HOST "name"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE TCPIP$INET_HOST "name"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE UCX$INET_DOMAIN "domain"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE TCPIP$INET_DOMAIN "domain"
   ```

3. Define the UCX equivalents:

   ```
   $ DEFINE/SYSTEM/EXECUTIVE_MODE UCX$BIND_DOMAIN "domain-name"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE UCX$BIND_SERVER000 "127.0.0.1"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE TCPIP$BIND_DOMAIN "domain-name"
   $ DEFINE/SYSTEM/EXECUTIVE_MODE TCPIP$BIND_SERVER000 "127.0.0.1"
   ```

4. Restart the IP$SERVER process:

   ```
   $ @IP$:START_SERVER
   ```

5. Restart the SMTP symbiont(s):

   ```
   $ @IP$:START_SMTP
   ```

1.1.3.1. Caching-Only Name Server Configuration with Forwarders

A caching-only name server usually sends queries directly to the name server that contains the answer. A *forwarders* option can be used to redirect these queries to a central name server that accepts

recursive queries from other servers and functions as a second-level cache. The central name server then queries the name server that contains the answer, and caches a copy.

Configure caching-only name servers with forwarders in a network with multiple caching servers to:

- Reduce load on your connection to the Internet.
- Improve DNS response to repeated queries.

Note

Although adding forwarders statements improves DNS response times, DNS does not require it.

If the DNS server configuration file specifies one or more *forwarders*, the server sends all queries for data not in the cache to the *forwarders*.

Use a text editor to add a *forwarders* statement to the *options {};* section of the `NAMED.CONF` file to forward for all zones. Forwarding can also be specified on a per-zone basis, or turned off on a per-zone basis using *forwarders { };*.

For example, to add two servers with these IP addresses 192.1.1.98 and 192.1.1.99, the *forwarders* statement would be:

```
forwarders { 192.1.1.98; 192.1.1.99;};
```

There is also a *forward* option that tells the server how to use the forwarders. The *forward* option is only meaningful if the *forwarders* list is not empty. There are two values to use with the *forward* option.

Value	Description
forward first; (default)	The server queries the *forwarders* first. If the *forwarders* fail to find an answer, the server queries the root servers.
forward only;	The server queries the *forwarders* only. If the *forwarders* fail to find an answer, the server does not query the root domain servers.

For example, to use the *forwarders* only:

```
options {
   forward only;
   forwarders { 192.1.1.98; 192.1.1.99;};
};
```

After adding the *forwarders* and *forward* lines, restart the name server.

```
$ REPLY/ENABLE=NETWORK/TEMPORARY
$ IP NETCONTROL DOMAINNAME RESTART
```

1.1.3.2. Using a Search List to Resolve Host Names

When you specify a simple host name, the DNS resolver expands automatically the simple name by appending the local host's domain to it. For example, if your host is farfel.flowers.com and you enter this command: **$ TELNET kasha** the DNS resolver expands kasha automatically into kasha.flowers.com and attempts to translate that name into an address.

To have a different domain appended to host names, or to search multiple domains, define the IP$SEARCHDOMAINS logical. If you specify multiple domains, separate them with blanks. For example, to have the resolver search for simple names in the domains sub1.flowers.com and sub2.flowers.com instead of the local domain, use the following logical name definition:

```
$ DEFINE IP$SEARCHDOMAINS "sub1.flowers.com sub2.flowers.com"
```

This logical can be defined by individual users or as a system-wide logical (with the /SYSTEM qualifier). The search list can be up to 511 bytes in length. To search the local domain in addition to other domains, include the local domain in the IP$SEARCHDOMAINS list. The maximum number of domains to search is 6.

1.1.3.3. Setting Up a Master Name Server

The following procedure sets up a master, or authoritative, name server. This type of name server gets data for its zone from files on the host where it runs. A zone is the domain, or portion of a domain, for which the master server has complete information. To set up a master server:

1. Determine whether DNS is enabled on your system by checking the logical name IP$NAMESERVERS:

   ```
   $ SHOW LOGICAL IP$NAMESERVERS
   ```

   If IP$NAMESERVERS is defined, your system uses DNS to resolve host names and addresses. If IP$NAMESERVERS is 127.0.0.1, your system also acts as its own DNS server.

2. If it is not already enabled, enable DNS by specifying one or more DNS servers with NET-CONFIG , use the **SET DOMAIN-NAMESERVERS** command. The name server list can include up to three IP addresses.

3. Make sure the Official Host Name configured in **IP CONFIGURE** is the fully qualified domain name of the name server. If not, define it without rebooting using the following command:

   ```
   $ DEFINE/SYSTEM/EXECUTIVE_MODE IP$HOST_NAME fully qualified domain name
   ```

4. To specify the time between DNS name server requests to a nonresponding server and the number of attempts to make to re-establish communications, use the NET-CONFIG **SET NAMESERVER-RETRANSMISSION** command.

5. Update the DNS configuration file, IP$:NAMED.CONF, to add information about your site, as described in Section 1.1.3.5.

6. For each zone of type master, add a zone file to describe the zone characteristics. A zone is a range of authority that includes one or more fully qualified domains or part of a domain. A zone file describes the contents of a zone. Configuring a zone information file is described in Section 1.1.9.

7. Reload the DNS server (see Section 1.1.9.1).

To prevent log files from being truncated when they are closed, use the logical IP $SERVER_DONT_TRUNCATE_LOG. Note that the default allocation quantity for the log files is 80 blocks, so they could have a moderate amount of empty space.

1.1.3.4. Domain Name versus Host Name

Your domain name should not also be the primary name of a host on your network. This is because of a possible conflict between the host name LOCALHOST and a registered Internet domain.

For example, if your domain name is flowers.com, and you want to have a host with the same name, give the host a different primary name, such as main.flowers.com. Create an appropriate A (address) and record and PTR (inverse lookup) entry with the name main.flowers.com. Then, add an A record for flowers.com that also points to the address for main.flowers.com.

If you already have a host with the same name as your domain name and the host is running VSI TCP/IP, configure a local domain for the VSI TCP/IP resolver on that system. For example:

```
$ IP CONFIGURE
NET-CONFIG>SET LOCAL-DOMAIN flowers.com
NET-CONFIG>EXIT
```

1.1.3.5. The IP$:NAMED.CONF File

The main DNS configuration file, from which the name server gets its initial data, is `IP$:NAMED.CONF`. The equivalent of this file in UNIX-based BIND implementations is `/etc/named.conf`. Use this file to add information about your site when setting up a master DNS server. An example configuration file follows.

```
/*
** Sample Configuration File for DNS server
*/
options {
   directory "SYS$SYSROOT:[IP]";
// forward only;
   forwarders { 128.0.1.1; 128.0.2.10; };
};
zone "flowers.com" in {
   type master;
   file "domain-name-service.iris";
};
zone "0.128.in-addr.arpa" in {
   type master;
   file "domain-name-service.iris-net";
};
zone "cc.flowers.com" in {
   type slave;
   masters { 128.0.1.1; };
   file "domain-name-service.cc";
};
zone "1.0.128.in-addr.arpa" in {
   type slave;
   masters { 128.0.1.1; };
   file "domain-name-service.cc-net";
};
   zone "0.0.127.in-addr.arpa" in {
         type master;
         file "domain-name-service.local";
   };
   zone "localhost" in {
         type master;
         file "domain-name-service.localhost";
   };
   zone "." in {
         type hint;
         file "domain-name-service.cache";
   };
```

The following sections describe the zone, options, and logging sections.

1.1.4. Zone

A *zone* is that part of a name server that contains complete information about the domain name space. You specify a *zone* in the following way:

```
zone "domain name" [class] {
          type type;
          [other statements...]
};
```

The following table defines the NAMED.CONF zone fields.

Table 1.3. NAMED.CONF Zone Fields

Field	Description
class	The class to which this zone applies. If the class is not specified, the type IN is used by default. The syntax is [(in | hs | hesiod | chaos)], • in (default) — Used for objects connected to the Internet. This is the only supported type. • hs or hesiod — Confined mostly to MIT. hs is the abbreviation for hesiod. • chaos — An historic network. Not used today.
domain name	The name of the domain for which this zone is authoritative.

Table 1.4. Zone Statements

Statement	Description
file *filename*;	Specifies the name of the zone file. The zone file should have W:RE privileges.
masters [port *ip_port*] {*ip_addr*; [*ip_addr*; ...]};	Specifies the IP address(es) and port from where the server is to transfer the zone data. This statement is meaningful only for slave or stub zones.
type (master | slave | stub | hint | forward);	See Table 1.6 for a description of these zones.

Table 1.5. Optional Zone Statements

Statement	Description
allow-query { *address_match_list*; }; allow-transfer { *address_match_list*; };	Overrides the "allow-query" and the "allow-transfer" statements in the global options section for this zone. See Table 1.7.
allow-update { *address_match_list*; };	Specifies the addresses of hosts that are allowed to modify the zone with dynamic updates. Defaults to none.

Statement	Description
also-notify {*ip_addr*; [*ip_addr*; ...] };	Overrides the “also-notify” statement in the global options section for this zone. See Table 1.7.
check-names (warn | fail | ignore);	Overrides the default name checking specified in the global options section. See the *check-names* statement in Table 1.7 for more details.
forward (only | first); forwarders {[*ip_addr*;...]};	Overrides the “forward” and “forwarders” statements in the global options section for this zone. See Table 1.7.
ixfr-base *path_name*;	Specifies the file name used for the IXFR transaction log file.
notify [(yes | no);]	Specifies if zone change notifications should be sent to the slave servers for the zone. This overrides the *notify* statement in the global options section. See the *notify* statement in Table 1.7 for more details.
pubkey *flags protocol algorithm key*;	Represents a publickey for this zone. The key is needed when this is the top level authoritative zone served by this server and there is no chain of trust to a trusted key. The key is considered secure, so data it signs will be considered secure. The DNSSEC flags (number), protocol (number), and algorithm (number) are specified, as well as a base-64 encoded string representing the key.
transfer-source *ip_addr*;	Overrides the “transfer-source” statement in the global options section for this zone. See Table 1.7.

Table 1.6. Zone Types

Type	Description
forward	A forward zone is use to direct all queries in it to other servers. The specification of the options in such a zone overrides any global options declared in the *options* statement. If a *forwarders* statement is not present in the zone or an empty list for *forwarders* is given, forwarding is not done for the zone, cancelling the effects of any *forwarders* in the *options* statement. Thus, if you want to use this type of zone to change the behavior of the global *forward* option, and not the servers used, you need to re-specify the global forwarders.
hint	The `IP$:NAMED.CONF` file example specifies that data in the `DOMAIN-NAME-SERVICE.CACHE` file, which is in standard resource record format, should be placed in the bootstrap cache. The hint zone definition is used to specify locations of root domain servers. An up-to-date list of root name servers is obtained automatically and stored in memory without replacing the cache file.
master	Specifies data for the zone and the domain. The first master zone definition in the example states that the file `DOMAIN-NAME-SERVICE.IRIS` contains authoritative data for the `FLOWERS.COM` zone, in standard resource record format. The second master zone definition in the example states that the file `DOMAIN-NAME-SERVICE.IRIS-NET` contains authoritative data for the domain

Type	Description
	0.128.IN-ADDR.ARPA, which is used in translating addresses in network 128.0.0.0 to host names. Each zone master file should begin with an SOA (Start Of Authority) resource record for the zone, as shown in Section 1.1.9.
slave	Specifies the zones for which this DNS server acts as a secondary name server. After this name server receives a "zone transfer," it becomes authoritative for the specified zone. The first slave zone definition in the example on page 10-13 specifies that all authoritative data under `CC.FLOWERS.COM` is to be transferred from the name server at 128.0.1.1. The file statement in this section is the file name in which to back up the transferred zone. When it boots, the name server loads the zone from this backup file, if it exists, providing a complete copy even if the master DNS server is unreachable. This file is updated whenever a new copy of the domain is received by automatic zone transfer from one of the master servers. The file statement is optional, but recommended to speed up server startup and eliminates needless bandwidth. The second slave zone definition in the example states that the address-to-hostname mapping for the subnet 128.0.1.0 should be obtained from the same list of master servers as the previous zone.
stub	Works like a slave zone, except it only transfers the name server records for the master zone rather than the full zone information.

1.1.5. Options

The *`options`* statement sets up global options to be used by BIND. Use this statement only once in a configuration file. If it is used more than once, the first occurrence determines what options to use, and a warning is generated. If no *`options`* statement is present, an options block is used setting each option to its default value. You specify *`options`* in the following way:

Table 1.7 shows some of the more commonly used option statements. For a full listing of the option commands that can be specified, please consult a detailed text such as O'Reilly's "DNS and BIND", or else consult the BIND RFC.

```
options {
    options statements
};
```

The following table defines the `NAMED.CONF` options.

Table 1.7. NAMED.CONF Options

Option	Description
allow-query {*`address_match_list`*; };	See Section 1.1.5.1. Specifies the addresses of hosts that are allowed to query the server for information. It defaults to all.

Option	Description
allow-recursion {*address_match_list*; };	See Section 1.1.5.1. Specifies which hosts are allowed to make recursive queries through this server. If not specified, recursive queries from all hosts are allowed (default).
allow-transfer {*address_match_list*; };	See Section 1.1.5.1. Specifies the addresses of hosts that are allowed to perform zone transfers from the server. It defaults to all.
also-notify {*ip_addr*; [*ip_addr*; ...] };	Defines a global list of IP addresses that get sent NOTIFY messages whenever a fresh copy of the zone is loaded. This ensures that copies of the zones converge quickly on "stealth" servers. If an *also-notify* list is given in a *zone* statement, it overrides the *options also-notify* statement. When a *zone notify* statement is set to NO, the IP addresses in the global *also-notify* list are not sent NOTIFY messages for that zone. The default is the empty list (no global notification list).
blackhole {*address_match_list*; };	See Section 1.1.5.1. Specifies a list of addresses the server will not accept queries from or use to resolve a query. Queries from these addresses will not be responded to.

check-names (master | slave | response)
 	
(warn | fail | ignore);
	The server checks names in three areas:
	
	• Master zone files.
	
	• Slave zone files.
	
	• Responds to queries the server has initiated.
	
	The server assumes the following defaults:
	
	`options {`
	`  check-names master fail;`
	`  check-names slave warn;`
	`  check-names response ignore;`
	`};`
	
	ignore — No checking is done.
	
	warn — Names are checked against their expected client contexts. Invalid names are logged, but processing continues normally.
	
	fail — Names are checked against their expected client contexts. Invalid names are logged, and the offending data is rejected.
	
	If *check-names response fail* has been specified, and answering the client's question requires

Option	Description
	sending an invalid name to the client, the server sends a REFUSED response code to the client instead.
directory *path*;	Causes the server to change its default directory to the specified directory. This can be important for the correct processing of `$INCLUDE` files in primary zone files, or file statements in zone definitions.
fake-iquery (yes | no);	BIND 9 does not do IQUERY simulation. Obsolete.
fetch-glue (yes | no);	Obsolete for BIND 9.

forward (only | first);	This statement is meaningful only if there is a non-empty *forwarders* statement.
	
	When *first* (default) is used, the server queries the *forwarders* first before consulting the root domain servers.
	
	When *only* is used, the server queries the *forwarders* only. If the *forwarders* fail to find an answer, the server does not query the root domain servers. For example:
	
	`options {`
	`  forward only;`
	`  forwarders`
	`  { 192.1.1.98; 192.1.1.99;};`
	`};`

forwarders {*ip_addr*; [*ip_addr*; ...]};	Specifies the addresses of site-wide servers that accept recursive queries from other servers. If the DNS server configuration file specifies one or more *forwarders*, the server sends all queries for data not in the cache to the *forwarders*. Central name servers designated to handle forwarded requests can then develop a cache of answers to external queries. The central cache reduces the number of requests sent to root name servers and improves DNS performance.

listen-on [port *ip_port*]
 	
{*address_match_list*; };
 	
also see "listen-on-v6"
	See Section 1.1.5.1.
	
	Specifies what port on what interface to listen on. The default is:
	
	`listen-on port 53 { any };`
	
	For example:
	
	`options {`
	`  // listen on port 53 for`
	`  // external interfaces.`
	`  listen-on { 192.42.95.0;};`
	`  // listen on port 43 for`
	`  // internal interfaces.`
	`  listen-on port 43`

Option	Description
	{ 127.0.0.1; 10.0.0.0; }; };
provide-ixfr (yes | no);	If *yes*, a transaction log is kept for incremental zone transfer of each zone. The default is *yes*.
max-ixfr-log-size *number*;	Obsolete for BIND 9.
max-transfer-time-in *number*	Terminates the inbound zone transfers (*named-xfer* processes) running longer than the minutes specified. The default is 120 minutes (2 hours).
min-roots *number*;	Obsolete for BIND 9.
notify (yes | no);	If *yes* (default), the server notifies slave servers if there are any changes to a domain for which the server is master or slave. The server determines the slave servers by the name server records contained in the zones data file. For more information, see the *also-notify* statement.
query-source [address (ip_addr | *)] [port (ip_port | *)] ; ***see also "query-source-v6"***	If the server does not know the answer to a question, it queries other name servers. *query-source* specifies the address and port used for such queries. If address is * or is omitted, a wildcard IP address (INADDR_ANY) is used. If port is * or is omitted, a random unprivileged port is used. The default is query-source address * port *; **Note** *query-source* currently applies only to UDP queries; TCP queries always use a wildcard IP address and a random unprivileged port.
recursion (yes | no);	If *yes* (default), the server attempts to do all the work required to answer a query that has requested recursion. Turning this off results in the server responding to the client with referrals. To prevent the server's cache from growing, use *recursion no* in combination with *fetch-glue-no*.
rrset-order {*order_spec*; [*order_spec*; ...] }	See Section 1.1.7.
sortlist {*address_match_list*; };	See Section 1.1.7.
topology {*address_match_list*; };	Not implemented for BIND 9
transfer-format *number*	The server supports two zone transfer methods. *one-answer* uses one DNS message per resource record transferred. *many-answers* packs as many resource records as possible into a message. *many-answers* is

Option	Description
	more efficient, but is only known to be understood by BIND 8.1 and higher and patched versions of BIND 4.9.5. The default is *one-answer*. *transfer-format* may be overridden on a per-server basis by using the server statement.
transfer-source *ip_addr*; ***see also "transfer-source-v6"***	Determines which local address will be bound to the TCP connection used to fetch all zones transferred inbound by the server. If not set, the value defaults to a system controlled value, usually the address of the interface "closest to" the remote end. This address must appear in the remote end's *allow-transfer* option for the zone being transferred, if one is specified. This statement sets the *transfer-source* for all zones, but can be overridden on a per-zone basis by including a *transfer-source* statement within the zone block in the configuration file.
transfers-in *number*	The maximum number of inbound zone transfers that can be running concurrently. The default value is 10. Increasing transfers-in may speed up the convergence of slave zones, but it also may increase the load on the local system.
transfers-per-ns *number*	The maximum number of inbound zone transfers (*named-xfer* processes) that can be concurrently transferred from a given remote name server. The default value is 2. Increasing transfers-per-ns may speed up the convergence of slave zones, but it also may increase the load on the remote name server. transfers-per-ns may be overridden on a per-server basis by using the transfers phrase of the server statement.
version *version_string*;	Specifies the version number the server should report via the *ndc* command or via a query of name version.bind in class *chaos*. The default is the real version number of the server.

1.1.5.1. Address_match_list

The following can be address match lists:

- An IP address (in dotted-decimal notation)
- Another address match list
- An IP prefix (in /- notation)
- An address match list defined with the *acl* statement
- A key ID, as defined by the *key* statement

The following Access Control Lists (ACLs) are predefined and are not case-sensitive:

- any
- none
- localhost
- localnets

Place the ! character in front of elements you want to negate.

Remember that address match lists follow the standard named.conf syntax and require a semi-colon (;) after each element. For example:

```
allow-update { !192.168.0.1; 192.168.0.0/16; };
```

When an IP address or prefix is compared to an address match list, the list is examined and the first match (regardless of its negated state) is used. The interpretation of a match depends on the conditions defined in the following table.

When a list is being used...	A non-negated match...	A negated match...
as anaccess control list	allows access.	denies access.
with the *listen-on* option	causes the DNS server to listen on matching interfaces.	causes the DNS server to NOT listen on matching interfaces.
with the `topology` clause	returns a distance based on its position on the list; the closer the match to the start of the list, the shorter the distance between it and the server.	is assigned the maximum distance from the server. **Note** If there is no match, the address gets a distance that is further than any non-negated list element, and closer than any negated element.

Since the address match list uses a first-match algorithm, care must be taken when using negation. In general, if an element is a subset of another element, the subset should be present in the list before the broader element.

For example, 10.0.0/24; !10.0.0.1 will never negate to the 10.0.0.1 address because a 10.0.0.1 address will match with the 10.0.0/24 element and not traverse any farther. So the 10.0.0.1 address will be accepted in the match list.

Using !10.0.0.1; 10.0.0/24 will elicit the desired effect. The 10.0.0.1 will be matched against the first, negated, element. All other 10.0.0.* addresses will pass by the 10.0.0.1 element and be matched against the 10.0.0/24 subnet element.

1.1.6. Logging

The `logging` section configures a wide variety of logging options for the name server. Its channel phrase associates output methods, format options and severity levels with a name that can be used with the category phrase to select how various classes of messages are logged. The basic `logging` syntax is as follows:

```
logging {
```

```
    channel channel_name {
          file name [versions number] [size bytes];
          | syslog daemon;
          | null;
          severity type;
          print-category yes_or_no;
          print-severity yes_or_no;
          print-time yes_or_no;
    };
    category category_name {
          channel_name; [ channel_name; ...]
    };
};
```

Only one logging section is used to define as many channels and categories as you want. If there are multiple *logging* sections in a configuration, the first one defined determines the logging, and warnings are issued for the others. If there is no logging section, the default logging configuration will be:

```
logging {
   category default { default_syslog; default_debug; };
   category panic { default_syslog; default_stderr; };
   category packet { default_debug; };
   category eventlib { default_debug; };
};
```

The following is an example:

```
logging {
   channel xfers {
         file “IP$:XFERS.LOG”;
         severity info;
         print-severity yes;
         print-time yes;
   };
   category xfer-in {
         xfers;
   };
};
```

Table 1.8 describes the logging options.

Table 1.8. Logging Options

Options	Description			
channel	Specifies where the logging data goes: to syslog (OPCOM), to a file, to stderr (SYS$ERROR), or to null (discarded).			
category	Specifies what data is logged. You can send a category to one channel or to many channels. These are the valid categories:			
	cname	lame-servers	packet	security
	config	load	panic	statistics
	db	maintenance	parser	update
	default	ncache	queries	xfer-in
	eventlib	notify	response-checks	xfer-out

Options	Description
	insist os
file	Specifies the path name of the file you want the message to go to, and if you want to have multiple versions of the file, and if you want to limit the size of the file.
syslog daemon	Specifies that the message goes to syslog (opcom) instead of to a file.
severity	Specifies the severity level for this channel. The severity choices are: critical, error, warning, notice, info, debug [level], and dynamic.
print-category print-severity print-time	Specifies whether to print the category, severity level, and time stamp of the messages. The default is NO for each item.

1.1.7. Resource Record Sorting

When returning multiple resource records (RRs), the name server returns them by default in round robin order, that is, after each request the first RR is placed at the end of the list. You can specify in the `NAMED.CONF` file that the name server should sort the RRs based on the client's address using the *`sortlist`* option, or you can use a default other than round-robin using the *`rrset-order`* option.

The *`sortlist`* option sorts the RRs based upon the *`address_match_list`*. Each top level statement in the address match list must be an address match list with one or two elements. The first element of each address match list is checked against the client's address until a match is found. When a match is found:

If the top level statement contains...	Then...
only one element	that network is moved to the front of the list.
two elements	the second element is treated like the address match list in the *`topology`* option (see Table 1.7).

Use the following *`sortlist`* statement to have the name server behave like the BIND 4.9.x name server.

Responses to...	
queries from the local host	prefers any of the directly connected networks
queries from any other hosts on a directly connected network	prefers addresses on that same network
other queries	are not sorted

```
sortlist {
   { localhost; localnets; };
   { localnets; };
};
```

The *`rrset-order`* option permits configuration of the ordering for the records in a multiple record response. An *`order_spec`* is defined as follows:

```
[ class class-name ] [ type type-name ] [ name fqdn ] order ordering;
```

The legal values for *ordering* are:

fixed	records are returned in the order they are defined in the zone file
random	records are returned in some random order
cyclic	records are returned in round-robin order (default)

The following example specifies that only A, NS, and MX records are round-robined. This provides the same behavior as the VSI TCP/IP 4.2 name server.

```
rrset-order {
   class IN type A name * order cyclic;
   class IN type NS name * order cyclic;
   class IN type MX name * order cyclic;
   order fixed;
};
```

1.1.8. Incremental Zone Transfer

BIND 9.7.2-p3 contains an implementation of incremental zone transfer (IXFR) -- it is on by default. If you need to turn it off for a particular slave server, use the *provide-ixfr server* substatement, which defaults to yes:

```
server ip_addr {
   provide-ixfr ( yes | no );
};
```

1.1.9. DNS Zone Information Files

The DNS server configuration file contains references to DNS zone information files, which contain control information and a list of resource records for objects in the zone. The file format of a zone information file is:

```
$INCLUDE filename [domain]
$ORIGIN domain
$TTL default-ttl
$GENERATE rangelhs type rhs
domain [ttl] [class] record-type resource_data
```

If the *domain* is specified as ".", the domain is the ROOT domain. If the *domain* is specified as an at sign (@), the domain is the current origin. Anything else is taken as a standard domain name, which if terminated by a dot (.) is used verbatim; otherwise, the current origin is appended to the specified domain name.

The optional *domain* field in a $INCLUDE line is used to define an origin for the data in an included file. It is equivalent to placing a $ORIGIN statement before the first line of the included file. The field is optional. Neither the optional *domain* field in the $INCLUDE line nor $ORIGIN statements in the included file modify the current origin for this file.

The $TTL statement sets the default time-to-live for records that do not have explicit *ttl* fields. If the zone file does not have a $TTL statement, the DNS server prints a warning on your computer screen and uses the minimum value from the SOA record.

The optional *ttl* field is an integer value to specify in the time-to-live field in the following ways:

Each of these is equivalent to one week.

- 604800
- 1w
- 7d
- 168h
- 10080m
- or any combination

For example: sigma 2h46m40s IN A 192.1.1.97

Loads the TTL as: ttl = 10000 (2 hours 46 mins 40 secs)

The default is to use the value specified in the $TTL directive or SOA resource record for the zone. The optional class field is the object address type; currently only one type (IN, for objects connected to the Internet) is supported.

The *record-type* field is also known as "resource record types." Table 1.9 shows the most commonly used DNS resource record types and their uses (the data expected in the *resource_data* field is shown in brackets [ ]).

Table 1.9. DNS Resource Record Types

Record Type	Use
A	A host address [IP-address]
CNAME	The canonical name for an alias [domain-name][1]
HINFO	Host information [CPU-type OS-type]
KEY	The publickey associated with a domain name [flags protocol algorithm key]
MB	A mailbox domain name [domain-name][2]
MG	A mail group member [domain-name][2]
MINFO	Mailbox or mail list information [request-domain error-domain]
MR	A mail rename domain name [domain-name]
MX	A mail exchanger [preference domain-name]
NS	An authoritative name server [domain-name]
NULL	A null resource record [none]
NXT	Used for secure negative responses. Tells a querier which record is lexicographically next in the zone [next-domain-name type-bitmap]
PTR	A domain name pointer [domain-name]
SIG	A security signature for an RRset [type algorithm labels ttl expiration inception tag name signature]
SOA	The start of a zone of authority [domain of originating host, domain address of maintainer, a serial number and the following parameters in seconds: refresh, retry, expire, and minimum time-to-live] (see RFC-1035)

Record Type	Use
SRV	Specifies the location of services [priority weight port target]
TXT	Arbitrary text [text-string]
WKS	A well-known service description [address protocol service-list]

[1]A canonical name is an alias name for a computer.

[2]An experimental resource record not in common use.

Resource records are usually single-line entries, but SOA records may be continued across lines by surrounding the resource record statements with open and close parentheses. Comments begin with semicolons and continue to the end of the line. Each zone information file should begin with an SOA resource record for the zone. An example SOA resource record follows.

```
@ IN  SOA VMSITE.FLOWERS.COM. .system.EMAIL.FLOWERS.COM. (
      2000022101 ; serial number as yyyymmddnn
      7200       ; refresh every 2 hours
      7200       ; retry every 2 hours
      12096000   ; expire in twenty weeks
      86400 )    ; minimum time-to-live
```

The SOA resource record lists a serial number that DNS administrators should increase each time they modify the master file. Secondary servers check the serial number at intervals specified by the refresh time. If the serial number has increased since the last zone transfer, the secondary name server requests a new zone transfer and then loads the new zone data.

In the preceding example, VMSITE is a host in the `FLOWERS.COM` domain. This should be the primary DNS server for this zone.

`system.EMAIL.FLOWERS.COM` is the email address of the DNS zone administrator on the EMAIL host in the `FLOWERS.COM` domain.

Note

Although the DNS zone administrator's email address is specified without an at sign (@), the effective email address requires changing the first period to @. For example, the email address of the DNS zone administrator in the preceding example is `system@EMAIL.FLOWERS.COM`.

If you configure your name server as a secondary name server, it contacts the primary name server for a new zone transfer after the refresh interval expires. If the server does not receive a response after the "retry" interval, it tries repeatedly to contact the primary name server until it succeeds. If the secondary server fails to contact the primary name server before the expire interval elapses, it discards all data from the zone.

The minimum value is the time-to-live value used by records in the file with no explicit value if there is no $TTL statement. It is used also as the time-to-live value for negative caching.

Set the expire time to a value long enough to accommodate the retry and refresh intervals. If the refresh interval exceeds the expiration time, the data on your secondary server will expire before new data can be loaded. The following example shows a zone information file for the zone `FLOWERS.COM`.

```
;
;   Authoritative data for FLOWERS.COM
;
```

```
$TTL 86400
@   IN   SOA VMSITE.FLOWERS.COM. system.EMAIL.FLOWERS.COM. (
         2000022107 ; serial number as yyyymmddss, where
                    ; ss is the zone change sequence count
         7200       ; refresh every 2 hours
         7200       ; retry every 2 hours
         12096000   ; expire in twenty weeks
         86400 )    ; minimum time-to-live
    IN   NS   VMSITE.FLOWERS.COM.
    IN   NS   SPACELY.SPROCKETS.COM.
;
;   Information about the host FLOWERS.COM
;
FLOWERS.COM.    IN   MX    0    FLOWERS.COM.
                IN   A          192.0.0.1
                IN   HINFO      HPE Integrity rx2800 i4 VMS
;
;   The loopback address and host
;
LOCALHOST.      IN   A          192.0.0.1
                IN   HINFO      LOOPBACK-HOST LOOPBACK
;
; The remaining hosts
;
AARDVARK        IN   A          192.0.0.2
                IN   MX         0           AARDVARK
                IN   HINFO      HPE Integrity rx2800 i4 VMS
John            IN   A          192.0.0.3
                IN   MX         0           John.FLOWERS.COM.
                IN   HINFO      HPE Integrity rx2800 i4 VMS
```

The SOA resource record indicates the start of authority for the zone `FLOWERS.COM`. The NS resource records indicate which DNS name servers are authoritative for the zone. The remainder of the file lists each host and information about it. VMSITE is a system in the `FLOWERS.COM` domain. The system. `EMAIL.FLOWERS.COM` is the email address on the EMAIL host in the `FLOWERS.COM` domain.

An example of a zone information file for the zone 0.128.IN-ADDR.ARPA follows. This file contains the information needed to map IP addresses in the network 128.0.0.0 into host names. This file contains an example of the PTR record type.

```
;
;   Authoritative data for 0.128.IN-ADDR.ARPA
;
$TTL 86400
@   IN   SOA VMSITE.FLOWERS.COM.   system.EMAIL.FLOWERS.COM. (
         2000022101   ; serial number as yyyymmddnn
         7200         ; refresh every 2 hours
         7200         ; retry every 2 hours
         12096000     ; expire in twenty weeks
         86400 )      ; minimum time-to-live
         IN   NS                   VMSITE.FLOWERS.COM.
         IN   NS                   SPACELY.SPROCKETS.COM.
;
; Network database
;
1.0      IN   PTR                  FLOWERS.COM.
2.0      IN   PTR                  AARDVARK.FLOWERS.COM.
```

```
3.0      IN   PTR                  JOHN.FLOWERS.COM.
```

1.1.9.1. Reloading the Name Server

To reload the running server:

```
$ IP NETCONTROL DOMAINNAME RELOAD
```

To restart the name server:

```
$ IP NETCONTROL DOMAINNAME RESTART
```

Note

The master server must be restarted before the name server restarts to load in any net-config or server-config changes **@IP$:start_server**.

1.1.9.2. Controlling the VSI TCP/IP DNS Server

You can use the VSI TCP/IP **NETCONTROL** server to request that the DNS server perform specific actions. The name to specify to **NETCONTROL** for the DNS server is DOMAINNAME.

Refer to the *VSI TCP/IP Administrator's Reference* for more information.

When the server is busy, **NETCONTROL** sends a message stating that your request has been queued, and it will be acted upon when it is the next one in the queue to be serviced. When the server is not busy, it performs your request while **NETCONTROL** waits (except for the case of RELOAD). For example,

```
DOMAINNAME>stat
<Dumping Nameserver Statistics
<Domain-Name-Server Busy, Request Queued
```

1.1.9.3. Using NSLOOKUP and DIG to Debug DNS

The VSI TCP/IP **NSLOOKUP** and DIG utilities send test queries to a DNS Name Server to test the configuration. **NSLOOKUP** enters interactive mode when invoked without arguments. Table 1.10 describes the valid interactive commands. For information about DIG and more information about **NSLOOKUP**, see the*VSI TCP/IP Administrator's Reference*.

Table 1.10. NSLOOKUP Commands

Command	Description
name	Prints information about *name* using the default server.
name server	Prints information about *name* using *server.*
exit	Exits NSLOOKUP.
help or ?	Prints help information.

Command	Description
`set all`	Prints the current status of all options.
`set class=`*`class`*	Sets the query class to one of these: IN, CHAOS, HESIOD, or ANY.
`set [no]debug`	Prints debugging information.
`set [no]d2`	Prints exhaustive debugging information.
`set [no]defname`	Appends the domain name to each query.
`set [no]recurse`	Asks for a recursive answer to query.
`set [no]vc`	Always uses a virtual circuit (TCP instead of UDP).
`set domain=`*`name`*	Sets the default domain name to *`name`*.
`set port=`*`port`*	Sets the port number on which to send a query.
`set retry=`*`n`*	Sets the number of retries to *n*.
`set srchlist=`*`name1`* [/*`name2`*/.../*`name6`*]	Sets the domain to *`name1`* and the search list to *`name1`* through *`name6`*.
`set timeout=`*`n`*	Sets the timeout interval to *n*.
`set querytype=`*`type`*`or` `set type=`*`type`*	Sets the resource record (RR) type to query for.
`server` *`name`*	Sets the default server to *`name`*, using the current default server.
`lserver` *`name`*	Sets the default server to *`name`*, using the original default server.

1.1.10. DNS Load Balancing

The VSI TCP/IP domain name server provides a feature called DNS load balancing, which is modeled after the service names available with HP's LAT terminal server support for OpenVMScluster systems. The domain name server maintains a load rating for each node offering a particular service name and, when queried for the addresses records for that name, it orders them based on the load rating. This allows TCP-based services such as TELNET and FTP to be offered cluster-wide in a load-balanced fashion. To configure DNS load balancing on each cluster node that will offer the service:

1. Set up cluster services offered by this node with NET-CONFIG (see Section 1.1.10.1).
2. If your host is multi-homed (one that has more than one IP network interface), specify the IP address associated with the cluster services with NET-CONFIG (see Section 1.1.10.2).
3. Configure service ratings for advertised services (see Section 1.1.10.3).
4. Add the new service names to your domain's DNS zone file (see Section 1.1.10.4).
5. Monitor and test the status of your cluster service names with the IP NETCONTROL DOMAINNAME SHOW command (see Section 1.1.10.5).

1.1.10.1. Setting Up a Cluster Service

To configure a cluster service name with NET-CONFIG:

```
$ IP CONFIGURE
NET-CONFIG>SET CLUSTER-SERVICE-NAMES name[,...]
```

Note

Do not confuse the DNS load balancing feature and the *CLUSTER-SERVICE-NAMES* parameter with the cluster alias feature and *IP-CLUSTER-ALIAS* parameter that is described in Section 2.2.10.15

Each cluster service name must be a name *not alreadyin use* on your network. Specify each name in its fully qualified form (for example, CLUSTER.FLOWERS.COM). If you configure more than one name, separate the names with commas when you specify them. To activate your cluster service names on the running system, use the command:

```
$ DEFINE/SYSTEM/EXEC IP$CLUSTER_SERVICE_NAMES name
```

If you are setting up multiple service names, enclose each name in quotation marks and separate the quoted names with commas. Once the logical name has been defined, restart the name server:

```
$ IP NETCONTROL DOMAINNAME RESTART
```

1.1.10.2. Advertised Cluster Service Addresses on Multi-Homed Hosts

For a multi-homed host, you can control the address advertised for a cluster service with the NET-CONFIG **SET CLUSTER-SERVICE-ADDRESS** command:

```
NET-CONFIG>SET CLUSTER-SERVICE-ADDRESS address
```

1.1.10.3. Setting Service Ratings

The load-rating algorithm used by the DNS server for cluster service names is similar to the algorithm used by LAT. It is based on the system load, the number of interactive processes on the system, and the amount of free physical memory on the system. If your OpenVMScluster system contains CPUs of vastly different speeds and/or memory configurations, you may find that the algorithm always favors a faster or larger-memory system over a slower or smaller-memory system.

In an unbalanced cluster configuration, you may need to set either a *static load rating* or weight the rating computation by setting a *CPU rating* on each node:

- Static Ratings — A static rating is set by defining a logical name:

  ```
  $ DEFINE/SYSTEM/EXEC IP$CLUSTER_SERVICE_STATIC_RATING n
  ```

  n is a decimal number ranging from 1 to 255.

  When defined, this rating will be used in all cluster service advertisements, bypassing the dynamic load rating computations. The higher the number you set, the more "available" (or less loaded) the system will be.

- CPU Ratings — A CPU rating is also set by defining a logical name:

  ```
  $ DEFINE/SYSTEM/EXEC IP$CLUSTER_SERVICE_CPU_RATING n
  ```

  n is a decimal number ranging from 1 to 100.

This is the weight value factored into the load rating calculation. To bias load ratings so that faster CPUs service more users, set a lower CPU rating value on your slower CPUs and a higher CPU rating on your faster CPUs. After setting ratings, restart the name server:

```
$ IP NETCONTROL DOMAINNAME RESTART
```

1.1.10.4. Adding Cluster Services to Your Domain's DNS Zone File

Once you have configured a cluster service name on your cluster, update your domain's primary name server DNS zone file to delegate authority over the cluster service name to the participating cluster nodes. To do this, add NS records that map each cluster service name to the participating nodes. For example, to add the cluster service name `CLUSTER.FLOWERS.COM`, add the following lines to the configuration files on `FLOWERS.COM`'s primary name server:

```
CLUSTER.FLOWERS.COM. IN NS NODE1.FLOWERS.COM.
CLUSTER.FLOWERS.COM. IN NS NODE2.FLOWERS.COM.
```

The name on the left side is the cluster service name; the name on the right side is the domain name of a node offering the cluster service.

1.1.10.5. Monitoring Cluster Service Names

To check the status of your cluster service names, use the following **NETCONTROL** command:

```
$ IP NETCONTROL DOMAINNAME SHOW
```

For each cluster service name, a listing of the nodes offering the service and their load ratings is displayed. For example:

```
$ IP NETCONTROL DOMAINNAME SHOW
Connected to NETCONTROL server on "NODE1"
 < Node1.Flowers.COM Network Control V10.5(10) at Mon 13-Mar-2017 03 4:35
 PM-EST
 <Service CLUSTER.FLOWERS.COM:
 < Nodename Address         Rating
 < -------- --------------- ------
< < NODE1   192.41.228.101  75
< < NODE2   192.41.228.102  83
< <End of line
```

You should also test a cluster service name using NSLOOKUP:

```
$ IP NSLOOKUP CLUSTER.FLOWERS.COM
Server:   LOCALHOST
Address:  192.0.0.1
Name:     CLUSTER.FLOWERS.COM
Addresses: 192.44.128.102, 192.44.128.101
```

NSLOOKUP should return the addresses from highest to lowest rating, although DNS caching can cause address ordering to lag behind rating changes for short periods of time.

1.1.11. DNS Security

BIND 9.7.2-p3 includes an implementation of DNS Security (DNSSEC). A complete description of DNSSEC and its use is beyond the scope of this chapter. DNSSEC is described in RFC 2065 "Domain Name System Security Extensions" and various internet drafts.

VSI TCP/IP includes the following BIND tools related to DNS Security:

- **DNSKEYGEN**: Used for generating and maintaining keys.
- **DNSSIGNER**: Used for signing zones.

For more information on these tools, see the *VSI TCP/IP Administrator's Reference*.

This section describes a simple scenario in which **DNSKEYGEN** and **DNSSIGNER** are used.

The simplest "normal" case is a zone, which has no delegations, and is to be signed with a single zone key. Assume that the parent zone is secured and is able to sign the public zone key.

The first step in signing a zone is to generate a private-publickey pair. This is done using **DNSKEYGEN**. This will generate a DNS zone master file version of the publickey in a file with the suffix "key".

1.1.11.1. Example of key generation:

```
$ IP dnskeygen/dsa=768/zone/noencrypt zz.test.
```

This result in the generation of two files, the names of which reflect the key owner, algorithm, and footprint. The names end in "key" and "private". The "key" file contains the DNS RR holding the publickey, the "private" file has the data defining the private key. The latter file is set to be read/write only by the file's owner.

1.1.11.2. Example key file (key represented in base64 characters):

```
zz.test. IN KEY 16641 3 3 AQPIc...
```

1.1.11.3. Example private file:

```
Private-key-format: v1.1
Algorithm: 3 (DSA)
Prime(p): base 64 characters
Subprime(q): base 64 characters
Base(g): base 64 characters
Private_value(x): base 64 characters
Public_value(y): base 64 characters
```

Note

The two numeric fields in the key filenames will be different for each time dnskeygen is run. Also note that the "private" key's format will depend on the algorithm used to derive the key.

The next step is to run **DNSSIGNER** over the data. To make things simple, all files involved will be considered to be in the current default directory unless otherwise stated. In the directory where the file `zone.1` resides there should be a "private" file for the key used for signing.

1.1.11.4. Example zone file (zone.1):

```
$ ORIGIN zz.test.
@        IN SOA a.test. a.a.test. 1 360 36 60480 12
         NS a.test.
         NS b.test.
```

```
one       A    10.10.10.10
two       A    10.10.10.100
          MX   10 one.zz.test.
a.test.   A    10.11.12.13
b.test.   A    10.13.12.11
```

The publickey (from the key file) is sent two different ways. One copy of the publickey is sent to the parent zone for signing with the parent's zone key. The publickey is also copied (or even $INCLUDEd) into the `zone.1` file. Signing may begin prior to receiving a response from the parent zone (which contains, among other things, the signed publickey).

Although the publickey is going to arrive from the parent at some time packaged with the signature, the unsigned key must be placed into the unsigned zone master file. The presence of the publickey record alerts **DNSSIGNER** to perform certain functions, such as generating NXT records and generating parent files for its delegation points.

Important

Although **DNSSIGNER** is flexible enough to withstand missing private keys, and late arriving parent files, it cannot be expected to behave correctly if the data used to derive the zone master file changes during the execution of the signing process. In accordance with this, the publickey should be added to the zone even though the key will also arrive from the parent later. **DNSSIGNER** will remove duplicate records.

1.1.11.5. Example of signing a zone:

```
$ IP dnssigner/zone=(inp=zone.1, out=zone.2) -
/sig=key=(dom=zz.test, alg=3, key_id=6750)
```

The result of the run will be a new zone file. The file `zone.2` will appear something like the following:

1.1.11.6. Example output of DNSSIGNER (zone.2):

```
$ ORIGIN zz.test.
zz.test.   12   IN   SOA   a.test.   a.a.test. 1 6M 36S 16h48m 12S
   SIG     SOA 1 12 19980223163147 19980123163147 6750 zz.test. (...)
zz.test.   KEY   0x4101 3 1 (...)
zz.test.   NS   a.test.
           NS   b.test.
   SIG     NS 1 12 19980223163147 19980123163147 6750 zz.test. (...)
zz.test.   NXT   one.zz.test. NS SOA SIG KEY NXT
   SIG     NXT 1 12 19980223163147 19980123163147 6750 zz.test. (...)
one        A   10.10.10.10
   SIG     A 1 12 19980223163147 19980123163147 6750 zz.test. (...)
one        NXT   two.zz.test. A SIG NXT
   SIG     NXT 1 12 19980223163147 19980123163147 6750 zz.test. (...)
two        A   10.10.10.100
   SIG     A 1 12 19980223163147 19980123163147 6750 zz.test. (...)
two        MX   10 one.zz.test.
   SIG     MX 1 12 19980223163147 19980123163147 6750 zz.test. (...)
two        NXT   zz.test. A MX SIG NXT
   SIG     NXT 1 12 19980223163147 19980123163147 6750 zz.test. (...)
a.test. A  10.11.12.13
b.test. A  10.13.12.11
```

All of the "(...)" fields are base64 encoded values. This file is complete except for the missing signature by test. over the zz.test. KEY record. If this file is sent to a secured name server, the zone data will be rejected unless the zone key happens to have been configured.

Note

It is wise not to configure the zone key for a zone unless the parent will not be signing the zone key.

Eventually, the parent file will arrive. After obtaining the file, the **DNSSIGNER** needs to be run again to include the new data.

1.1.11.7. Example parent file (parent.1):

```
zz.test. NXT   zzz.test. NS SIG KEY NXT
zz.test. SIG   NXT 1 12 19980229163147 19980129163147 12345 test. (...)
zz.test. KEY   0x401 3 1 (...)
zz.test. SIG   KEY 1 12 19980229163147 19980129163147 12345 test. (...)
```

The final run of **DNSSIGNER** is:

```
$ IP dnssigner/zone=(in=zone.2,out=zone.3)/parent=in=parent.1
```

Note

The specification of the key is no longer needed. However, now that the records are signed, **DNSSIGNER** will verify all the existing signatures.

In the case that a signature fails during these checks, the action taken by **DNSSIGNER** depends on whether the key of the signature is specified on the **DNSSIGNER** command line during the run. In the example, failing signatures are just dropped. If the run command included

```
/sig=key=(dom=zz.test.,alg=dsa,key_id=6750)
```

then failing signatures would be replaced.

The result of the second run of **DNSSIGNER** is `zone.3`, which is the final zone file and would be used by the name server. `zone.3` is a merger of `zone.2` and `parent.1`, minus the records which appear in both files; that is, duplicates are removed.

1.1.12. Multicast Name Resolution

Multicast name resolution aims to eliminate the need to maintain HOSTS files or configure a name server on networks that are contained within a single logical LAN. Systems participate by sending out a multicast request to resolve a name and any system that recognize the name responds to the request. Systems that participate in multicast name resolution use one of two protocols: LLMNR (RFC 4795), or mDNS. Both protocols use packets that are very similar to standard DNS packets; they operate in different multicast groups and use different port numbers.

VSI TCP/IP offers a responder that participates in both protocols and the ability to configure the resolver to use one of the two protocols. Using a multicast group disables one of the resolver's checks for authenticity of the answers that it receives. The multicast name responder works for both IPv4 and IPv6 addresses. Zero configuration of systems is one of the goals for IPv6 on small networks, and multicast name resolution helps in meeting this goal. Configuring VSI TCP/IP to use multicast name resolution involves enabling the server (LLMNR) and setting the name server address and port with

the **IP CONFIGURE/NETWORK** command. See the following example for configuring multicast name resolution:

```
$ IP configure/server

SERVER-CONFIG>ENABLE LLMNR
SERVER-CONFIG>WRITE
SERVER-CONFIG>EXIT

$ IP configure/network
NET-CONFIG> set multicast-name-resolution {LLMNR | mDNS}

NET-CONFIG>WRITE
NET-CONFIG>EXIT
LLMNR (port 5355 on 224.0.0.252 and FF02::1:3)
mDNS (port 5353 on 224.0.0.251 and FF02::FB)
$ define/system/exec/nolog IP$nameservers 224.0.0.252
$ define/system/exec/nolog IP$dns_port 5355
```

Chapter 2. Establishing IP Connectivity and Configuring Services

This chapter explains how to establish IP connectivity between your computer and other computers on your network. Connectivity depends upon the network interfaces in your system.

The second section of this chapter describes how to configure VSI TCP/IP services.

2.1. About IP Connectivity

Establishing IP connectivity ensures that users can perform the tasks described in the *VSI TCP/IP User's Guide*, including:

- Obtaining information about remote systems and users with FINGER and WHOIS
- Accessing files on other computers with the FTP, RCP, TFTP, SCP, and SFTP commands
- Logging into other computers with the RLOGIN, TELNET, and SSH commands
- Using remote printers

Note

Establishing IP connectivity only ensures you can reach other systems if you know their IP addresses. It does not ensure you can reach other systems by name. For example, there is no guarantee you can send mail to users on remote systems without first configuring host tables or the Domain Name System (DNS) (See Appendix B).

2.2. Network Interface Configuration Overview

At startup, VSI TCP/IP obtains global configuration data (such as the default route) and device-specific network interface configuration data (such as the IP address) from the following files:

- The `IP$:NETWORK_DEVICES.CONFIGURATION` network database file describes the current network configuration, including a list of the device interfaces you have specified. This file is used to determine which devices are present when the network is started.
- The `IP$:IP$SYSTARTUP.COM` network initialization command procedure starts and configures the network, and initializes individual device interfaces and global parameters. This file is overwritten each time you use NET-CONFIG to update and save the configuration.

Note

Network interface configuration changes take effect the next time your system reboots. In contrast, most global parameter changes do not require rebooting.

Some network interfaces require configuration data that is not accessible through either NET-CONFIG. When configuring such interfaces, additional configuration files are required. For details, see Section 2.2.4.1.

2.2.1. Supported Network Interface Devices

VSI TCP/IP allows you to configure multiple interface devices on your network. Each interface is named according to its type (for example, shared Ethernet interfaces are of the type "se").

In general, VSI TCP/IP accommodates a maximum of ten devices of each type. Table 2.1 lists the devices VSI TCP/IP supports and the interface names to use when specifying devices to be added or modified.

Table 2.1. Supported Network Interface Devices

Device	Interface Name
Raw packet (dbridge)	rp[0...49]
SLIP (Serial line IP) using any OpenVMS-supported terminal multiplexer	sl[0...49]
PPP (Point-to-Point Protocol)	ppp[0...49]
VMS Ethernet, FDDI, ATM, or Alpha Token-Ring controller	se[0...19]
STF (IPv6 encapsulated in IPv4)	stf0

VSI TCP/IP contains a driver for each of these device types. The driver either accesses the device directly or is an interface to an appropriate OpenVMS device driver.

2.2.2. Viewing Interface Configuration

You can view the currently configured network interfaces by using the NET-CONFIG **SHOW** command.

2.2.2.1. Viewing Interface Configuration with NET-CONFIG

You can use the NET-CONFIG **SHOW** command to display the maximum configuration or the current configuration.

2.2.2.2. Viewing the Maximum Configuration

Use the following command to display all interface types supported by VSI TCP/IP are displayed, including the default settings for the Adapter, CSR, Flags, and Vector parameters:

```
$ IP CONFIGURE /INTERFACE
NET-CONFIG>SHOW MAXIMUM
```

For example:

```
NET-CONFIG>SHOW MAXIMUM
Devices     Adapter                        CSR Address   FlagsVector
-------     -------                        -----------   -----------
rp[0-9]    (Raw Packet)                    -NONE-        -NONE-    -NONE-
ppp[0-49]  (Point-to-Point Protocol)       -NONE-        -NONE-    -NONE-
se[0-9]    (Shared OpenVMS Ethernet/FDDI) -NONE-        -NONE-    -NONE-
sl[0-49]   (Serial Line IP)                -NONE-        -NONE-    -NONE-
```

2.2.2.3. Viewing the Current Configuration

Use the following command to display global parameter settings and device interfaces currently in your network configuration (including the actual settings for Adapter, CSR, Flags, and Vector):

```
NET-CONFIG>SHOW [CURRENT]
```

For example:

```
NET-CONFIG>SHOW
Interface    Adapter   CSR Address       Flags/Vector
---------    -------   -----------        ------------
se0          (Shared OpenVMS Ethernet/FDDI)   -NONE-    -NONE-   -NONE-
   [TCP/IP$: 192.41.228.68, IP-SubNet: 255.255.255.0]
   [VMS Device: EZA0, Link Level: Ethernet]
ppp0         (Point-to-Point Protocol)    -NONE-        -NONE-   -NONE-
   [VMS Terminal: TTA0]
   [ACCM: 0x0, Authentication: None]
   [Protocol Compression: Off, Address and Control Field Compression: Off]
[Idle Timeout: 0, Configuration Timeout: 0]
[MRU: 0, ICMP Allowed: Yes]
[Configuration Retries: 0, Termination Retries: 0]
[TCP Header Compression: Disabled]
Official Host Name:             BANZAI.FLOWERS.COM
Default IP Route:               192.41.228.129
Domain Nameserver:              127.0.0.1
Timezone:                       EST
Default TFTP Directory:         IP_ROOT:[IP.TFTP]
Anonymous FTP Directory:        ANONVILLE:[ANONYMOUS]
Load UCX $QIO driver:           TRUE
Load PWIP (Pathworks) driver:   TRUE
```

The **SHOW** command with no parameters defaults to **SHOW CURRENT**.

2.2.2.4. Modifying the Configuration

When VSI TCP/IP starts, it obtains configuration parameter values from a set of configuration files found in the `IP$CONFIG:` directory. Once VSI TCP/IP is running, you can change parameters in two ways:

1. Modify configuration files using the various `IP CONFIGURE/`*`qualifier`* commands. After modification, restarts of VSI TCP/IP processes may be required, up to and including a system reboot.

2. Modify the current parameters in memory using the `IP SET` command or by defining VSI TCP/IP logicals.

VSI TCP/IP provides a command-line interface to modify the configuration parameters. Follow the steps in this section to modify your configuration. Using

1. **Check Logicals:** Before running IP CONFIGURE, make sure the logical name IP$ is defined. This logical name is defined during installation, but can be undefined manually or by rebooting without starting VSI TCP/IP automatically. To verify this logical name is defined, enter:

   ```
   $ SHOW LOGICAL IP$*
   ```

   If the logical name is undefined, issue one of the 2 following commands:

```
$ @SYS$STARTUP:IP$LOGICAL_NAMES

$ @SYS$STARTUP:IP$STARTUP_LOGICAlS
```

If the logical name is previously defined, executing this command does not cause a problem.

2. **Start IP CONFIGURE:** To start the command-line configuration utilities, enter:

   ```
   $ IP CONFIGURE /qualifier
   ```

   where */qualifier* specifies the VSI TCP/IP feature that you want to configure.

   The following table lists VSI TCP/IP utilities and how to enable them by command line.

Table 2.2. Command-Line Utilities

Configuration Type	DCL Command	Configuration Utility
Network Interfaces	`$ IP CONFIGURE /INTERFACE`	NET-CONFIG
Electronic Mail	`$ IP CONFIGURE /MAIL`	MAIL-CONFIG
Remote Print Queues	`$ IP CONFIGURE /PRINTERS`	PRINTER-CONFIG
VSI TCP/IP Servers	`$ IP CONFIGURE /SERVERS`	SERVER-CONFIG
VSI TCP/IP DECnet-over-IP Circuits	`$ IP CONFIGURE /DECNET`	DECNET-CONFIG
VSI TCP/IP NFS Server	`$ IP CONFIGURE /NFS`	NFS-CONFIG

3. **Getting and Loading Alternate Configuration Files**

   By default, each command-line configuration utility modifies the configuration files that VSI TCP/IP reads when it starts. Usually, you only need to modify configuration information and exit from the utility to change a configuration file. To maintain multiple configuration files, however, use the configuration utility `GET` command to load and replace the configuration currently loaded. This feature is useful for abandoning modified configurations without quitting the utility.

   For example, to load a test configuration file named `IP$:TEST.CONFIGURATION` into `NET-CONFIG`, enter:

   ```
   NET-CONFIG> GET IP$:TEST.CONFIGURATION
   ```

   Directions for other services are listed in Table 2.4

4. **Create an alternate configuration file:** You must create an alternate file first because by default VSI TCP/IP

   Use the WRITE command to save the alternate configuration file as in the following example:

   ```
   NETCONFIG> WRITE filename
   ```

5. **Modifying the Configuration**

   Once you have loaded a configuration file into the configuration utility, you can modify the configuration with any commands the utility offers. Most utilities provide an assortment of SET and ADD commands.

After you modify the configuration, you can save the file under the same name or a different name. Although all command-line configuration utilities allow you to maintain multiple configuration files under different names, VSI TCP/IP only reads the standard configuration files, all of which are in the IP$: directory. Table 2.3 lists these files.

Table 2.3. General Configuration Files

Service	File name and Description
Device Configuration Listing	NETWORK_DEVICES.CONFIGURATION — Lists network devices and their configurations. This file is maintained by NET-CONFIG, and is read when VSI TCP/IP starts. *Do not edit this file!*
General	IP_VERSION — Contains the VSI TCP/IP version and revision level. *Do not edit this file!*
Host Tables	HOSTS.LOCAL — Contains the local host table, with protocols and services for your local network. This file can be edited; use a text editor to maintain this file.
	HOSTS.SERVICES — Contains the official protocols and services supported by the local host (refer to RFC-943 for additional information). *Do not edit this file!*
	HOSTS.TXT — Contains the DDN NIC host table, which lists the host names and Internet addresses of the Internet hosts known to the NIC.
	HOSTTBLUK.DAT — Contains a compiled, binary version of the two host table files IP$:HOSTS.LOCAL and IP$:HOSTS.TXT. It contains the names of all hosts (in alphabetical order) in the host table files. VSI TCP/IP utilities refer to HOSTTBLUK.DAT when attempting to complete a partially specified host name. *Do not edit this file!*
	NETWORK_DATABASE — A compiled, binary version of the host, protocol, and services information contained in the files IP$:HOSTS.LOCAL, IP$:HOSTS.SERVICES, and IP $:HOSTS.TXT. The NETWORK_DATABASE file allows applications to quickly look up the name or address of any host known in the host table files. *Do not edit this file!* When DNS is disabled, the NETWORK_DATABASE file provides the sole host lookup facility. When DNS is enabled, this database is only referred to if a name service query fails.
Printing	REMOTE-PRINTER-QUEUES.COM — Configuration file for customizing printer queues. You must use IP CONFIGURE/ PRINTER to modify this file. You cannot edit it directly.
“R” services	HOSTS.EQUIV — Used by BSD "R" services to perform authentication on incoming connections. This file can be edited
Servers	SERVICES.MASTER_SERVER — Contains a list of the VSI TCP/IP servers and their configurations. This file is maintained by SERVER-CONFIG and is used by the master server process. *Do not edit this file!*

Service	File name and Description
Routing	GATED.CONF — Used by the Gateway Daemon (GATED) service to configure the RIP, EGP, HELLO, OSPF, and BGP dynamic IP routing protocols. This file can be edited; use a text editor to maintain this file.

Table 2.4. VSI TCP/IP Specific Configuration Files

Feature	To enable, enter:	File and Description

BOOTP	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE BOOTP`
	`SERVER-CONFIG>RESTART`
		BOOTP-SERVER.CONFIGURATION — Provides booting information for disk-less hosts using the BOOTP protocol. This file can be edited; use a text editor to maintain this file.

DHCP	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE DHCP`
	`SERVER-CONFIG>RESTART`
		DHCPD.CONF — Provides booting information for disk-less hosts using the DHCP or BOOTP protocol. This file can be edited; use a text editor to maintain this file.

DNS	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE DOMAINNAME`
	`SERVER-CONFIG>RESTART`
		NAMED.CONF — Contains configuration information for the Internet DNS (Domain Name System) and the locations of other name service database files. This file can be edited.

NFS	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE NFS`
	`SERVER-CONFIG>ENABLE RPCMOUNT`
	`SERVER-CONFIG>ENABLE RPCQUOTAD`
	`SERVER-CONFIG>ENABLE RPCPORTMAP`
	`SERVER-CONFIG>ENABLE RPCLOCKMGR`
	`SERVER-CONFIG>ENABLE RPCSTATUS`
	`SERVER-CONFIG>RESTART`
		NFS_EXPORT.DAT, NFS_GROUP.DAT, NFS_MNTLST.DAT, and NFS_PROXY.DAT — Contain configuration information for the VSI TCP/IP NFS Client and NFS Server products. *Do not edit these files!*

NTP	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE NTP`
	`SERVER-CONFIG>RESTART`
		NTP.CONF — Contains configuration information for the Network Time Protocol (NTP) service. This file can be edited.

RARP	`IP CONFIGURE/SERVER`
	`SERVER-CONFIG>ENABLE RARP`
	`SERVER-CONFIG>RESTART`
		RARP.CONFIGURATION — Used by the VSI TCP/IP RARP server to determine Ethernet-to-IP address mappings so hosts and PCs can query the server to obtain their IP address. This file can be edited.

SNMP	`IP CONFIGURE/SERVER` `SERVER-CONFIG>ENABLE SNMP`	SNMPD.CONF — Contains configuration information for

	SERVER-CONFIG>RESTART	the VSI TCP/IP SNMP (Simple Network Management Protocol) agent. Edit this file manually.
SMTP	$ @IP$:START_SMTP or $ @START_SMTP_LOCAL	SMTP_ALIASES — Contains SMTP (Simple Mail Transfer Protocol) mail alias and mailing list expansions. This file can be edited. START_SMTP.COM and START_SMTP_LOCAL.COM — Contain SMTP configuration information. *Do not edit these files.*
SSH	$ IP CONFIGURE/SERVER SERVER-CONFIG>SELECT SSH SERVER-CONFIG>SHOW/FULL SSH Service "SSH": ***DISABLED*** INIT() = Merge_Image Program = "IP$:LOADABLE_SSH_CONTROL" Priority = 5 Parameters = “enable-ssh2” SERVER-CONFIG> SET PARAMETER Add Parameter: ENABLE-SSH2 Add Parameter: [Service specific parameters for SSH changed] SERVER-CONFIG>EXIT	SSH2_DIR:SSHD_CONFIG - Contains configuration information for the SSH server. This file can be edited; use a text editor to maintain this file. SSH2_DIR:SSHD2_CONFIG - Contains configuration information for the SSH2 server. This file can be edited; use a text editor to maintain this file.
TELNET	$ IP CONFIGURE/SERVERS SERVER-CONFIG>SELECT TELNET	MAP3270.DAT — Contains a keymap used by the TN3270 module of the TELNET client. The keymap maps keystrokes on standard ASCII terminals to IBM 327x special function keys. This file can be edited; use a text editor to maintain this file.

6. **Verifying the Configuration**

   Because modified configuration files are not read until the next time VSI TCP/IP starts, it is useful to verify the validity of your changes before restarting. The NET-CONFIG and SNMP-CONFIG configuration utilities provide a CHECK command specifically for verifying your configuration. To verify changes made with either of these configuration utilities, use the CHECK command at the configuration utility prompt.

Note

All VSI TCP/IP configuration utilities automatically execute the CHECK function when you exit.

   If there is a problem in the configuration, CHECK issues an error or warning message describing the problem.

If the utility displays error messages, use the SHOW command to view your changes and spot the error that caused the message. Correct the error using the configuration utility, then use the CHECK command to confirm the validity of the configuration.

The following example shows the automatic operation of CHECK after a NET-CONFIG session. Because the user executed an ADD command but specified a nonexistent OpenVMS device, CHECK issued an ERROR message. Because the user gave no IP address, CHECK issued a WARNING message.

```
NET-CONFIG>ADD SL1
[Adding new configuration entry for device "sl1"]
VMS Device [TTA0] TXA0
Baud Rate: [UNSPECIFIED]
Header Compression Mode: [DISABLED]
IP Address: [NONE]
Point-to-Point Device IP Destination Address: [NONE]
IP SubNet Mask: [NONE]
[add (Serial Line IP): Csr=NONE, Flags=%X0]
NET-CONFIG>EXIT
ERROR: sl1 can't $ASSIGN to SLIP Device:
%SYSTEM-W-NOSUCHDEV, no such device available
WARNING: sl1 has no protocol addresses specified
This network configuration FAILED the sanity check.
Write startup file anyway ? [NO] NO
```

7. **Exiting IP CONFIGURE**

   To quit a configuration utility without saving your changes, use the QUIT command and enter NO when prompted to save the configuration.

   To save the configuration and exit:

   - Use the EXIT command, which automatically saves the configuration without prompting.
   - Use the QUIT command and enter YES when prompted to save the configuration.

2.2.2.5. Modifying the Current Configuration

The IP SET command provides a mechanism for system managers to manipulate the current configuration without restarting the IP$SERVER process or rebooting the system. *Use IP SET with caution.*

You can use IP SET to:

- Modify local timezone information
- Modify DECnet-over-IP circuits
- Modify network interface configuration
- Manipulate the ARP table
- Manipulate the routing table

If you use the `/SNMP_HOST` qualifier, IP SET can affect ARP tables, routing tables, and network interface configuration of remote hosts running a MIB-II-compliant SNMP agent, such as the VSI TCP/IP SNMP agent (see the VSI TCP/IP Installation and Administrator's Guide: Volume II).

For a detailed description of IP SET, refer to the *VSI TCP/IP for OpenVMS Administrator's Reference*.

Note

Because IP SET affects only the current configuration and does not affect the configuration files, any changes made with IP SET are lost the next time you start VSI TCP/IP.

2.2.3. Adding Network Interfaces

To add a new network interface to your VSI TCP/IP configuration, you can use NET-CONFIG (see Section 2.2.4).

Table 2.5 lists all network interface parameters used by VSI TCP/IP-supported interfaces. Table 2.6 lists the parameters required by each type of interface.

2.2.3.1. Network Interface Parameters

The supported network devices can be classified into three categories that determine the parameters you enter when configuring the device:

- Hardware devices with which VSI TCP/IP communicates directly. For each of these devices, NET-CONFIG requires that you specify the CSR and the UNIBUS adapter into which the device is plugged. Most of the devices also require you to specify device-specific parameters.

Some of these devices have programmable interrupt vectors that you specify with NET-CONFIG; VSI TCP/IP programs these vectors during startup. Others have interrupt vectors that are determined by the hardware. For each of these devices, set the vector and the CSR on the hardware using the DIP switches or jumpers on the card as described in the device manual. As described in Table 2.5, each interrupt vector must be unique.

- Hardware devices through which VSI TCP/IP communicates using a OpenVMS device driver. For these devices, NET-CONFIG requires that you identify the OpenVMS device through which VSI TCP/IP is to communicate; for most of these devices, you must also specify device-specific parameters.
- Software, or pseudo, devices whose interfaces communicate with software and for which no hardware is directly associated. These interfaces are typically used to implement special-purpose transports and deliver packets to other software. For example, the IP-over-DECnet interface encapsulates IP packets in DECnet datagrams for transmission over a DECnet network. All parameters for these devices are device-specific.

Table 2.5 lists the prompts that appear when you run NET-CONFIG. Make sure you respond to at least one network address prompt so the device can be started from the boot process. Table 2.6 lists the prompts displayed for each device type.

Table 2.5. Network Interface Parameters

Parameter	Function
ACCM Mask	Asynchronous Control Character Map Mask. A 32-bit mask that indicates the set of ASCII control characters to be mapped into two-character sequences for transparent transmission over the line. Default is %x00000000.
Adapter	Identifies the UNIBUS to which the device is connected. The setting can be the name of a UNIBUS (UBA0, UBA1, UBA2, or UBA3), or ANY,

Parameter	Function
	which tells VSI TCP/IP to search each UNIBUS until it finds a device at the specified CSR.
Address and Control Field Compression (ACFC)	When ON, PPP eliminates the address and control fields when they are identical over a series of frames. Default is OFF.
Baud Rate	Indicates the transmission baud rate. Valid settings are 110, 300, 1200, 2400, 4800, 9600, 19200, and UNSPECIFIED.
BSD Trailer Encapsulation	For 10Mb/sec Ethernet controllers only. Can be used to enable Berkeley Trailer encapsulation of IP packets on those Ethernets. Two valid settings: NEGOTIATED or DISABLED (the default, which prevents the use of trailer encapsulation).
Communications Mode	For communications devices that share a dialup line with either a modem or a terminal. Use DTE (Data Transmit Enable, the default) to specify that the line can originate serial communications, or DCE (Data Carrier Enable) to specify the opposite.
CSR	Control Status Register. Identifies the device's octal bus address. The CSR is usually programmed by setting DIP switches or jumpers on the card as described in the device's documentation.
Flags	Some devices have a Flags prompt whose meaning is device-dependent.
Hardware Device	The name of the real Ethernet device; for example, se0.
Header Compression Mode	For SLIP devices, indicates whether to use Van Jacobson's header compression algorithm. The parameter has three valid settings: • DISABLED — Indicates that headers should never be compressed (default). • ENABLED — Indicates that headers should always be compressed. • NEGOTIATED — Indicates that headers should not be compressed until a compressed header is received from the other side. At least one side of a link must be ENABLED for compression to be used; that is, both sides of a link cannot be set to NEGOTIATED.
ICMP	When ENABLED (the default), PPP allows ICMP packets over the PPP connection. Administrators may want to disable ICMP packets if they are concerned with "service attacks" from dial-up connections.
IP Address	Indicates the Internet address associated with the interface.
IP Address of Remote System	Indicates the Internet address of the system to which the interface will connect.
IP Broadcast Address	Used with devices that support broadcasts. Allows the setting of a non-standard IP broadcast address; defaults to an address with a host portion of all 1's.
IP Over DECnet Peer Host's DECnet Name	Used with IP-over-DECnet links to indicate the name of the DECnet node on the other end of the connection.
IP SubNet Mask	Allows setting a non-standard IP subnet mask.

Parameter	Function
IPv6 global address	Indicates the global unique address associated with this interface. The interface may also have a link-local address, which will be automatically generated when the interface is started.
IPv6 mask length	The length of the mask for the IPv6 address.
Jumbo Frames	Used with Ethernet devices to indicate whether to use standard length Ethernet packets (1500 bytes) or larger (9000 bytes) Jumbo frames. Jumbo frames can provide a higher throughput rate because more data is processed on a single interrupt.
Link Level Encapsulation Mode	Valid setting is ETHERNET.
Maximum Receive Unit (MRU) Size	Determines the maximum number of 8-bit bytes for the PPP Information field, including padding, but not including the Protocol field. Because opposite ends of a PPP connection may have different MRU values, PPP negotiates a suitable MRU for both systems. Default: 500.
Point-To-Point Device IP Destination Address	Used with point-to-point interfaces to indicate the IP address of the system on the other side of the line.
Protocol Compression	When ON, PPP negotiates with the peer to use one byte instead of two for the Protocol fields to improve transmission efficiency on low-speed lines. Default: OFF.
Retry Count	Determines the number of attempts PPP makes to configure a connection with "Configure-Request" packets. Default: 0.
Termination Retry Count	Determines the number of attempts PPP makes to terminate a connection with "Terminate-Request" packets. Default: 0.
Timeout	Determines the time (in seconds) between successive Configure-Request or Terminate-Request packets. Default: 0.
VMS Device	Used with devices that use a OpenVMS device driver to interface to the hardware. Indicates the name of the OpenVMS device that VSI TCP/IP is to use. This parameter is used with Ethernet interfaces and SLIP interfaces.
Vector	Used with programmable vector devices only. Identifies the interrupt vector that VSI TCP/IP assigns to the device during the boot process. Each interrupt vector (both fixed and programmable types) must be unique. Refer to Section 2.2.3.2 for an example of how to display the current system interrupt vectors.

Note

If your network requires a network interface to be initialized with parameters other than those listed in Table 2.5, create a custom initialization command procedure as described in theSection 2.2.4.1.

Table 2.6. Interfaces and Parameters

Type	Description
se	**Interface name:** se0, se1, se2, ... se19 **Device type:** Any OpenVMS Ethernet, FDDI, ATM, or Alpha Token-Ring controller

Type	Description
	Parameter Prompt Example Value: VMS Device: XEA0 Link Level Encapsulation Mode: ETHERNET BSD Trailer Encapsulation: DISABLED IP Address: 192.41.228.70 IP SubNet Mask: 255.255.255.0 Non-Standard IP Broadcast Address: 192.41.228.71 DHCP CLIENT [DISABLED]: DISABLED Jumbo Frames [DISABLED]: ENABLED IPv6 on this interface [DISABLED]: ENABLED IPv6 global address [NONE]: 3FFE:1200:3006::C673:8EBE IPv6 mask length: 48 The se interface uses any Ethernet controller to provide access to a 10/100/1000 Mb/s Ethernet network, any FDDI controller to provide access to a 100 Mb/s FDDI network, and any Alpha Token-Ring controller to provide access to 4 Mb/s or 16 Mb/s Token-Ring networks. The se interface uses the standard OpenVMS Ethernet driver to allow VSI TCP/IP to share the Ethernet device with other protocols such as LAT, LAVC, and DECnet.
sl	**Interface name:** sl0, sl1, sl2, ...sl49 **Device type:** Any OpenVMS-supported terminal interface **Parameter Prompt Example Value:** OpenVMS Device: TTA0 Baud Rate: 19200 Header Compression Mode: DISABLED IP Address: 192.41.228.70 Point-To-Point Device IP Destination Address: 192.41.228.71 IP SubNet Mask: 255.255.255.0 The VSI TCP/IP software supports SLIP, reducing the size of the headers and increasing the bandwidth available to data. Header compression mode is determined by what both sides can support.
rp	**Interface name:** rp0, rp1, rp2, ...rp49 **Device type:** Raw packet **Parameter Prompt Example Value:** IP Address: 192.41.228.70

Type	Description
	IP SubNet Mask: 255.255.255.0 The rp interface allows IP packets, normally destined for transmission, to be directed instead to a user process by way of an AF_RAWPACKET socket.
ppp	**Interface name:** ppp0, ppp1, ppp2, ...ppp49 **Device type:** Any supported PPP terminal interface **Parameter Prompt Example Value:** VMS Device: TTA0 Baud Rate: 19200 PPP ACCM Mask: 0 PPP Authentication Method: None PPP Protocol Compression: OFF PPP Address and Control Field Compression: OFF PPP Retry Count: 0 (If 0, defaults to the compiled-in value of 10.) PPP Idle Timeout: 0 (If 0, defaults to the compiled-in value of 300 seconds.) PPP MRU Size: 0 PPP ICMP: ENABLED PPP TCP Compression: OFF PPP Termination Retry Count: 0 (If 0, defaults to the compiled-in value of 10.) PPP Timeout: 0 (If 0, defaults to the compiled-in value of 30 seconds.) IP Address: 0.0.0.0 Point-To-Point Device: 0.0.0.0 IP Destination Address: IP SubNet Mask: 255.255.255.0
stf	**Interface name:** stf0 - only one six to four interface can exist on a system. Note that the **CREATE** command is used to create this interface. **Parameter Prompt Example Value:** IPv4 address to use [NONE]: 192.168.1.2 Mask length [48]: 48

2.2.3.2. Displaying Interrupt Vectors

You can display the current interrupt vector used by OpenVMS by invoking the SYSMAN utility, then using the **SHOW /CONFIGURATION** command, as follows:

```
$ MCR SYSMAN
SYSMAN>SHOW /CONFIGURATION
```

You can also display the maximum device configuration and the vectors currently used by VSI TCP/IP by invoking NET-CONFIG as shown in Section 2.2.2.1.

2.2.4. Adding Network Interfaces with NET-CONFIG

To add an interface to the configuration:

1. Start NET-CONFIG:

   ```
   $ IP CONFIGURE /INTERFACES
   ```

2. At the NET-CONFIG prompt, enter:

   NET-CONFIG>ADD *interface_name*

   interface_name is a name from the Table 2.1.

   Do not use an interface name currently in use; to modify an existing interface, see Section 2.2.5.

   For example, to add a third shared Ethernet (se) interface to your network configuration, enter:
   NET-CONFIG>**ADD SE2**

   NET-CONFIG prompts you for interface parameter values required by the *interface_name* interface.

3. Enter interface configuration parameter values at each NET-CONFIG prompt. For descriptions of the required parameters for your network interface, refer to the Table 2.6.

4. When the NET-CONFIG prompt returns, verify the validity of the new interface configuration with the **CHECK** command.

5. If the **CHECK** command produces error messages, view the configuration parameters with the **SHOW** command to determine the cause of the error. To correct the error, modify the configuration as described in Section 2.2.5 or abandon your changes with the **GET** command (which reloads the configuration file) and repeat from Step 2.

6. If the **CHECK** command produces no error messages, quit NET-CONFIG with the **EXIT** command.

   Your changes take effect after the next VSI TCP/IP reload.

   ```
   $ IP CONFIGURE
   VSI TCP/IP for OpenVMS Network Configuration Utility 10.5(nnn)
   [Reading in MAXIMUM configuration from IP$:IP.EXE]
   [Reading in configuration from IP$:NETWORK_DEVICES.CONFIGURATION]
   NET-CONFIG>ADD SL0
   [Adding new configuration entry for device "sl0"]
   VMS Device [TTA0] TTA2
   Baud Rate: [UNSPECIFIED]
   ```

```
Header Compression Mode: [DISABLED]
IP Address: [NONE] 196.22.19.1
Point-To-Point Device IP Destination Address: [NONE] 196.22.19.2
IP SubNet Mask: [NONE]
[sl0 (Serial Line IP): Csr=NONE, Flags=%X0]
NET-CONFIG>EXIT
[Writing configuration to IP$:NETWORK_DEVICES.CONFIGURATION]
[Writing Startup file IP$:IP$SYSTARTUP.COM]
[Changes take effect after the next OpenVMS reboot]
```

2.2.4.1. Creating a Custom Interface Initialization Procedure

If your network requires that a device be initialized with parameters not supported by NET-CONFIG, you can create a custom initialization command procedure for the device. At network startup, VSI TCP/IP uses this file instead of the commands for the device in the `IP$:IP$SYSTARTUP.COM` file.

The device must already be part of the network configuration, and commands to its interface must already exist in the `IP$:IP$SYSTARTUP.COM` file. To change the device's initialization, create a command file using a text editor:

1. Create a file named `IP$:`*`interface`*`_CONFIGURE.COM`, *`interface`* is an interface in your configuration.
2. Copy the section of `IP$:IP$SYSTARTUP.COM` containing the initialization commands into the new file.
3. Edit the new file to specify the new initialization.

2.2.5. Modifying Network Interfaces

To modify the configuration of an existing interface:

1. Start NET-CONFIG:

   ```
   $ IP CONFIGURE /INTERFACES
   ```

2. At the NET-CONFIG prompt, enter:

   ```
   NET-CONFIG>MODIFY interface_name
   ```

   `interface_name` is the name of the network interface you want to modify.

   For example, to modify the third shared Ethernet interface in your network configuration, enter:

   ```
   NET-CONFIG>MODIFY SE2
   ```

   NET-CONFIG prompts you for interface parameter values required by the specified interface.

3. Enter interface configuration parameter values at each of the NET-CONFIG prompts. For descriptions of the required parameters for your network interface, refer to the Table 2.6.
4. When the NET-CONFIG prompt returns, verify the validity of the new interface configuration with the **CHECK** command.
5. If the **CHECK** command produces error messages, view the configuration parameters with the **SHOW** command to determine what is causing the error. Correct the error by repeating from Step 2, or abandon your changes with the **GET** command (which reloads the configuration file) and repeat from Step 2.

6. If the **CHECK** command produces no error messages, quit NET-CONFIG with the **EXIT** command.

   Your changes take effect the next time your system reboots.

2.2.6. Deleting Network Interfaces

You can delete network interfaces from your VSI TCP/IP configuration by using NET-CONFIG (see Section 2.2.6.1)

2.2.6.1. Deleting Network Interfaces with NET-CONFIG

Use NET-CONFIG to delete one or all interfaces from the current configuration:

1. Start NET-CONFIG:

   ```
   $ IP CONFIGURE /INTERFACES
   ```

2. At the NET-CONFIG prompt, enter:

   ```
   NET-CONFIG>DELETE interface_name
   ```

   interface_name is the name of the existing network interface you want to delete.

   For example, to delete the third shared Ethernet interface in your network configuration, enter:

   ```
   NET-CONFIG>DELETE SE2
   ```

3. When the NET-CONFIG prompt reappears, verify the validity of the new interface configuration with the **CHECK** command.

4. If the **CHECK** command produces error messages, view the configuration parameters with the **SHOW** command to determine the cause of the error. Correct the error by repeating from Step 2, or abandon your changes with the **GET** command (which reloads the configuration file) and repeat from Step 2.

5. If the **CHECK** command produces no error messages, quit NET-CONFIG with the **EXIT** command.

   Your changes take effect the next time your system reboots.

2.2.6.2. Enabling and Disabling Interfaces

You can disable and re-enable interfaces individually. If you disable a device, it is not configured when the network is started.

To disable a device, use the **DISABLE** command; for example:

```
NET-CONFIG>DISABLE SE0
```

To re-enable a device, use the **ENABLE** command; for example:

```
NET-CONFIG>ENABLE SE0
```

2.2.6.3. Assigning Multiple Addresses to a Network Interface

Sometimes it is necessary to assign multiple IP addresses to a single physical interface; for example, when multiple subnets or networks are running on a single network segment.

You do this by using a pseudo device interface (pd). Use NET-CONFIG to add the device as you do other devices (such as se devices). VSI TCP/IP supports up to 500 pseudo device interfaces. Instead of specifying the OpenVMS device name, however, specify the VSI TCP/IP name for the interface (for example, se0). You must reboot the system after adding the pseudo device.

Caution

Careless assignment of a secondary address can cause network problems. In general, you should assign pseudo devices (pd) addresses on the same network or subnet as the se device to which the pd device is linked.

If the pd interface is not in the same IP network as its associated se interface, some TCP/IP packages (such as early versions of SunOS) retransmit broadcast packets for the other IP network back to the network segment from which they were transmitted. This can cause network storms.

Note

Some services listen to traffic on se interfaces only and ignore traffic on pd interfaces. One such service is the RIP listener in GATED.

The following example shows how to add a pseudo device:

```
$ IP CONFIG
VSI TCP/IP for OpenVMS Network Configuration Utility 10.5
[Reading in MAXIMUM configuration from IP$:IP.EXE]
[Reading in configuration from IP$:NETWORK_DEVICES.CONFIGURATION]
NET-CONFIG>SHOW
Interface                          Adapter   CSR Address   Flags/Vector
---------                          -------   -----------   ------------
se0 (Shared OpenVMS Ethernet/FDDI) -NONE-    -NONE-        -NONE-|
   [TCP/IP$: 161.44.128.20, IP-SubNet: 255.255.255.0]
   [VMS Device: ESA0, Link Level: Ethernet]|
Official Host Name:             lobo.process.com
Domain Nameservers:             127.0.0.1
Timezone:                       EST
Timezone Rules:                 US/EASTERN
Load UCX $QIO driver:           TRUE
Load PWIP (Pathworks) driver:   TRUE
Nameserver Retrans Timeout:     9   (6 Retries)
WHOIS Default Server:           rs.internic.net
NET-CONFIG>ADD PD0
[Adding new configuration entry for device "pd0"]
Hardware Device: [NONE] SE0
IP Address: [NONE] 161.44.128.21
IP SubNet Mask: [NONE]
Non-Standard IP Broadcast Address: [NONE]
[pd0 (Secondary Ethernet Address): Csr=NONE, Flags=%X0]
NET-CONFIG>SHOW
Interface                           Adapter   CSR Address   Flags/Vector
---------                           -------   -----------   ------------
se0 (Shared OpenVMS Ethernet/FDDI)  -NONE-    -NONE-        -NONE-
   [TCP/IP$: 161.44.128.20, IP-SubNet: 255.255.255.0]
   [VMS Device: ESA0, Link Level: Ethernet]
pd0 (Secondary Ethernet Address)    -NONE-    -NONE-        -NONE-
   [TCP/IP$: 161.44.128.21]
```

```
   [Hardware-Device: se0]
Official Host Name:             lobo.process.com
Domain Nameservers:             127.0.0.1
Timezone:                       EST
Timezone Rules:                 US/EASTERN
Load UCX $QIO driver:           TRUE
Load PWIP (Pathworks) driver:   TRUE
Nameserver Retrans Timeout:     9   (6 Retries)
WHOIS Default Server:           rs.internic.net
NET-CONFIG>EXIT
```

2.2.7. Using Packet Filtering for Security

Packet filtering is used today in almost all (from basic to sophisticated) security firewalls. Packet filtering *firewalls* apply filtering rules to each packet received to determine whether to accept or discard it. These filtering rules specify the protocol, source and destination IP addresses, and destination ports (for TCP and UDP) for accepted or discarded packets.

You use packet filtering on routers between an internal network and one or more external networks (such as a connection to the Internet). Packet filter rules restrict what may come in through the interface connected to the external network.

Packet filtering can also be useful on hosts. For example, you can restrict the hosts that are allowed access to services. In particular, these are UDP-based services and services that the VSI TCP/IP master server does not activate, and thus cannot use incoming access restrictions.

Note

When you use packet filtering, each datagram received on the interface is filtered. This increases processing overhead depending on the size of the filter list.

Packet filtering can be an effective and useful security mechanism; however, it cannot solve all your security problems. To be effective, you must construct the filtering rules carefully.

Both ipv4 and ipv6 addresses may be filtered.

2.2.8. Cautions When Creating Packet Filters

Observe the following cautions when setting up packet filtering on an interface:

- Packet filtering does not use state information. Each datagram is filtered without any knowledge of packets that preceded it. This means that for UDP-based applications, it is not possible to add a rule that says to accept replies to requests. This also affects connection-oriented protocols, such as FTP, that use two connections: one for commands and the other for data.

- Fragmented datagrams for UDP or TCP are difficult to filter, since only the first fragment has the necessary port information. VSI TCP/IP solves this problem by applying the filter rules to only the first fragment of UDP and TCP datagrams. The other fragments are accepted and processed or forwarded, but are eventually discarded because they cannot be reassembled without the first fragment. For all other IP protocols, the filter rules apply to each fragment.

- To set up secure packet filtering lists, you need detailed knowledge of IP, ICMP, TCP, UDP and applications protocols.

2.2.9. Packet Filter File

Packet filtering uses a filter list to determine whether you can receive a datagram. Filter lists are in packet filter files having the .DAT extension by default. Create one of these files first and then edit the file using the formats described in the following table.

Note

The format of the individual filter source address and mask, and destination address and mask, has changed. In previous releases, these were specified as an IPv4 address and IPv4 mask (e.g., “192.168.0.11 255.255.255.0”). This has been changed to use addresses and masks specified in CIDR (Classless InterDomain Routing) format (e.g., “192.168.0.11/24”). This not only makes the specification of addresses and masks clearer, it also allows for the implementation of IPv6 addresses which are substantially longer than IPv4 addresses, leading to potential problems with long filter file lines. The FILTER_CONVERT utility is provided to change existing filter files from the old format to the new format.

Table 2.7. Fields in a Packet Filter Entry

Field...	With valid values...	Means...
action protocol	*saddr [sport]*	*daddr [dport [doption]]*
action	permit deny drop	permit allows the datagram; deny denies the datagram and sends the ICMP; drop denies the datagram without sending an ICMP destination unreachable message to the sender.
protocol	ip *ip-number*tcp udp icmp	Protocol to check: the values indicated or the numeric IP protocol number. The value IP matches any IP protocol.
saddr	Example: 192.168.123.123/24	Source IP address to check in CIDR format. This may be in ipv4 or ipv6 format.
sport	operator *operand* lt *port* le eq| ge gt ne	Optional source port specification to check (for TCP and UDP entries only). Consists of an operator, space, and port name or number. If port name, must be defined in the `IP$:HOSTS.SERVICES` file. If omitted, any source port is valid. Example: gt 1023
daddr	Example: 192.168.123.123/240	Destination IP address to check in CIDR format. This may be in ipv4 or ipv6 format, but must be of the same address family as *saddr*
dport	*operator operand* lt *port* le *icmp-msg-type*eq	Optional destination port (for TCP and UDP entries) or ICMP message type specification consisting of an operator, space, and operand. If a port name, must be in the `IP$:HOSTS.SERVICES` file. If *icmp-msg-type*, use:

Field...	With valid values...	Means...
	ge gt ne	0-Echo Reply 3-Destination Unreachable 4-Source Quench 5-Redirect 8-Echo 11-Time Exceeded 12-Parameter Problem 13-Timestamp 14-Timestamp Reply 15-Information Request 16-Information Reply
doption	established	Matches only established connections (TCP segments with ACK or RST bits set).
start time		VMS-format absolute or delta date/time. If specified, the filter takes effect starting at the time specified. By default, a filter takes effect when loaded by VSI TCP/IP if the start time is not defined.
end time		Absolute or delta OpenVMS-format date/time. If specified, the filter is ignored after the time specified is reached if an absolute time is specified, or after the time calculated by adding the end time to the start time if a delta time is specified. If no start time was specified, the delta time is added to the current time. If the end time for a non-repeating filter has already passed when the filter definition is parsed, the filter is not loaded and the user is informed of that fact.
repeat		If Y and start/end times are specified, this filter repeats until it is removed. Both a start and end time must be specified in order to specify a repeating filter.
log		Logs events from this filter.

Each entry specifies a packet filtering condition for a particular protocol type, source or destination address and mask, and destination port and mask specification, with certain additional options. The system looks at each condition in sequence, looks for a match, and takes a permit (accept) or deny (reject) action. The system stops testing conditions after the first match. This means that the order of

the entries in the file is important; if the file lists a subsequent condition for an address, the system ignores it.

An implicit deny terminates the list of entries in the packet filter file. This means that if no condition matches, the system rejects the datagram. To use packet filtering:

1. Create address list entries in the packet filter file.
2. Apply the list to interfaces on your system by using packet filtering commands.

To create a packet filter file, edit a file and add address list entries in the format described.

Any number of spaces or tabs can separate each entity. Lines beginning with an exclamation point (!) are comment lines. You can use the dash (-) continuation character at the end of a line that needs to continue onto the next.

To apply the list to a particular network interface or interfaces on your system, use the **IP SET/INTERFACE/FILTER** command, as described in *VSI TCP/IP Administrator's Reference*.

2.2.9.1. Configuration Recommendations

Constructing an address filter list requires care in that you want to allow only the packets you need. Here are some recommendations in setting up an address filter list for an interface:

- Add an entry to prevent IP "spoofing"-having an external host send a datagram as if it came from a local machine. For a router, this makes sense because no datagram received from an external network should ever have a local source address. Add the following entry to the filter list for the external interface:

  ```
  deny ip local-network
  ```

- Be careful with services that use "unprotected" port numbers (greater than 1024). Some examples are NFS (port 2049) and X Windows (port 6000 and higher). Explicitly denying these services is a good idea:

  ```
  deny udp 0/0 0/0 eq 2049
  deny tcp 0/0 0/0 eq 2049
  deny tcp 0/0 0/0 eq 6000
  deny tcp 0/0 0/0 eq 6001
  ```

- Prevent broadcast and loopback packets from entering your network. It is best to restrict the broadcast (the first two of the following entries) to an external interface; apply the loopback restriction (the last entry) to any interface:

  ```
  drop ip 0.0.0.0/32
  drop ip 255.255.255.255/32
  drop ip 127.0.0.0/8
  ```

- Guard against datagrams from invalid source addresses when connected to the Internet (provided you are not using these network numbers for internal-only traffic purposes). Add the following to the filter list for the external interface:

  ```
  drop ip 10.0.0.0/8
  drop ip 172.16.0.0/12
  drop ip 192.168.0.0/16
  ```

- You generally need to allow domain name (DNS) requests using:

  ```
  permit udp 0/32 eq 53 0 0
  ```

Whether to allow TCP DNS traffic (usually used for zone transfers) is also something to consider. To disallow TCP DNS traffic, add:

```
deny tcp 0/32 eq 53 0 0
```

- You should not be concerned with what services local users use in the external world. You would want to add:

  ```
  permit tcp 0/32 0/32 gt 1023 established
  ```

  This allows all TCP datagrams in to ports greater than 1023 that have either the ACK or RST bits set in the TCP flags. Connection establishment requests have just the SYN bit set, so they are not allowed by this entry.

  You might want to drop the established option if you want to allow incoming connections to unprotected ports. This would allow use of the FTP PASV capability.

- You may offer services to the external world such as a World Wide Web or anonymous FTP server. Add the following entries:

  ```
  permit tcp 0/32 web-server-address/32 eq 80
  permit tcp 0/32 ftp-server-address/32 eq 21
  ```

  If you have several hosts for each service, add an entry for each.

Note

For the FTP Server, the data connections are normally outgoing and thus the earlier permit TCP 0 0 0 0 gt 1023 established configuration works to allow these. However, if users switch to PASV mode, the connections will be incoming (to unprotected port numbers) and therefore the permit TCP 0 0 0 0 gt 1023 configuration (without the established option) might be more effective.

- Allow all ICMPs except ICMP redirects:

  ```
  deny icmp 0/32 0/32 eq 5
  permit icmp
  ```

  This is useful for informing hosts about problems. But it can open up denial of service attacks, especially if hosts are not careful about the ICMP redirects they accept. That is why discarding them is recommended.

- Watch the order of the entries in the table carefully.

  ```
  permit tcp 0/32 0/32 gt 1023
  deny tcp 0/32 0/32 eq 2049
  ```

  This entry would not work since the permit entry allows the datagram and processing stops as soon as a match is found. VSI TCP/IP processes the entries in the order in which you specify them.

- Remember that an implicit "deny everything" is added to the end of the filtering list. This means that to permit a datagram, you need to have a permit entry in the list.

- Once you applied your filter list, test it first. Get an account on a host on an outside network that you can use to connect to your local hosts. Check that you are not allowing any access you do

not want, and that you are allowing access that you do want. If something is not right, modify the filter list, reload it, and retest.

While packet filtering is very useful, it is by no means the only step you should take to secure your network. You must take special care to secure the system to assure that it cannot be compromised. One way to do this is to greatly limit the services it offers.

2.2.9.2. Filtering by Time

Filters may be set to be active only during a specified time period. These filters may be specified as a one-time filter or as a filter that repeats. For example, a filter may be set up to filter all traffic from a specific address during the hours of 5am to 5pm each day, or a filter may be specified that filters traffic starting from the time the filter is loaded and for the next 3 hours.

Time-based filtering is done by specifying a start time, an end time, or both start and end times for a filter in the filter definition file. For repeating filters, both start and end times must be specified. Note that all time values for start and end times must be specified in OpenVMS absolute or delta time format. For example, the following are all valid:

- “29-MAR-2017” would be interpreted as "29-MAR-2017 <current time>
- “29-MAR-2017 18:03” would be interpreted as "29-MAR-2017 18:03:00.00"
- 18:03:00 would be interpreted as "<current date> 18:03:00.00"
- “1 15:00” would be interpreted as a delta time of 1 day, 15 hours

Note that if an absolute time is specified that contains both a date and time (example 2 above), it MUST be enclosed by double quotes. For example:

```
deny icmp 0 0 eq 5 start 17:00:00 end "29-MAR-2017 6:00:00"
Given the following filter file:
deny tcp 15.1.94.2/32 2.22.2.5 255/32 start 15:20 end 18:30 repeat
deny tcp 1.1.94.2/32 207.225.29.51/32 end "1-JAN-2017 18:30"
deny tcp 195.101.94.209/32 207.225.29.51/32 start 18:00 end "1 00:30"
 repeat
deny tcp 195.101.94.209/32 207.225.29.51/32
deny tcp 195.101.94.209/32 207.225.29.51/32 start 17:00 end 18:30
deny tcp 15.1.94.2/32 2.22.2.5/32 start "2 00:00" end 3 00:00"
```

Line 1 will filter from 15:20 to 18:30 each day.

Line 2 will filter from the time the filter is loaded through 18:30 on January 1, 2017 with no logging. After that time, if the filters are reloaded, this filter will not be loaded.

Line 3 will filter from 18:00 to 19:30 each day.

Line 4 has no time limits on it.

Line 5 will log from 17:00 through 18:30 today.

Line 6 will filter starting 2 days from the time the filter is loaded, through 3 days after that.

2.2.9.3. Filter Logging

Filter "hits" may be logged, either to OPCOM or to a file defined by the user. Logging is enabled on a filter-by-filter basis, by using the "log" keyword on the end of a filter definition line. For example:

```
deny TCP 192.10.9.209/32 207.225.29.51/32 log
```

Logging for the interface is controlled via the **IP SET/INTERFACE** command. The actual logging is performed by the IP_FLOG process, which is started the first time an **IP SET/INTERFACE / LOG** command is issued (a single IP_FLOG process handles logging for all interfaces defined on the system).

The **IP SET/INTERFACE** command switches used to support logging are:

Qualifier	Values	Default	Description
/[NO]LOGGING	OPCOM or a valid filename	None	Used to turn logging on or off. Filter events may be logged to OPCOM or to a specified file. Only those events with the LOG qualifier in their definition are affected by this qualifier.
/FORMAT	COMMA or NORMAL	NORMAL	If NORMAL, then the normal formatting as seen by **IP SHOW/INTERFACE/ FILTER** will be used. If COMMA, then a comma-delimited file will be created that can be, for example, loaded into a spreadsheet.
/INTERVAL	Number of seconds, between 5 and 2^31	5 seconds	Reporting interval. The minimum reporting interval is 5 seconds, so that a flood of filter events does not drag the system down. When reporting events, a count of missed events will be included for each event where the event could not be reported before the next event occurred.

When filter logging is enabled, the IP_FLOG process will be started. This process checks each interface at the interval defined by the **/INTERVAL** qualifier for the **IP SET/INTERFACE** command. As unlogged filter hits are found, it will log them to OPCOM or to a file, according on the parameters set by the **/LOG** and **/FORMAT** qualifiers for the **IP SET/INTERFACE** command.

When logging to OPCOM, only NORMAL formatting is allowed. An OPCOM message, formatted as the filter output from **IP SHOW/INTERFACE /FILTER**, will be displayed for each filter with unlogged hits on it.

When logging to a file, the output will be identical to that of the filter displays from **IP SHOW/ INTERFACE /FILTER** command, if **/FORMAT=NORMAL** is specified. If **/FORMAT=COMMA** is specified, the data will be recorded as comma-delimited fields, one line per filter, to the file. The first line of this file will contain the field names (comma-delimited) to aid in interpreting the contents of the file. Examples:

```
$ IP SET /INTERFACE SE0 /LOG=OPCOM/INTERVAL=10
```

enables logging to OPCOM, with a reporting interval of 10 seconds.

```
$ IP SET /INTERFACE SE0 /LOG=FOO.DAT/FORMAT=COMMA
```

enables logging to the file FOO.DAT in comma-delimited format, and a reporting interval of 5 seconds (the default).

```
$ IP SET /INTERFACE SE0 /NOLOG
```

This disables all logging for the interface, closing all open log files.

2.2.9.4. Setting the Filter List at Startup

When you start VSI TCP/IP, the START_IP procedure looks for a `IP$:FILTER-`*`interface`*`.DAT` file for each interface it starts. If the file exists, START_IP issues the following command to set the filter list for the interface:

```
$ IP SET/INTERFACE interface /FILTER=IP$:FILTER-line-id.DAT
```

You can also add the necessary IP commands to the `IP$:LOCAL_ROUTES.COM` file.

Note

The contents of `IP$:LOCAL_ROUTES.COM` will not be populated if DHCP client is being used. Please contact VSI support for a workaround.

If you want to know if filtering is enabled and what the settings are, use the **IP SHOW / INTERFACE/FILTER SE0** command.

VSI TCP/IP also supports the use of a **LOCAL_INITIALIZATION** command procedure during startup. As with the `IP$:LOCAL_ROUTES.COM` file, you can put the necessary VSI TCP/IP filter commands in the `IP$:LOCAL_INITIALIZATION.COM` file to have them executed as VSI TCP/IP starts.

2.2.9.5. Converting an Old-Format Filter File

The FILTER_CONVERT utility is provided to convert from the old-format filter file (one which uses separate address/mask fields) to the new-format filter file (one which uses CIDR format address specification). To use this:

```
$ FILTER_CONVERT :==
$ IP$:FILTER_CONVERT
$ FILTER_CONVERT infile outfile
```

When a filter file has been converted, the resulting output file should be checked for correctness prior to using it.

2.2.10. Configuring Transport over Serial Lines with SLIP and PPP

VSI TCP/IP supports remote IP transport over serial lines with SLIP (Serial Line IP) or PPP (Point-to-Point Protocol).

2.2.10.1. Understanding SLIP and PPP

Both SLIP and PPP use a simple framing protocol to transfer datagrams over a terminal line. Both require an RS232-C serial line, or one that looks like an asynchronous line to OpenVMS.

VSI TCP/IP SLIP interfaces (identified by the name **sl**) and PPP interfaces (identified by the name **ppp**) can send and receive packets over any asynchronous terminal line (OpenVMS device of types TTcn or TXcn) connected to other systems that support the SLIP and PPP protocols, respectively.

As a result, VSI TCP/IP systems can communicate asynchronously with other VSI TCP/IP systems (or UNIX and other systems) that support SLIP or PPP.

With SLIP or PPP, users on one system can connect with another system using modems over telephone lines or over hardwired connections. To use SLIP or PPP over modems, the modems must be 8-bit transparent so all 256 ASCII codes can be sent and received.

If the remote system is configured as a gateway to a network, local users can also reach other systems on that network. VSI TCP/IP supports SLIP and PPP running over any OpenVMS-supported terminal multiplexer. VSI TCP/IP does not support SLIP or PPP over LAT.

You must know the IP address of the serial interface on every remote host with which you establish serial communications. If you configure VSI TCP/IP with PPP interfaces for multiple remote hosts, the remote hosts can obtain their IP addresses when they connect. Similarly, you can configure a PPP interface on VSI TCP/IP without knowing your own IP address, and obtain it when you connect to a remote system.

The two methods of connecting hosts via PPP or SLIP are *dynamic* and *static*. The following sections explain these methods.

2.2.10.2. Dynamic Interfaces-Defined

The usual SLIP or PPP configuration consists of two systems connected by serial line only when needed. For these situations, configure a *dynamic* SLIP or PPP interface. Dynamic interfaces are not associated with a specific OpenVMS device until the remote host connects to a device.

For a dynamic interface, you do not specify an OpenVMS device name when configuring the interface. When VSI TCP/IP starts, new dynamic interfaces are available for serial communication; however, an administrator must attach the interfaces to OpenVMS devices.

2.2.10.3. Static Interfaces-Defined

Large organizations often use SLIP and PPP to connect separate LANs into a single wide area network (WAN) with dedicated serial lines. The host at each end of the serial connection is always the same, and no other hosts are allowed to connect to either serial device. In these situations, configure a *static* SLIP or PPP interface. Static interfaces are attached to a specific OpenVMS device, which prevents the serial device from being used for any other purpose.

For a static interface, you specify an OpenVMS device name when configuring the interface.

As soon as you connect two static interfaces via modem or hardwired connection, they can communicate over the chosen serial protocol; no user authentication is required.

Note

Because IP connectivity is established as soon as the two serial interfaces connect, do not configure static interfaces for public dial-in access.

2.2.10.4. Configuring Static SLIP Interfaces

To configure a static SLIP interface:

1. Use NET-CONFIG to add the SLIP interface as described in Section 2.2.3.

   Table 2.8 describes the interface parameters you must define for basic SLIP operation. Space is provided in the table so you can write down the values appropriate for your configuration.

Be sure to specify an OpenVMS device name.

2. If desired, create a custom startup command procedure for the new interface. For details, see Section 2.2.10.10.

3. Reboot your system.

   When VSI TCP/IP starts, you can connect your serial device to a remote system.

2.2.10.5. Configuring Dynamic SLIP Interfaces

To configure a dynamic SLIP interface:

1. Use NET-CONFIG to add the SLIP interface as described in Section 2.2.3.

   Table 2.8 describes the interface parameters you must define for basic SLIP operation. Space is provided in the table so you can write down the values appropriate for your configuration.

   Do not specify an OpenVMS device name.

2. If desired, create a custom startup command procedure for the new interface as described in the Section 2.2.10.10.

3. Reboot your system.

4. The new dynamic interfaces are created when VSI TCP/IP starts, but they are not yet connected to a OpenVMS device.

2.2.10.6. SLIP Configuration Parameters

The following table lists the configuration parameters for configuring static and dynamic SLIP interfaces.

Table 2.8. SLIP Configuration Parameters

Parameter	Description	Your Value
SLIP interface name	Determines the interface name. Must be of the form sl*n*, • *n* is a positive integer. • "sl0" is a suitable interface name for the first SLIP interface. • "sl1" is a suitable interface name for the second one.	
OpenVMS device name	For a *static* SLIP interface (one that uses a hardwired terminal line or dedicated modem and telephone line), specify a device name. For a *dynamic* SLIP interface (one to which you assign a device name when the connection is made, for example, by modem dial-up), do not specify a	

Parameter	Description	Your Value
	name. When a modem hangs up on a dynamic SLIP interface, each end of the SLIP interface automatically reverts to a normal terminal line. Use a value of "none" to override any specified device name. Doing so might be useful to make a previously configured interface dynamic.	
Baud rate	Data transfer baud rate (110, 300, 1200, 2400, 4800, 9600, 19200, or UNSPECIFIED) of the SLIP interface.	
SLIP compression mode	VSI TCP/IP SLIP supports to reduce the bandwidth required for the TCP and IP headers. If both sides of a SLIP interface support compression, turnaround improves significantly. Compression modes are: • **ENABLED** — Headers should always be compressed. • **DISABLED** — Headers should never be compressed. • **NEGOTIATED** — Headers should not be compressed until a compressed header is received from the other side. Negotiated compression is useful on dialup gateways that do not know if the other side of SLIP interfaces support compression. **Note** Compression must be enabled (not just negotiated) on at least one side of a SLIP interface. Disable SLIP compression for compatibility with SLIP interfaces that do not support it.	
IP address	IP address associated with the SLIP interface on the local system.	
Point-to-point IP destination address	IP address associated with the SLIP interface on the target system.	

2.2.10.7. Configuring Static PPP Interfaces

To define a static PPP interface:

1. Use NET-CONFIG to add a PPP interface to your network configuration as described in Section 2.2.3.

   Table 2.9 describes the interface parameters you must define for basic PPP operation. Space is provided in the table so you can write down the values appropriate for your configuration.

   Be sure to define an OpenVMS device for the interface.

2. If desired, create a custom startup command procedure for the new interface. For details, see Section 2.2.10.10.

3. Reboot your system.

   After rebooting, the interfaces are attached to OpenVMS terminal lines.

2.2.10.8. Configuring Dynamic PPP Interfaces

To define a dynamic PPP interface:

1. Use NET-CONFIG to add a PPP interface to your network configuration as described in Section 2.2.3.

   Table 2.9 describes the interface parameters you must define for basic PPP operation. Space is provided in the table so you can write down the values appropriate for your configuration.

   Do not define an OpenVMS device for the interface.

2. Enable virtual terminal (VTA) devices for your dynamic PPP interfaces. For information on configuring virtual terminals, see your OpenVMS system management documentation.

Note

Dynamic PPP interfaces require CMKRNL, LOG_IO, and SYSPRV privileges. Because these privileges are potentially dangerous, login name and password information should be safeguarded carefully.

---

3. If desired, create a custom startup command procedure for the new interface. For details, see Section 2.2.10.10.

4. Reboot your system.

   The dynamic interfaces are created when VSI TCP/IP starts, but they are not associated with a OpenVMS terminal line. The interfaces must now be attached to terminal lines.

2.2.10.9. PPP Configuration Parameters

The following table lists the configuration parameters for static and dynamic PPP interfaces.

Table 2.9. PPP Configuration Parameters

Parameter	Description	Your Value
PPP interface name	Determines the interface name. Must be of the form ppp*n*, • *n* is a positive integer.	

Parameter	Description	Your Value
	• "ppp0" is a suitable interface name for a first PPP interface.	
OpenVMS device name	Dedicates the device name to a *static* PPP interface (one that uses a hardwired terminal line or dedicated modem and telephone line). To configure a *dynamic* PPP interface (one to which you assign a device name when a connection is made-for example, by modem dial-up), do not specify a name. When a modem hangs up on a dynamic PPP interface, each end of the PPP connection automatically reverts to a normal terminal line. Use a value of "none" to override any specified device name. Doing so might be useful to make a previously configured interface dynamic.	
Baud rate	Determines the data transfer baud rate (110, 300, 1200, 2400, 4800, 9600, 19200, or UNSPECIFIED) of the PPP interface.	
Asynchronous Control Character Map (ACCM) Mask	A 32-bit mask that indicates the set of ASCII control characters to be mapped into two-character sequences for transparent transmission over the line. Default: %x00000000. The map is sent the most significant 8 bits first. Each numbered bit corresponds to the ASCII control character of the same value. If the bit is cleared to zero, the corresponding ASCII control character need not be mapped. If the bit is set to one, the corresponding ASCII control character must remain mapped. For example, if bit 19 is set to zero, the ASCII control character 19 (DC3, Control-S) can be sent in the clear. The least significant bit of the least significant 8 bits (the final 8 bits transmitted) is numbered bit 0, and corresponds to the ASCII control character "NUL."	

Parameter	Description	Your Value
Protocol Compression	When ON, PPP negotiates with the peer to use one byte instead of two for the Protocol fields to improve transmission efficiency on low-speed lines. Default: OFF.	
Address and Control Field Compression (ACFC)	When ON, PPP eliminates the address and control fields when they are identical over a series of frames. Default: OFF.	
Retry Count	Determines the number of attempts PPP makes to configure a connection with "Configure-Request" packets. Default: 10.	
Idle Timeout	Determines how long (in seconds) the connection must remain idle before PPP attempts to close the connection with "Terminate-Request" packets. Default: 300.	
Maximum Receive Unit (MRU) Size	Determines the maximum number of 8-bit bytes for the PPP Information field, including padding, but not including the Protocol field. Because opposite ends of a PPP connection may have different MRU values, PPP negotiates a suitable MRU for both systems. Default: 1500.	
ICMP	When ENABLED, PPP allows ICMP packets over the PPP connection. Administrators may want to disable ICMP packets if they are concerned with "service attacks" from dial-up connections. Default: ENABLED.	
TCP Header Compression	When ENABLED, requests the IPCP driver to employ Van Jacobson TCP header compression to improve performance. Default: DISABLED.	
Termination Retry Count	Determines the number of attempts PPP makes to terminate a connection with "Terminate-Request" packets. Default: 10.	
Timeout	Determines the time (in seconds) between successive Configure-Request or Terminate-Request packets. Default: 30.	
IP Address	The IP address of the local PPP interface in dotted-decimal format. You may also specify 0.0.0.0 (the default) to indicate that the local IP address will be	

Parameter	Description	Your Value
	specified by the remote peer when the serial connection is established.	
Point-To-Point Device IP Destination Address	The IP address of the peer PPP interface in dotted-decimal format. Default: 0.0.0.0.	
SubNet Mask	The subnet mask of the local PPP interface in dotted-decimal format. The default depends on the local PPP interface IP address. For example, a class An address results in a default subnet mask of 255.0.0.0.	

2.2.10.10. Configuring Permanent SLIP and PPP Interfaces

When you configure an interface, the following line is added to your `IP$SYSTARTUP.COM` file to initialize the device:

```
$ SET TERM/PERM/NOTYPE_AHEAD/NOAUTOBAUD/SPEED=config_speed dev_name
```

- *`config_speed`* is the baud rate for transmitting and receiving data.
- *`dev_name`* specifies a SLIP or PPP interface such as TTA1:.

This setting is used for permanent interfaces to prevent LOGINOUT from gaining control of the device because of extraneous noise on the line.

If you want to override this behavior, create a custom command procedure, `IP $:`*`dev_name`*`_CONFIGURE.COM`, for that interface. If you have an IP address assigned to the device, you custom command procedure is invoked at startup instead of the command line described previously.

2.2.10.11. Attaching Dynamic SLIP or PPP Interfaces to OpenVMS Devices

After a remote host connects to a VSI TCP/IP system over a serial line, the remote system administrator must log into the VSI TCP/IP system and attach the appropriate VSI TCP/IP dynamic interface to the OpenVMS terminal line to which the remote host is connected.

For example, if your system provides serial access via two modems, but you need to provide access to three or more hosts, configure a dynamic interface for each host you plan to accommodate. Then make sure the administrator on each remote host knows the name of the serial interface you have configured and the commands to execute to attach SLIP or PPP interfaces to the appropriate terminal lines.

To convert a terminal line into a SLIP or PPP interface:

1. Log into an account with CMKRNL, LOG_IO, and SYSPRV privileges.
2. Determine the name of the serial interface corresponding to the remote host. If the administrator knows the remote IP address or host name, the corresponding serial interface name can be determined from the IP SHOW command output. For example:

```
$ IP SHOW /STATISTICS
VSI TCP/IP for OpenVMS Network Interface statistics:
Name   Mtu   Network Address         Ipkts    Ierrs   Opkts    Oerrs
 Collis
----   ---   ------- -------         -----    -----   -----    -----
 ------
se0    1500   ABC-NET FOO.BAR.COM    120360   0       143384   1       4
sl0    296    ABC-NET FOO.BAR.COM    0        0       1        0       0
lo0    1536   LOOPBACK-NET LOCALHOST 917      0       917      0       0
$ IP SHOW /INTERFACE
_Network Device: SL0
Device sl0: flags=71<UP,POINTOPOINT,NOTRAILERS,RUNNING>
IP Address = 192.41.228.78
IP Sub-Net Mask = 255.255.255.192
IP Point to Point Destination = 192.41.228.80
```

3. Attach the serial interface to the OpenVMS device with the following DCL command:

   ```
   $ IP SET /INTERFACE interface_name -
   _$ /DYNAMIC/VMS_DEVICE/LINK_LEVEL=protocol
   ```

   protocol is SLIP or PPP, according to the type of serial interface.

Note

If the remote host is also a VSI TCP/IP system, the remote administrator must also attach the remote serial interface to the remote OpenVMS device.

The following example illustrates how to establish connectivity between two VSI TCP/IP systems over dynamic SLIP interfaces. The first **IP SET /INTERFACE** command converts the remote host's dial-in port into a SLIP interface. After the user types the control-backslash **Ctrl**/\ escape key sequence to return to a local DCL command line, the second **IP SET /INTERFACE** command converts the local terminal line into a SLIP interface. The **IP PING** command confirms IP connectivity.

```
WHARFIN$ ALLOCATE TTA1:
%DCL-I-ALLOC, _WHARFIN$TTA1: allocated
WHARFIN$ SET TERMINAL/SPEED=2400 TTA1:
WHARFIN$ SET HOST/DTE TTA1:
%REM-I-TOEXIT, connection established, type ^\ to exit
ATDT1415555-1212
RRING
CONNECT 2400
BIGBOOTE SLIP Gateway Username: SYSTEM
Password:
Welcome to the BIGBOOTE SLIP Gateway
Last interactive login on Monday, 28-FEB-2017 14:47
Last non-interactive login on Monday, 28-FEB-2017 13:16
BIGBOOTE$ IP SET/INTERFACE SL1/DYNAMIC/LINK_LEVEL=SLIP/VMS_DEVICE
The line you are logged in over is now a SLIP line.
[You now type the Ctrl/\escape sequence to return to WHARFIN.]
%REM-S-END, control returned to node _WHARFIN::
WHARFIN$ IP SET/INTERFACE SL1/DYNAMIC/LINK_LEVEL=SLIP/VMS_DEVICE=TTA1:
WHARFIN$ IP PING 192.0.0.3
PING 192.0.0.3 (192.0.0.3): 56 data bytes
64 bytes from 192.0.0.3: icmp_seq=0 time=860 ms
```

```
64 bytes from 192.0.0.3: icmp_seq=1 time=860 ms
64 bytes from 192.0.0.3: icmp_seq=2 time=860 ms
Ctrl/C
----192.0.0.3 PING Statistics----
4 packets transmitted, 3 packets received, 25% packet loss
round-trip (ms) min/avg/max = 860/860/860
WHARFIN$
```

2.2.10.12. Shutting Down a PPP or SLIP Interface

To bring down an interface, issue the following command:

```
$ IP SET /INTERFACE interface_name -
_$ /LINK_LEVEL=protocol /VMS_DEVICE=device /DOWN
```

2.2.10.13. Modifying Global Parameters

VSI TCP/IP maintains a set of global parameters that affect the behavior of all network interfaces. For example, your system's default route is specified as a global parameter and affects how all network interfaces direct data packets over the network.

To configure global parameters use the **SET *parameter_name*** command of NET-CONFIG. See the *VSI TCP/IP Administrator's Reference* for a complete list of **SET** commands. You can modify all global parameters without rebooting your system with the following exceptions:

- LOAD-UCX-DRIVER
- WINS-COMPATIBILITY

Changes to these parameters take effect after you reboot the system.

The following subsections describe how to modify global parameters to perform the following tasks:

- Configuring DECwindows support (see Section 2.2.10.14)
- Configuring the cluster alias feature, which permits another OpenVMScluster node to continue to provide some connectionless network services if the primary node that provides those services fails (see Section 2.2.10.15)
- Ensuring PATHWORKS 5.0 support is enabled (see Section 2.2.10.16)

2.2.10.14. Using the TCP/IP Transport Over UCX

You can configure the DECwindows server and X clients to use the TCP/IP Transport over UCX driver of VSI TCP/IP. VSI TCP/IP supports DECwindows over TCP/IP under VSI OpenVMS V8.4-2L1 by emulating the TCP/IP Services for the OpenVMS $QIO interface.

The UCX driver for VSI TCP/IP is enabled by default.

1. Edit your system startup command procedure to invoke VSI TCP/IP before starting DECwindows.
2. Reboot your system to start VSI TCP/IP with the UCX $QIO driver loaded.
3. To create a display, issue the following command:

```
$ SET DISPLAY/CREATE/NODE=ip-node-name /TRANSPORT=TCPIP
```

For complete information on the **SET DISPLAY** command and running remote DECwindows applications, see the *VMS DECwindows User's Guide*.

Each user must enable TCP/IP access on a host-by-host basis using the DECwindows Session Manager Customize Security menu so they can run applications over the TCP/IP transport.

From that menu, specify:

- TCP/IP as the Transport
- The remote host's Internet host name as the Node
- A question mark (?) for the Username

The DECwindows chapter of the *VSI TCP/IP User's Guide* contains information about running DECwindows applications over VSI TCP/IP.

2.2.10.15. Configuring OpenVMScluster Aliasing

If you have the cluster alias feature configured on more than one node in a OpenVMScluster, the nodes negotiate among themselves so that only one node at a time answers requests to the cluster alias. The cluster alias feature is also known as "cluster failover."

If the node serving the cluster alias fails and more than one node has the cluster alias configured, the service provided by the failed node is provided by another node in the cluster. Without the cluster alias feature, the services provided by the failed node are not available.

To use cluster aliasing, you provide a list of addresses to which each node will answer. If the node currently serving the cluster alias fails, one of the other nodes takes over the connectionless services for that address.

This feature lets you specify one or more nodes in the cluster as having an additional IP address, so only one of the nodes will use that additional IP address at any one time.

Enable cluster aliases with the NET-CONFIG **SET IP-CLUSTER-ALIASES** command. Specify the extra IP addresses for which this node should also answer requests. Disable cluster aliases by invoking the command without addresses. You can change the value of IP-CLUSTER-ALIASES without rebooting by also defining or redefining the system-wide logical name `IP$IP_CLUSTER_ALIASES` and restarting the IP$SERVER process.

Since this service is disabled by default, you must enable it in this way:

```
$ IP CONFIGURE /SERVER
SERVER-CONFIG>ENABLE CLUSTERALIAS
SERVER-CONFIG>EXIT
```

Now you can enable cluster failover as shown in the following example:

```
$ IP CONFIGURE
VSI TCP/IP for OpenVMS Network Configuration Utility 10.5(nnn)
[Reading in MAXIMUM configuration from IP$:IP.EXE]
[Reading in configuration from IP$:NETWORK_DEVICES.CONFIGURATION]
NET-CONFIG>SET IP-CLUSTER-ALIASES 192.1.1.2
```

```
NET-CONFIG>EXIT
[Writing configuration to IP$:NETWORK_DEVICES.CONFIGURATION]
[Writing Startup file IP$:IP$SYSTARTUP.COM]
[Changes take effect after the next OpenVMS reload]
$ DEFINE /SYSTEM /EXECUTIVE IP$IP_CLUSTER_ALIASES "192.1.1.2"
$ @IP$:START_SERVER
```

2.2.10.16. Ensuring PATHWORKS Support is Enabled

By default, VSI TCP/IP is configured to support PATHWORKS 5.0 running concurrently. To ensure this support is enabled:

1. Verify the presence of the PWIP (Pathworks over IP) device:

```
$ SHOW DEV PWIP
```

1. Invoke NET-CONFIG and enter the SHOW command. Check the "Load PWIP (Pathworks) driver:" line.

2. If VSI TCP/IP is not configured to load the PWIP driver, enter the **SET LOAD-PWIP-DRIVER** command.

3. Save the configuration to ensure it is loaded the next time your system reboots.

2.2.10.17. Enabling and Disabling MTU Discovery

Maximum Transmission Unit (MTU) discovery determines the maximum size of a TCP packet that can be sent through the network between two hosts. Performance improves when the largest, most efficient packet size possible with the hardware at each hop is enabled. RFC-1191 describes this feature, which is enabled by default.

When MTU discovery becomes active for a remote host, it places a host route in the routing table with the MTU set to the appropriate size. This feature is potentially useful for tracing unusual routes.

MTU discovery sets the Don't Fragment (DF) bit in IP packets. It is difficult to predict how routers from different vendors will handle the DF bit; some handle it correctly, some do not, some work until they need to fragment a packet, and some simply drop the packet. If you suspect a routing problem is affecting communications, disable MTU discovery by issuing the following command:

```
$ IP SET/KERNEL TCP_PMTU 0
```

To enable it again, issue this command:

```
$ IP SET/KERNEL TCP_PMTU 1
```

Both of these commands take effect immediately.

2.2.10.18. Manipulating the ARP Table

The Address Resolution Protocol (ARP) dynamically maps addresses between Internet and Ethernet. ARP is used by all VSI TCP/IP Ethernet interface drivers and Computer FDDI drivers.

ARP caches Internet-Ethernet address mappings. When an interface requests a mapping for an address not in the cache, ARP queues the message requiring the mapping and broadcasts an ARP request on the associated network requesting the address mapping.

If a response is provided, the new mapping is cached in the ARP table and any pending messages are transmitted. ARP queues no more than one packet while waiting for a mapping request to be responded to; only the most recently "transmitted" packet is kept.

To enable communications with systems that do not use ARP, the **IP SET /ARP** utility allows you to add and delete entries in the Internet-to-Ethernet tables.

Caution

Adding or modifying entries in the ARP table can seriously affect TCP/IP communications. Do not create or modify ARP table entries unless you are sure of their effects on your network.

The **SET /ARP** qualifiers are:

Qualifier	Description
/ADD	Adds a specified host-to-Ethernet address translation to the ARP tables
/DELETE	Deletes a specified host-to-Ethernet address translation from the ARP tables
/FLUSH	Flushes temporary entries in the ARP tables
/PERMANENT	Used with **/ADD** or **/FLUSH** to specify whether an entry is added or flushed permanently
/PROXY	Used with **/ADD** to indicate that the translation of the local host's Ethernet address should be published on behalf of another host
/PUBLISH	Used with **/ADD** to indicate that the translation to be added is to be published on behalf of another host
/TEMPORARY	Used with **/ADD** or **/FLUSH** to specify whether an entry should be added or flushed temporarily. This is the default.

2.2.10.19. GIF (generic/gateway) Interface Usage

The gif interface allows for the creation of Virtual Private Networks (VPNs) by encapsulating the traffic directed to the interface's remote address to within an additional IP header, creating a virtual network. If the traffic over this interface is subject to IPSEC, then the virtual network is private.

Each gif interface has four IP addresses that need to be configured:

1. The local address for the interface
2. The remote (point to point peer) address for the interface
3. The gateway address for this side of the tunnel
4. The destination address for the remote side of the tunnel.

The gif is configured with the following commands:

local system:

```
$ IP SET/INTERFACE/CREATE GIFn
$! n is unit number, compile time limited
$ IP SET/INTERFACE GIFn/PROTOCOL=IP/ADDRESS=A.B.C.D-
```

```
$_/POINT_TO_POINT=E.F.G.H
$ IP SET/ROUTE/ADD=(DESTINATION=A.B.C.D,GATEWAY=127.0.0.1)
$ IP SET/ROUTE/ADD=(DESTINATION=E.F.G.H,GATEWAY=A.B.C.D)
$ IP SET/INTERFACE GIFn/TUNNEL=(DESTINATION=I.J.K.L,-
$_ GATEWAY=M.N.O.P)
```

remote system:

```
$ IP SET/INTERFACE/CREATE GIFn
$! n is unit number, compile time limited
$ IP SET/INTERFACE GIFn/PROTOCOL=IP/ADDRESS=E.F.G.H-
$_/POINT_TO_POINT=A.B.C.D
$ IP SET/ROUTE/ADD=(DESTINATION=E.F.G.H,GATEWAY=127.0.0.1)
$ IP SET/ROUTE/ADD=(DESTINATION=A.B.C.D,GATEWAY=E.F.G.H)
$ IP SET/INTERFACE GIFn/TUNNEL=(DESTINATION=M.N.O.P,-
$_ GATEWAY=I.J.K.L)
```

M.N.O.P is a public IP address (interface) on the local system. I.J.K.L is a public IP address (interface) on the remote system. A.B.C.D is the private network address on the local system. E.F.G.H is the private network address on the remote system. Routing can be set up to pass traffic for other systems through the tunnel. A command procedure could be written to create the tunnel and be used on each side with some minor exchanging of parameters. IPSEC traffic could be statically configured, or managed with the RACOON IPSEC Daemon.

To get rid of the tunnel:

```
$ IP SET/INTERFACE/DELETE GIFn !delete tunnel and interface
$ IP SET/ROUTE/DELETE=(DESTINATION=A.B.C.D, GATEWAY=127.0.0.1)
```

The VPN encapsulates IPv4 traffic within another IPv4 packet (RFC 1853, RFC 2003).

This VPN is not compatible with Microsoft VPN, which uses either PPTP (Microsoft Proprietary) or L2TP/IPSec (RFC 2661).

2.3. Configuring Services

This section describes how to configure services for VSI TCP/IP, providing specific information about configuring the RLOGIN, RSHELL, NTY, TFTP, and SYSLOG services.

For a list of the servers you can configure with SERVER-CONFIG, see Appendix A.

2.4. Introducing Service Configuration

When configured, services of VSI TCP/IP start when a request for service is accepted by the master server (IP$SERVER) process, which listens for incoming connections. When a connection request is received, either a separate, detached process is created to handle the connection, or the IP$SERVER process handles the service internally.

Configuration information for the master server is in the configuration file `IP$:SERVICES.MASTER_SERVER` which is read by the IP$SERVER process when it starts to determine its configuration.

When VSI TCP/IP is first installed, the normal set of services is enabled. Before starting VSI TCP/IP, you may want to verify that the server configuration meets your needs. This may require:

- Disabling servers your site does not want accessed
- Enabling servers your site wants accessed
- Adding servers specific to your site

The server configuration supplied with VSI TCP/IP is adequate to get a system up and running. The server configuration can be easily changed as needed.

Generally, services are configured with one of the following configuration utilities:

- SERVER-CONFIG, invoked with the **IP CONFIGURE /SERVERS** command

This chapter describes how to use these utilities and provides information about other VSI TCP/IP servers.

2.4.1. Using SERVER-CONFIG to Configure Services

The VSI TCP/IP command line-based server configuration utility (SERVER-CONFIG) is an interactive utility that controls which network servers are available on the local node. In addition, you can use SERVER-CONFIG to set restrictions on a server to prevent access from unauthorized sites, to keep a log file of connections to a server, and to limit the system resources available to a server.

2.4.1.1. Invoking SERVER_CONFIG

To invoke SERVER-CONFIG, enter this command:

```
$ IP CONFIGURE /SERVERS
```

Exit this utility with the **EXIT** or **QUIT** command.

To display the list of servers available under VSI TCP/IP, run SERVER-CONFIG, and issue the **SHOW** command. You can use the **SHOW /FULL** command to display the characteristics of a particular server. To change the default configuration, enable a server with the **ENABLE** command and modify server configuration parameters with the **SELECT** and **SET** commands.

In the following example, SERVER-CONFIG enables the TFTP server (which is disabled by default) displays the TFTP server's configuration, and restarts the IP$SERVER process so the changes take effect immediately.

```
$ IP CONFIGURE /SERVERS
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>ENABLE TFTP
SERVER-CONFIG>SHOW TFTP/full
Service "TFTP":
  UDP socket (AF_INET,SOCK_DGRAM), Port 69
  INIT() = Merge_Image
  Program = "IP$:LOADABLE_TFTP.EXE"
SERVER-CONFIG>RESTART
Configuration modified, do you want to save it first?[YES]
[Writing configuration to IP_COMMON_ROOT:[IP]
SERVICES.MASTER_SERVER]
%RUN-S-PROC_ID, identification of created process is 20600046
SERVER-CONFIG>EXIT
[Configuration not modified, so no update needed]
```

2.4.1.2. SERVER-CONFIG Commands

Table 2.10 lists the SERVER-CONFIG commands.

Before using a **SET** command, use the **SELECT** command to select a service.

Table 2.10. SERVER-CONFIG Commands

Command	Description
ADD	Adds a service to the configuration
ATTACH	Detaches a terminal from calling process and attaches it to another process
COPY	Copies a service to create a new service
DELETE	Deletes a service from the current configuration
DISABLE	Disables a service in the current configuration
ENABLE	Enables a service in the current configuration
EXIT	Exits from the SERVER-CONFIG session
GET	Reads in a server configuration file
NETCONTROL	Contacts the NETCONTROL server
PUSH	Accesses the DCL command line while pausing SERVER-CONFIG
QUIT	Exits SERVER-CONFIG without saving changes
RESTART	Restarts the master server process
SAVE	Writes out the current server configuration file
SELECT	Selects a server for **SET** command
SET ACCEPT-HOSTS	Specifies hosts that can access the server
SET ACCEPT-NETS	Specifies networks that can access the server
SET BACKLOG	Specifies server connection queue limits
SET CONNECTED	Specifies a connection-request-received routine
SET DISABLED-NODES	Specifies OpenVMScluster nodes on which the service is disabled
SET ENABLED-NODES	Specifies OpenVMScluster nodes on which the service is enabled
SET FLAGS	Specifies a flag bit mask for service operation control
SET INIT	Specifies an initialize-service routine
SET KEEPALIVE-TIMERS	Specifies probes for cleaning up dormant connections
SET LISTEN	Specifies a listen-for-connections routine
SET LOG-ACCEPTS	Enables/disables successful connections logging
SET LOG-FILE	Specifies a log message destination
SET LOG-REJECTS	Enables/disables failed connections logging
SET MAX-SERVERS	Specifies the maximum number of processes
SET PARAMETERS	Specifies service-dependent parameters; affects the current configuration and becomes active the next time IP$SERVER starts

Command	Description
SET PQL-ASTLM [1]	Specifies the OpenVMS AST (asynchronous system trap) limit (the number of pending ASTs available to a process)
SET PQL-BIOLM	Specifies the OpenVMS buffered I/O limit (the number of outstanding buffered I/O requests available to a process)
SET PQL-BYTLM	Specifies the OpenVMS buffered I/O byte count limit (the number of bytes allowed in any single buffered I/O request)
SET PQL-CPULM	Specifies the OpenVMS CPU time limit of the created process
SET PQL-DIOLM	Specifies the OpenVMS direct I/O limit (the number of outstanding direct I/O requests available to a process)
SET PQL-ENQLM	Specifies the OpenVMS enqueue limit of the created process
SET PQL-FILLM	Specifies the OpenVMS open file limit (the number of open files available to a process)
SET PQL-JTQUOTA	Specifies the OpenVMS job-wide logical name table byte quota (the quota allocated to the job-wide logical name table on its creation)
SET PQL-PGFLQUOTA	Specifies the OpenVMS paging file quota of the created process
SET PQL-PRCLM	Specifies the OpenVMS sub-process limit (the number of sub-processes available to a process)
SET PQL-TQELM	Specifies the OpenVMS timer queue entry limit (the number of timer queue entries available to a process)
SET PRIORITY	Specifies the OpenVMS priority for created processes
SET PROGRAM	Specifies an OpenVMS file name for run or merged images
SET RECEIVE-BUFFER-SPACE	Specifies the size of receive socket buffer
SET REJECT-BY-DEFAULT	Specifies what happens if no match is found on **SET ACCEPT-HOSTS** and **SET REJECT-HOSTS**, and on **SET ACCEPT-NETS** and **SET REJECT-NETS**
SET REJECT-HOSTS	Specifies hosts not allowed service access
SET REJECT-MESSAGE	Specifies a rejected connection message
SET REJECT-NETS	Specifies networks not allowed service access
SET SEND-BUFFER-SPACE	Specifies the size of the send socket buffer
SET SERVICE	Specifies a perform-service routine
SET SERVICE-NAME	Changes a service name. The underscore character can be used in the service name.
SET SOCKET-FAMILY	Specifies service family address
SET SOCKET-OPTIONS	Specifies `setsockopt( )` options
SET SOCKET-PORT	Specifies the port for connection listening
SET SOCKET-TYPE	Specifies the socket type
SET USERNAME	Specifies the name of the user under which the service is started; applies only to UCX services. A UCX service has the UCX_SERVER flag set

Command	Description
SET WORKING-SET-EXTENT	Specifies the OpenVMS working set extent of the created process
SET WORKING-SET-QUOTA	Specifies the OpenVMS working set quota of the created process
SHOW	Shows the current server configuration
SHUTDOWN	Stops the master server process
SPAWN	Executes a DCL command, or mimics PUSH
STATUS	Shows the SERVER-CONFIG status
USE	Reads in a server configuration file
VERSION	Shows the SERVER-CONFIG version
WRITE	Writes out the current server configuration file

[1]PQL stands for process quota limits.

When you run SERVER-CONFIG commands, a number of prompts are displayed. These prompts are explained in Appendix A. There are a number of per-service prompts when you modify server parameters. Some of these parameters control the operation of the server, including process priority, working set limit, restrictions, and auditing. Other parameters define operations of the server, such as the protocol and port number.

2.4.2. Adding Your Own Services

The IP$SERVER process can listen for user-written services (including IPX/SPX) and, when a connection request arrives, create an OpenVMS detached process running the user-written program. This process is created with full privileges. SYS$INPUT, SYS$OUTPUT, and SYS$ERROR are set to the network. See the *VSI TCP/IP Programmer's Reference* for descriptions of library routines for writing your own services that interface to VSI TCP/IP.

The following example shows how to add a user-written service called NNTP to the VSI TCP/IP configuration. The program NNTP_SERVER.EXE is invoked when an NNTP connection arrives.

```
SERVER-CONFIG>ADD NNTP
[Adding new configuration entry for service "NNTP"]
Protocol: [TCP] TCP
TCP Port number: 119
Program to run: USER$DISK:[NNTP]NNTP_SERVER.EXE
[Added service NNTP to configuration]
[Selected service is now NNTP]
SERVER-CONFIG>
```

Note

If your service uses the **getservbyname( )** or **getservbyport( )** socket library functions, you must also add your service to the HOSTS.LOCAL file and recompile your host tables.

2.4.3. Disabling, Enabling, and Deleting Services

You can tailor the IP$SERVER to meet your specific needs by enabling or disabling services using the **ENABLE**, **DISABLE**, or **DELETE** commands.

For example, to enable BOOTP service with the **SERVER-CONFIG ENABLE BOOTP** command and restart the server to make the change immediately available, issue these commands:

```
$ IP CONFIGURE /SERVER
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>ENABLE BOOTP
SERVER-CONFIG>RESTART
Configuration modified, do you want to save it first ? [YES]
[Writing configuration to IP_COMMON_ROOT:[IP]
SERVICES.MASTER_SERVER]
%RUN-S-PROC_ID, identification of created process is 20600046
SERVER-CONFIG>EXIT
[Configuration not modified, so no update needed]
```

The following example shows how to disable the SMTP service:

```
SERVER-CONFIG>DISABLE SMTP
```

2.4.3.1. Disabling or Enabling Services on a Per-Cluster-Node Basis

You can enable or disable services on a per-node basis in a OpenVMScluster. Using the **SET ENABLED-NODES** or **SET DISABLED-NODES** commands, you can specify a list of OpenVMScluster nodes on which the service does or does not run. You must also enable the service using the **ENABLE** command. For example, to enable the SMTP service to run only on the OpenVMScluster node VMSA:

```
SERVER-CONFIG>SELECT SMTP
[The Selected SERVER entry is now SMTP]
SERVER-CONFIG>SET ENABLED-NODES
You can now add new OpenVMScluster nodes for SMTP.
An empty line terminates.
Add OpenVMScluster node: VMSA
Add OpenVMScluster node:
SERVER-CONFIG>
```

2.4.4. Restricting Access to Servers

VSI TCP/IP allows a system manager to restrict access to services on a per-service, per-network, or per-host basis.

Note

Restriction lists are supported only for services that listen for connections through IP$SERVER. For example, the NFS Server, which reads datagrams directly, ignores any restrictions configured through SERVER-CONFIG.

Five service parameters (ACCEPT-HOSTS, ACCEPT-NETS, REJECT-HOSTS, REJECT-NETS, and REJECT-BY-DEFAULT) control whether the IP$SERVER process allows or rejects a connection request based on the requesting host address, network, or subnetwork.

- If the connection comes from a host listed in the ACCEPT-HOSTS or REJECT-HOSTS lists, the connection is accepted or rejected, respectively.
- If the host is not found in one of these lists, the ACCEPT-NETS and REJECT-NETS lists are examined.

- If a match is found, the connection is accepted or rejected.
- If the host does not match any of these four lists, the action taken is governed by the REJECT-BY-DEFAULT parameter, which is normally set to FALSE, indicating that all connections are accepted.

You can also use the ACCEPT-NETS and REJECT-NETS parameters to specify a subnetwork number. You specify a subnetwork by supplying the subnetwork number followed by a space and the subnetwork mask for that subnetwork. For example: `REJECT-NETS 128.1.1.0 255.255.255.0` rejects only the hosts on the 128.1.1.n network. All other hosts on 128.1.n.n have access.

Note

If you answer YES after the Internet address at the prompt, only connections from a port number below 1024 are accepted.

The action taken to reject a connection depends on the protocol involved; for example, a UDP datagram is rejected by ignoring it, but a TCP connection is rejected by immediately closing the connection.

The REJECT-MESSAGE parameter specifies a text string sent to the client before the connection is closed. The following example shows how to restrict access to the TELNET server to hosts that are on a local network at address 128.1.0.0, with the one exception of the host at the address 192.0.0.2.

```
$ IP CONFIGURE /SERVERS
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>SELECT TELNET
[The Selected SERVER entry is now TELNET]
SERVER-CONFIG>SET ACCEPT-NETS
You can now add new addresses for TELNET. An empty line terminates.
Add Address: 128.1.0.0
Add Address:
SERVER-CONFIG>SET ACCEPT-HOSTS
You can now add new addresses for TELNET. An empty line terminates.
Add Address: 192.0.0.2
Add Address:
SERVER-CONFIG>SET REJECT-BY-DEFAULT TRUE
SERVER-CONFIG>SET REJECT-MESSAGE Illegal source of TELNET connection
SERVER-CONFIG>SHOW /FULL
Service "TELNET":
   TCP socket (AF_INET,SOCK_STREAM), Port 23
   Socket Options = SO_KEEPALIVE
   INIT() = TCP_Init
   LISTEN() = TCP_Listen
   CONNECTED() = TCP_Connected
   SERVICE() = Internal_Telnet
   Accept Hosts = IP-192.0.0.2
   Accept Nets = IP-128.1.0.0
   Reject by default all other hosts and nets
   Reject Message = "Illegal source of TELNET connection"
SERVER-CONFIG>RESTART
Configuration modified, do you want to save it first ? [YES] YES
[Writing configuration to
IP_COMMON_ROOT:[IP]SERVICES.MASTER_SERVER]
```

```
%RUN-S-PROC_ID, identification of created process is 20600054
SERVER-CONFIG>
```

2.4.5. Auditing Access to Servers

VSI TCP/IP allows the security-conscious system manager to audit access to a service, file, or the OpenVMS process. Three service parameters govern the auditing that occurs when a connection is accepted or rejected:

LOG-ACCEPTS	Enables or disables logging for accepted connection requests.
LOG-FILE	Specifies the OpenVMS file name to which log messages are written. The auditing data collected by the VSI TCP/IP Server is collected in this file and flushed (written to disk) approximately every five to ten minutes. To maintain different versions of this file, you can copy an empty file over the old one on a daily basis (or at a frequency that meets your needs).
LOG-REJECTS	Enables or disables logging for rejected connection requests. A request can be rejected if the REJECT-HOSTS, REJECT-NETS, or REJECT-BY-DEFAULT security restrictions are enabled and succeed, or if the ACCEPT-HOSTS and ACCEPT-NETS restrictions are enabled and fail. For more information, refer to Section 2.4.4.

Note

The only services that support auditing are those that listen for connections through the IP$SERVER. For example, the optional NFS server, which reads datagrams directly, ignores the auditing parameters. You should not turn on logging for a UDP service. Because there is no "formal" connection for UDP, every packet sent to the UDP service would be logged.

The following example shows how to enable a log file on the TELNET service.

```
$ IP CONFIGURE /SERVERS
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>SELECT TELNET
[The Selected SERVER entry is now TELNET]
SERVER-CONFIG>SET LOG-ACCEPTS TRUE
SERVER-CONFIG>SET LOG-REJECTS TRUE
SERVER-CONFIG>SET LOG-FILE IP$:SERVER.LOG
SERVER-CONFIG>SHOW/FULL
Service "TELNET":
TCP socket (AF_INET,SOCK_STREAM), Port 23
Socket Options = SO_KEEPALIVE
INIT() = TCP_Init
LISTEN() = TCP_Listen
CONNECTED() = TCP_Connected
SERVICE() = Internal_Telnet
Log File for Accepts & Rejects = IP$:SERVER.LOG
SERVER-CONFIG>RESTART
Configuration modified, do you want to save it first ? [YES] YES
[Writing configuration to IP_COMMON_ROOT:[IP]
SERVICES.MASTER_SERVER]
%RUN-S-PROC_ID, identification of created process is 20600054
SERVER-CONFIG>
```

The next example shows the auditing records written to the log file.

```
15-JUN-2017 17:27:00 RPCPORTMAP (accepted) from [127.0.0.1,108] (localhost)
15-JUN-2017 17:50:25 FINGER (accepted) from [192.41.228.65,1071] (ABC.COM)
15-JUN-2017 21:10:40 RLOGIN (accepted) from [192.41.228.65,1022] (ABC.COM)
16-JUN-2017 11:49:46 FINGER (accepted) from [192.41.228.65,1214] (ABC.COM)
16-JUN-2017 11:51:05 RLOGIN (accepted) from [192.41.228.68,1022] (Bubba)
16-JUN-2017 20:09:48 FTP (accepted) from [192.41.228.68,1039] (Bubba)
```

2.4.5.1. Writing an Auditing Dispatcher

VSI TCP/IP allows a system manager to further customize the auditing facilities.

You can provide a user-written shareable image that is merged into the IP$SERVER when the server is restarted. The user-written shareable image is called whenever a connection arrives with information about the connection and can perform the desired auditing.

2.4.6. Detecting Intruders

VSI TCP/IP provides the IP address and an optional IP port number of a suspected intruder to OpenVMS accounting and intrusion detection. The IP address is recorded in hexadecimal format to make room for the optional IP port address. By default, the IP port address is included. To disable this feature, set the LGI_BRK_TERM system parameter to zero.

This example shows accounting and intrusion reports generated with LGI_BRK_TERM set to one (the default).

```
$ ACCOUNTING/NODE=TELNET
Remote node name:   TELNET          Privilege <31-00>: 00148000
Remote ID:          A12C800C:0F94   Privilege <63-32>: FFFFFFC0
                    ^^^^^^^^ ^^^^
                    IP address IP port
$ SHOW INTRUSION
Intrusion   Type      Count   Expiration    Source
NETWORK     SUSPECT   2       17:35:09.28   TELNET::A12C800C:0F94
                                            ^^^^^^^^^^^^^^^^^^^^
```

In these examples, A12C800C is the IP address 191.87.34.22 and 0F94 is IP port 3988, both expressed in hexadecimal. You can use the following DCL code to convert hexadecimal IP addresses and port numbers (of the form *hex-address:hex-port-number*) into dotted decimal format:

```
$ IP_ADDRESS = F$ELEMENT(0,":",P1)
$ PORT_NUMBER = F$ELEMENT(1,":",P1)
$ WRITE SYS$OUTPUT F$FAO("IP address/port: !UL.!UL.!UL.!UL/!UL", -
F$INTEGER("%X''F$EXTRACT(0,2,IP_ADDRESS)'"), -
F$INTEGER("%X''F$EXTRACT(2,2,IP_ADDRESS)'"), -
F$INTEGER("%X''F$EXTRACT(4,2,IP_ADDRESS)'"), -
F$INTEGER("%X''F$EXTRACT(6,2,IP_ADDRESS)'"), -
F$INTEGER("%X''PORT_NUMBER'"))
$ EXIT
```

Intrusion detection uses the physical (Ethernet) address, SPX port number, and possibly the target user name. The target user name appears only when the break-in attempt targets a valid account and LGI_BRK_TERM is set to one (the default), as in the following example intrusion report.

```
$ SHOW INTRUSION
Intrusion   Type      Count   Expiration    Source
```

```
TERM_USER   SUSPECT   1       16:23:38.67
 [A12C8000:AA.00.04.00.08.60#ECB]:
                                                    ^^^^^^^^ ^^^^^^^^^^^^^^^^ ^^^
                                                    Network  Ethernet address  SPX
 User
                                                    number
 port Name
```

In this example,

- A12C8000 is the network number, expressed in hexadecimal.
- AA.00.04.00.08.60 is the Ethernet address, expressed in dotted-hexadecimal format.
- ECB is the SPX port.
- No user name is specified; the user name normally follows the ending colon.

The OpenVMS SHOW INTRUSION Utility truncates the Source display field to 33 characters. The TERM_USER record is actually associated with a particular user name; you can delete it (with **DELETE /INTRUSION**) only by specifying the full source specification, including the invisible, truncated data.

2.4.6.1. Detecting Intruders on an FTP Server

The following example of an `FTP_SERVER.COM` file shows the use of the FTP server qualifiers. The first section ensures that global and local symbols appear as expected. Information is then taken from the logical names that store data about the person who has accessed this FTP server:

- `IP$FTP_ADDRESS`: Provides an ASCII representation of the user's IP address;
- `IP$FTP_HOSTNAME`: Provides an ASCII representation of the user's host name;
- `IP$ANONYMOUS_PASSWORD`: Contains an ASCII representation of the user's anonymous FTP password;
- `IP$FTP_MAXIMUM_IDLE_TIME`: Controls the duration (in seconds) the FTP server allows a connection to be idle before it is closed. The default is 300 seconds. This logical can be specified in any logical name table accessible to the user running the FTP server.

```
$ Set := "Set"
$ Set Symbol/Scope=(NoGlobal,NoLocal)
$ If F$TrnLNM("IP$FTP_ADDRESS") .Eqs. "" Then -
VSI TCP/IP for OpenVMS FTP/Server/Get_Remote_Info
$ FTP_Address = F$TrnLNM("IP$FTP_ADDRESS")
$ FTP_Hostname = F$TrnLNM("IP$FTP_HOSTNAME")
$ FTP_Password = F$TrnLNM("IP$ANONYMOUS_PASSWORD")
$ Ident = FTP_Hostname
$ If FTP_Hostname .Eqs. "" Then Ident = FTP_Address
```

In the next section, if FTP_Hostname is null, the IP address could not be found in the host table or by DNS lookup. The Ident symbol is set to flag this event:

```
$ Message = ""
$ If FTP_Hostname .Eqs. "" Then -
  Message = Message + ","Unknown hostname: ''Ident'; DNS Problem!"""
$ If F$Edit(FTP_Password,"UPCASE") .Eqs. "GUEST" Then -
```

```
  Message = Message + ","""''Ident'; say who really you are."""
$ Message = Message + ","""@WELCOME.TXT"""
$ If Message .Nes. "" Then -
  Message = "/Message=("+F$Extract(1,256,Message)+")"
```

In the following section, the banner message is altered to fit the circumstances under which the user logs in.

```
$ If F$Extract(0,6,FTP_Address) .Eqs. "192.0." Then Goto Intruder
$ DirOptions = "/Directory=Users:[Anonymous]"
$ AccOptions = "/Access=NoWrite"
$ IP FTP/Server 'AccOptions' 'DirOptions' 'Message'
$ Logout/Brief
```

The following section tests the IP address for a known intruder. If the intruder is discovered, control is passed to the Intruder label. Next, the FTP server is called to handle the anonymous login, and when done, logs out.

1. An offensive stance is taken when an intruder is discovered by running a FINGER of the intruder's system.
2. `INFLAME.TXT` is invoked to display a personalized message.
3. The FINGER output is displayed on the intruder's terminal.
4. Mail is sent to the system manager to indicate that an intrusion event occurred.
5. The procedure exits.

```
$ Intruder:
$ Set NoOn
$ create finger.temp
$ define/user sys$output finger.temp
$ IP Finger @'FTP_Address
$ IP FTP/Server/Reject/Message=("@Inflame.txt",-
"Thanks for listening ''FTP_Password'@''Ident';
  now smile:","","@finger.temp")
$ Mail/Subject="intruder FTP access from
  ''FTP_Password'@''Ident'" finger.temp system
$ del finger.temp;*
$ logout/Brief
```

2.4.6.2. Detecting Intruders with NETCONTROL Accounting

If OpenVMS Accounting is enabled on your system, the following process header fields are set for network server processes created by the VSI TCP/IP master server:

- `CTL$T_NODENAME` is the name of the service being run; for example, FTP, TELNET, or SMTP.
- `CTL$T_REMOTEID` is the IP address of the remote client system in dotted-decimal format.

This change can make attempted break-ins to your system easier to track. It also provides a simple mechanism for tracking remote access to your system.

For example, to determine which users are using TELNET to gain access to your system, you can issue the command:

```
$ ACCOUNTING /NODE=TELNET
```

The TELNET service complies with RFC-779 and provides location information.

Note

IP address and port information is displayed in hexadecimal in accounting reports.

The IP NETCONTROL command features ACCOUNTING and DEBUG parameters for the NETCONTROL (master server) service.

The format of this command is:

```
$ IP NETCONTROL NETCONTROL ACCOUNTING n
```

By default (as described above), additional information is provided in the accounting record by the VSI TCP/IP server. You can disable this feature by setting *n* to 0. When set to 1, the remote name and service name are added to the ACCOUNTING record.

OpenVMS adds accounting records to a central database each time a special system event occurs. These events include processes exiting, images (programs) ending, failed logins, and so on. OpenVMS stores a record of the event with information from the process.

One item that OpenVMS includes in the accounting record is the remote node ID, which it gets from the internals of the process. For most processes, the remote node ID is empty.

When you use **SET HOST** to create a log for another node, DECnet fills in the remote node ID field when it creates the process. This information then appears in any accounting record generated for that process.

If you suspect an intruder has attempted a security breach of your system, you can examine the accounting records to see who has logged in and identify where a login attempt originated.

The master server fills in both the remote node ID and the node name field. The remote node ID field is set to the ASCII representation of the dotted-decimal IP address of the node that requested the service. The NODE NAME field is set to the service name, such as FTP, TELNET, and so on.

The following example shows a LOGIN failure accounting record:

```
$ LOGIN FAILURE
-------------
Username:      MILLER   UIC:          [SYSTEM]
Account:       <net>    Finish time:  23-JUN-2017 21:27:34.47
Process ID:    21200124 Start time:   23-JUN-2017 21:27:33.81
Owner ID:      Elapsed time:   0 00:00:00.66
Terminal name: Processor time: 0 00:00:00.26
Remote node addr: Priority:    4
Remote node name: FTP      Privilege <31-00>: FFFFFFFF
Remote ID:     161.44.128.94Privilege <63-32>: FFFFFFFF
Remote full name:
Queue entry:   Final status code: 00D380FC
Queue name:
Job name:
Final status text:%LOGIN-F-INVPWD, invalid password
Page faults:        166   Direct IO:         7
Page fault reads:   5     Buffered IO:       12
Peak working set:   246   Volumes mounted:   0
Peak page file:     3280  Images executed:   1
```

2.4.7. Using UCX-Compatible Services under VSI TCP/IP

Services written to be started by the auxiliary server (INETD) in TCP/IP Services can be configured for use with VSI TCP/IP by setting the service parameter **SET FLAGS UCX_SERVER**.

Driver enhancements support TeamLinks, POSIX sockets, DCE, and TCP/IP Services V2.0 keepalive compatibility.

The logical names `UCX$INET_HOSTADDR` and `TCPIP$INET_HOSTADDR` contain the text value of the primary interface. VSI TCP/IP defines `UCX$INET_HOSTADDR` and `TCPIP$INET_HOSTADDR` automatically, much the same as other TCP/IP Services logical names. It is defined in `IP$SYSTARTUP.COM`.

Performing a close (dassgn) operation on any TCP/IP Services (BG) device used in a select list cancels the select operation.

The following logicals are defined for improved UCX compatibility:

`TCPIP$BIND_DOMAIN`

`TCPIP$BIND_SERVER000`

`TCPIP$BIND_SERVER001` (if more than one BIND server is defined)

`TCPIP$BIND_SERVER002` (if more than one BIND server is defined)

`TCPIP$DEVICE = BG:`

`TCPIP$INET_DEVICE = BG:`

`TCPIP$INET_HOST`

`TCPIP$INET_HOSTADDR`

`TCPIP$IPC_SHR = IP$:UCX$IPC_SHR`

2.4.8. Associating Command Procedures with Services

You can specify DCL command procedures (`.COM` files) as the programs associated with VSI TCP/IP services. When a service is initiated, VSI TCP/IP calls LOGINOUT to invoke the user's `LOGIN.COM` file and the specified DCL command procedure. When called, the DCL and CLI are mapped for use by the process. This feature gives the system manager a "hook" into a service with an easy-to-create command procedure. Note, however, that the command procedure cannot use `SYS$INPUT`. Therefore, do not use the **READ SYS$INPUT** or **INQUIRE** commands in the command procedure or a user's `LOGIN.COM` file.

Make sure that all command procedures associated with services include a command that assigns `SYS$INPUT` to `SYS$OUTPUT` as follows:

```
$ DEFINE /USER SYS$INPUT SYS$OUTUT
```

This command must appear immediately before any command that runs an image, but is only in effect while the image is running. To ensure the assignment lasts for the duration of the entire command procedure, use the above command without the **/USER** qualifier:

```
$ DEFINE SYS$INPUT SYS$OUTPUT
```

2.4.9. Setting Keepalive Timers

Keepalives are useful when other systems that connect to services provided by your system are subject to frequent crashing, resets, or power-offs (as with personal computers). TCP/IP connections must normally pass through a three-way handshake sequence to be closed and removed from the connection table. If a connection is open but idle, and the remote system is shut down, reset, or crashes, the connection is not closed down until your system attempts to communicate with the remote system. If an application or service does not attempt to communicate, a keepalive probe can clean up these dormant connections.

The format for the **SET KEEPALIVE-TIMERS** function is:

```
SERVER-CONFIG>SELECT service
```

```
SERVER-CONFIG>SET KEEPALIVE-TIMERS idle-time probe-interval probe-count
```

where:

- *idle-time* is the amount of time, in seconds, that a connection should be idle before the first keepalive probe is sent.
- *probe-interval* is the number of seconds between keepalive probes (75 seconds minimum).
- *probe-count* is the number of probes sent, with no reply from the other side of the connection, before the connection is destroyed.

Setting any of these parameters to 0 retains its default setting.

If you set the SO_KEEPALIVE socket option for a service, but you do not explicitly set KEEPALIVE-TIMERS, the default values are:

- *idle-time* is 2 hours.
- *probe-interval* is 75 seconds.
- *probe-count* is 8.

If you do not set the SO_KEEPALIVE socket option for a service, no keepalive probes are sent for connections to that service.

2.4.10. Configuring TFTP (Trivial File Transfer Protocol)

The VSI TCP/IP TFTP service uses standard Internet TFTP to perform file transfers. Like FTP, TFTP can also be used to transfer files between a host running OpenVMS and a remote host. Unlike FTP, TFTP cannot perform operations other than transferring files between a local system and a remote one; that is, TFTP cannot list directories, delete files, and so on.

TFTP does not perform any authentication when transferring files, so a user name and password on the remote host are not required. In general, only files with world-read (W:R) access in certain directories on the remote host are available for reading, and only certain directories are available for writing.

Use SERVER-CONFIG to enable TFTP as follows:

```
$ IP CONFIGURE /SERVER
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>ENABLE TFTP
SERVER-CONFIG>EXIT
[Writing configuration to IP_COMMON_ROOT:[IP]|SERVICES.MASTER_SERVER]
```

Note

The permissions of the accessed directories are not checked before the access is attempted. Because the TFTP protocol does not specify any user login or validation, the TFTP server permits only WORLD-readable files to be accessed.

The TFTP server normally requires full file system pathnames, as it operates without a default directory. If you are using TFTP to download network servers which may not be able to provide full pathnames, you can set a TFTP default directory using the NET-CONFIG **SET TFTP-DIRECTORY** command.

Note

The TFTP mail option, as defined in RFC-783, is obsolete and not supported under the VSI TCP/IP TFTP server. The TFTP server supports the BLKSIZE option (RFC 2348) with a minimum of 512 and maximum of 8192. For image (octet) transfers the size must be a multiple of 512.

2.4.10.1. TFTP File Name Translations

The `IP$:TFTP.FILENAME-TRANSLATIONS` file can be used to translate between TFTP client file names and OpenVMS file names, and restrict TFTP access to certain files or directories. The following is an example of a `IP$:TFTP.FILENAME-TRANSLATIONS` file:

```
TFTP filename fixups for broken TFTP
# loaders & file access restrictions
RESTRICT-ACCESS
#This is a translation for a single file
/Foo/bar.dat                 SYS$MANAGER:BAR.DAT
#These two translations map an entire OpenVMS directory
/usr/lib/X11/ncd/configs/*   NCD_CONFIGS:[000000]
/ncd_fonts/dw75dpi/*         NCD_FONTS:[DW75DPI]
#This translation is a file access restriction
SYS$SYSROOT:[SYSFONT.DECW.*  SYS$SYSROOT:[SYSFONT.DECW.
```

Use either "#" or "!" to start a comment.

If the keyword RESTRICT-ACCESS is on a line by itself in the file, only files that match one of the translation specifications are accessible. This restricts TFTP access to specific directory hierarchies. If this string is not specified in the `FTP.FILENAME-TRANSLATIONS` file, all WORLD-readable files are accessible to TFTP clients.

The first translation causes a client reference to /Foo/bar.dat to access the OpenVMS file `SYS$MANAGER:BAR.DAT` (note that comparisons are not case-sensitive). This is an example of a translation for a single file.

The next two translations are examples of mapping an entire OpenVMS directory. If the client reference matches everything up to the asterisk (*), the rest of the client reference is appended to

the translation string. For example, /usr/lib/X11/ncd/configs/foo.dat becomes `NCD_CONFIGS:[000000]FOO.DAT`.

The last translation is an example of a file access restriction. The result of the translation is the same as the client-specified file. The TFTP server disallows subdirectory specifications that include ".-" to ensure you cannot bypass access restrictions by going back up the OpenVMS directory hierarchy.

If the file `IP$:TFTP.FILENAME-TRANSLATIONS` is edited, NETCONTROL must be used to RELOAD it:

```
$ IP NETCONTROL TFTP RELOAD
```

2.4.10.2. Configuring "R" Services

This section describes configuration of the "R" services, RLOGIN and RSHELL.

- RLOGIN provides a means of logging in to another system.
- RSHELL executes commands remotely on another system.

These services are enabled when VSI TCP/IP is installed.

The authentication scheme used by RLOGIN and RSHELL is based on trusted users and trusted hosts specified in files on the destination system. The `IP$:HOSTS.EQUIV` file (/etc/hosts.equiv on UNIX systems) grants access on a system-wide basis. The file SYS$LOGIN:.RHOSTS (~/.rhosts on UNIX systems) can be used by individual users on a system to grant remote users access to their accounts.

Note

Do not use IP addresses in the `HOSTS.EQUIV` or `.RHOSTS` file.

Access control requirements differ between RLOGIN and other "R" services. RLOGIN requires both NETWORK and LOCAL access, while RSHELL, REXEC, RMT, and RCP only require NETWORK access.

If you remove a user's NETWORK access, the user can still log in until their RLOGIN cache entry expires or is flushed. However, if you remove that user's LOCAL access, the user is denied access immediately, even if they have a current cache entry.

The format of an entry in the `HOSTS.EQUIV` or `.RHOSTS` file is:

`hostname` [*`username`*]

If an entry containing only the host name is in the file `IP$:HOSTS.EQUIV` (or /etc/hosts.equiv) on the target system, all users on *`hostname`* with the same user name as on the target system can access the target without specifying a user name or password.

Both `IP$:HOSTS.EQUIV` and `SYS$LOGIN:.RHOSTS` accept "wildcards" in host names. For example, to specify all hosts at `FLOWERS.COM`, include the following line:

```
*.FLOWERS.COM
```

If an entry in `SYS$LOGIN:.RHOSTS` or a user's ~/.rhosts file contains the following format, the specified *`username`* on the *`hostname`* system can access the user's account on the target system without specifying a password (or a user name if the user names are identical on the two systems):

`hostname username`

The next example shows a HOSTS.EQUIV file on the host SALES.FLOWERS.COM that gives users on SALES.FLOWERS.COM RLOGIN and RSHELL access to their own accounts on the system (this is allowed by the first two entries).

In this example, FLOWERS.COM and BUBBA.FLOWERS.COM are identified (in the last two entries) as trusted hosts, allowing any user on either of these systems to have RLOGIN and RSHELL access to the account of the same name on SALES.FLOWERS.COM without specifying a user name or password.

```
localhost
sales.flowers.com
flowers.com
bubba.flowers.com
```

This example shows a .RHOSTS file that belongs to a user on SALES.FLOWERS.COM:

```
flowers.com   system
unix.flowers.com     root
```

The first entry grants access to the user's account on the host SALES.FLOWERS.COM to user SYSTEM on host FLOWERS.COM. The second entry grants access to the user's account to user "root" on host UNIX.FLOWERS.COM. Hence, either of these two remote users can use RLOGIN or RSHELL to access the user's account on SALES.FLOWERS.COM without specifying a password.

Note

When specifying a user in any of the authentication files (particularly on the UNIX operating system), make sure to specify the user name in the correct case. "ROOT" and "root" are treated as different user names under case-sensitive systems such as the UNIX Operating System.

The host initiating the RLOGIN or RSHELL request must be listed in the destination host's hostname database or must be resolvable within the Domain Name System (DNS), if domain name service is enabled. If the destination host cannot determine the initiating host's name from the IP address in the connection request, it rejects the request.

The RLOGIN server parameters INCLUDE-AUTHENTICATION-INDICATION, INCLUDE-PORT-NUMBER and INCLUDE-SEND-LOCATION cause an authentication indication, a port number, or a send location (or all three) to be placed in the TT_ACCPORTNAM field of the NTY device control block. These parameters are disabled by default. To enable them, use SERVER-CONFIG (**IP CONFIGURE /SERVER**).

2.4.10.3. Disabling the Standard Error RSHELL Connection

The **RSHELL /NOSTDERR** option disables creation of the connection from the remote RSHELL server back to the client for "standard error" output. This allows you to use the RSHELL command through firewalls that do not allow TCP connections that originate from outside the local network. When you use this option, the remote RSHELL server sends messages to the "standard output" connection, so error messages are still displayed.

2.4.10.4. RLOGIN and RSHELL Authentication Cache

The VSI TCP/IP RLOGIN and RSHELL servers cache the contents of the .RHOSTS and HOSTS.EQUIV files and authentication information from the SYSUAF file in memory for ten minutes to improve performance. This means that changes made to these files may not be noticed by the network immediately. Use the following command to flush the cache before the timeout period:

```
$ IP NETCONTROL RLOGIN FLUSH
```

You can use SERVER-CONFIG to change the timeout on these caches by setting the RHOSTS-TIMEOUT or UAF-TIMEOUT parameters for RSHELL, RLOGIN, or REXEC. The default value for these parameters is 600 seconds. Setting a value of 0 (zero) disables the cache. Because the "R" services share a common authentication cache, you need only set these parameters for one service. If you set different values for different servers, only one of the values is used, so set these parameters on only one server. The following example shows how to set the timeout parameter.

```
$ IP CONFIGURE/SERVER
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>SELECT RLOGIN
[The Selected SERVER entry is now RLOGIN]
SERVER-CONFIG>SET PARAMETERS
Delete parameter "rhosts-timeout 0" ? [NO] YES
[Parameter "rhosts-timeout 0" deleted from RLOGIN]
You can now add new parameters for RLOGIN. An empty line terminates.
Add Parameter: RHOSTS-TIMEOUT 3600
Add Parameter: UAF-TIMEOUT 60
Add Parameter:
[Service specific parameters for RLOGIN changed]
SERVER-CONFIG>
```

The UAF access type specification for RSHELL access is NETWORK. VSI TCP/IP checks all access times.

The **IP NETCONTROL RLOGIN SHOW** command shows the time the information read in from `specific.RHOSTS` files will expire. This is the time remaining until the `.RHOSTS` file is read in again.

Note

A non-existent `.RHOSTS` file is treated the same as an empty `.RHOSTS` file.

2.4.11. Controlling RSHELL and REXEC Process Deletion

If a client closes a connection before the remote process finishes, the RSHELL and REXEC servers may delete the process. This behavior affects PC-based X servers that use REXEC to launch X applications.

This default action can be changed by using SERVER-CONFIG to **SET PARAMETER** as in the following example:

```
$ IP CONFIGURE /SERVER
. . . startup messages . . .
SERVER-CONFIG>select rshell
[The Selected SERVER entry is now RSHELL]
SERVER-CONFIG>set parameter
You can now add new parameters for RSHELL. An empty line terminates.
Add Parameter: ? parameter, one of the following:
DEBUG                     DISALLOW-RHOSTS   DISALLOW-X-DISPLAY
PREVENT-PROCESS-DELETION   RHOSTS-TIMEOUT    UAF-TIMEOUT
or confirm with carriage return
```

```
Add Parameter: PREVENT-PROCESS-DELETION
Add Parameter:
[Service specific parameters for RSHELL changed]
SERVER-CONFIG>
```

2.4.11.1. Controlling Automatic WSA Device Creation

By default, VSI TCP/IP "R" services create WSA devices and set displays to simplify setup for X client users, allowing users to run X clients without explicitly issuing the **SET DISPLAY** command. To disable this feature, set DISALLOW-X-DISPLAY with the **SET PARAMETER** command (in SERVER-CONFIG).

2.4.11.2. Inhibiting Output in Command Procedures for "R" Services

Problems arise when remote users log into systems using a login command procedure (`SYS$LOGIN:SYLOGIN.COM` or `SYS$MANAGER:SYLOGIN.COM`) that requires screen output. To inhibit this behavior, make sure the following lines are included at the top of all login command procedures:

```
$ VERIFY = 'F$VERIFY(0)                 ! Turn off verify without echoing
$ IF F$MODE() .EQS. "OTHER" THEN EXIT   ! If a DETACHED process (RSHELL)
$ IF VERIFY THEN SET VERIFY             ! If a batch job, may want to turn
                                        ! verify back on.
```

2.4.11.3. Permitting "R" Service Access to Captive or Restricted Accounts

In general, "R" services should not be permitted access to captive or restricted OpenVMS accounts. However, if your users depend on such access, define the following logical to allow access to these types of accounts:

```
$ DEFINE/SYSTEM/EXEC IP_RSHELL_ALLOW_CAPTIVE "TRUE"
```

2.4.11.4. Configuring the TELNET Server for Kerberos V5

Enable the Kerberos V5 functionality with the following commands:

```
$ IP CONFIGURE /SERVER... startup messages...
SERVER-CONFIG>SELECT TELNET
(The Selected SERVER entry is now TELNET)
SERVER-CONFIG>SET PROGRAM IP$:LOADABLE_KTELNET_CONTROL
(Program to run for TELNET set to IP$:LOADABLE_KTELNET_CONTROL)
SERVER-CONFIG>SET INIT Merge_Image
(Init action of TELNET set to Merge_Image)
```

After the values are saved and the Master Server is restarted, Kerberos 5 functionality is available.

The authentication behavior on the TELNET Server is determined by the system logical `IP$AUTH_TELNET`. It has 3 possible values: ALLOWED, REQUIRED, and DISABLED.

Note

Not all configuration options are available with the Kerberos V5 Server. While the Kerberos V5 Server is started and terminated by the Master Server, it runs as a separate process. It uses

a limited subset of server control options. server control options currently supported are: INIT, Program, Priority, and Log-Accepts. To set the SOCKET-PORT option, use the system logical `IP $TELNET_PORT`.

The default is DISABLED; a login prompt will result. When the value is REQUIRED, any user without a valid Kerberos V5 Ticket Granting Ticket (TGT) will be rejected. Finally, if the value is ALLOWED, the user can log-in to the server with or without a valid Kerberos V5 TGT (with a login prompt resulting if no TGT).

For example, to force authentication by any remote telnet client, set the logical as follows:

```
$ DEFINE/SYSTEM IP$TELNET_AUTH REQUIRED
```

Note

Kerberos V5 requires Kerberos (Version 2.0), which is available from the Web site. The Kerberos V5 applications can also run with any Kerberos V5-compliant Key Distribution Center (KDC) software.

2.4.11.5. Configuring the TELNET Server for NTY Devices

The TELNET server parameters INCLUDE-AUTHENTICATION-INDICATION, INCLUDE-PORT-NUMBER and INCLUDE-SEND-LOCATION cause an authentication indication, a port number, or a send location (or all three) to be placed in the TT_ACCPORNAM field of the NTY device control block. These parameters are disabled by default. To enable them, use SERVER-CONFIG (**IP CONFIGURE /SERVER**).

Note

These parameters will not function with the Kerberos 5 Telnet server, as it uses pty devices.

Enable these parameters with the following commands:

```
$ IP CONFIGURE /SERVER
. . . startup messages . . .
SERVER-CONFIG>SELECT TELNET
[The Selected SERVER entry is now TELNET]
SERVER-CONFIG>SET PARAMETERS
You can now add new parameters for TELNET. An empty line terminates.
Add Parameter: INCLUDE-AUTHENTICATION-INDICATION
Add Parameter: INCLUDE-PORT-NUMBER
Add Parameter: INCLUDE-SEND-LOCATION
Add Parameter:
[Service specific parameters for TELNET changed]
SERVER-CONFIG>
```

If these parameters are defined, the appropriate information is stored in the TT_ACCPORNAM field.

These parameters appear in the Remote Port Info field of the **SHOW TERMINAL** and SHOW USERS command output. The INCLUDE-SEND-LOCATION parameter enables support for RFC 779 in the Telnet server. In the following example, the port number is 1021, and the /AUTH qualifier indicates that WHORFIN used authentication to log in from `LOT49.FLOWERS.COM`.

```
$ SHOW TERMINAL
Terminal:    _VTA84:   Device_Type:         VT100
```

```
Owner:       WHORFIN   Physical terminal:   _NTY3:   Remote
Port Info:   LOT49.FLOWERS.COM/1021/AUTH
Input:       9600      LFfill:   0   Width:   80   Parity: None
Output:      9600      CRfill:   0   Page:    24

Terminal Characteristics:

Interactive    Echo               Type_ahead          No Escape
No Hostsync    TTsync  Lowercase  Tab
Wrap           Scope              Remote              No Eightbit
No Broadcast   No Readsync        No Form             Fulldup
No Modem       No Local_echo      Autobaud            Hangup
No Brdcstmbx   No DMA             Altypeahd           Set_speed
No Commsync    Line Editing       Insert editing      No Fallback
No Dialup      No Secure server   Disconnect          No Pasthru
No Syspassword No SIXEL Graphics  No Soft Characters  No Printer Port
Numeric Keypad ANSI_CRT           No Regis            No Block_mode
Advanced_video No Edit_mode       DECEC_CRT2
No DEC_CRT3    No DEC_CRT4        OpenVMS Style Input
```

Authentication information is not valid if someone uses the **LOGOUT /NOHANG** command and then logs in again.

2.4.11.6. Configuring SYSLOG

SYSLOG receives messages from remote IP nodes that have been configured to forward SYSLOG messages to the VSI TCP/IP host. SYSLOG then directs the messages to a file, terminal, or OPCOM. In addition, messages can be forwarded elsewhere. Forwarding messages are specified with the Forwarding command in the SYSLOG configuration file.

SYSLOG is provided with VSI TCP/IP, and you can enable it with SERVER-CONFIG. By default, when VSI TCP/IP is installed, SYSLOG is disabled.

Note

Messages generated by the IP$SERVER process are not sent via SYSLOG, but are instead directed to OPCOM. OPCOM copies the messages to the `SYS$MANAGER:OPERATOR.LOG` file.

With SYSLOG enabled, you can determine the precise output of message classes and specify how priority messages are handled. Message classes and facility keywords are described in Table 2.11.

Table 2.11. SYSLOG Message Classes

Message Type	Facility Keyword
Authorization messages	auth
BOOTP messages	bootpd
Daemon (background processes) messages	daemon
Domain Name System messages	named
GATED (gateway messages)	gated
Kernel messages	kern
LPR messages	lpr
mesages from local facilities	local0 - local7

Message Type	Facility Keyword
NEWS messages	news
Mail utility messages	mail
Network Time Protocol (NTP) messages	ntpd
PPP messages	ppp
Security services messages	security
Messages generated by user programs	user

2.4.11.7. Enabling SYSLOG

Enable SYSLOG with SERVER-CONFIG and modify the file `IP$:SYSLOG.CONFIGURATION`. Enable SYSLOG as follows:

```
$ IP CONFIGURE /SERVER
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>ENABLE SYSLOG
SERVER-CONFIG>EXIT
[Writing configuration to IP_COMMON_ROOT:[IP] SERVICES.MASTER_SERVER]
```

Add entries to the configuration file in the following form: `selector action`.

Separate the fields with tabs. The *`selector`* is a semicolon-separated list of priority specifications in this form: `facility.level[;facility.level].`

- *`facility`* is a keyword (see Table 2.11).

Both facility and level are generated by applications on the remote host. The action specifies how SYSLOG responds to these messages. If the applications on the remote host do not write messages to SYSLOG, they are not displayed.

Possible values for *`facility`* are auth, bootpd, daemon, gated, kern, mail, named, ntpd, security, user, and asterisk (*). (Specify an asterisk to include all messages written to SYSLOG of the specified priority.)

Each facility represents a different source of system message. The level is the priority level for each message. Possible values ranging from most severe to least severe are panic, emerg, alert, crit, err, error, warn, warning, notice, info, debug, and asterisk (*). (Specify an asterisk to include all messages for the specified level.)

Examples of *`facility.level`* statements are:

- *.debug — All debug messages
- ntp.info — Network Time Protocol information messages
- gated.warn — GATED warning messages

The *`action`* is the destination of the message. Possible message destinations include:

Destination	Specified As	Example
Broadcast (RWALL)	Asterisk	*

Destination	Specified As	Example
Console	Use OPCOM	**/OPCOM**
File	Precede the fully specified file name with a slash	**/USERS: [HOLMES]OUTFILE.**
Forwarding	Precede the IP address or host name with an at sign (@)	@192.92.38.1
OPCOM	Precede with a slash	**/OPCOM**
Terminal	Precede the terminal name with a slash and end name with a colon	**/TXA3:**

Each message handled by SYSLOG includes a time stamp, the facility name at the beginning of the message, and a newline character at the end of the message.

Comments are entered in the `SYSLOG.CONFIGURATION` file as pound signs (#) at the beginning of the line.

2.4.11.8. SYSLOG Configuration File Examples

Some example `SYSLOG.CONFIGURATION` file entries are shown below:

```
# SYSLOG.CONFIGURATION examples
# Each entry is tab-separated and has this form:
# selector                action
*.debug                   /OPCOM
kern.panic;kern.warning   /OPCOM
syslog.info               /users:[treefrog]syslog_info_messages.
*.error                   @forwardhost
user.*                    /users:[treefrog]all_user_messages.
```

Chapter 3. Network Time Protocol (NTP)

NTP is the application set for network time protocol functions and refers to the NTPv4 version.

3.1. Overview of NTP

The standard timescale used by most nations of the world is Coordinated Universal Time (UTC) . NTP provides a distributed network clock synchronization protocol, which transmits an accurate reading to one or more clients and adjust each client clock as required.

The synchronization protocol determines the time offset of the server clock relative to the client clock. On request, the server sends a message including the time the request arrived and the time the response was returned. The client includes the time it sent the request in the request message, and records the time the response arrived back from the server as well. With these *timestamps*, the client can determine the server-client propagation delay (by assuming that this is half the round-trip time) and subtract this from the time difference between client and server time settings to determine its clock offset relative to the server. In general, this is a useful approximation; however, in the Internet of today, network paths and the associated delays can differ significantly due to the individual service providers, and this can contribute to error in determining offset. A stable, symmetrical (in terms of propagation delay) and reliable connection to the time server is important in minimizing this type of error.

NTP attempts to compensate for the problem of network instability by allowing the use of several servers as time sources and determining which of them is most reliable through statistical means that compare them to each other. All sources are assumed to have correct times, but those that differ markedly from the group are eventually ignored as having unreliable connections, or being otherwise poor sources of correct time information. This tends to limit malicious activities as well, where a server that reports false times is inserted in a network, as well as bad time servers that result from hardware failure that can have much the same effect.

Clock errors can be due to other causes than variations in network delay. Other causes include latencies in computer hardware and software (jitter), as well as clock oscillator instability (wander). Despite these sources of error, NTP can, over many updates, discipline a clock to stay remarkably close to the actual time, even when a time server is not available for some period

3.2. Programs and Files

There are several programs and files that make up NTP in VSI TCP/IP. These are described in more detail later in this chapter.

3.2.1. Program Files

The following programs make up the NTP implementation in VSI TCP/IP$:

NTPD	The NTP server process used to maintain the system clock and to pass time information to lower stratum clients and servers. While this program runs as a server process, it also functions as a client in requesting time data from other servers on the network.

NTPDC	NTPDC is used to query the NTPD server about its current state and to request changes in that state. Extensive state and statistics information is available through the NTPDC interface. In addition, nearly all the configuration options, which can be specified at startup using NTPD's configuration file may also be specified at run time using NTPDC.
NTPQ	The NTPQ utility program is used to query NTP servers about current state and to request changes in that state. Requests to read and write arbitrary variables can be assembled, with raw and pretty-printed output options being available. NTPQ can also obtain and print a list of peers in a common format by sending multiple queries to the server. NTPQ and NTPDC perform similar functions, but use different protocols to communicate with NTPD.

3.2.2. Configuration Files

NTP uses the following configuration files:

NTP.CONF	IP$:NTP.CONF is used to specify servers from which time information is requested as well as many other aspects of NTPD behavior. See the description of NTPD for a list of options and their definitions. NTP.CONF is read only at NTPD startup, so if you make changes you will need to restart NTPD.
NTP.KEYS	IP$:NTP.KEYS is used to define security information used in authorization operations.
TIMEZONES.DAT	A default set of timezone rules is compiled into NTP. You can use the **IP SET / TIMEZONE** command in conjunction with this file to add other timezone rules. See Section 3.3.3 for more information about timezones and the use of this file.

3.2.3. Other Files

NTP uses the following files:

NTPD.LOG	The NTPD server outputs progress and error information to the IP$:NTPD.LOG file.
NTP.DRIFT	The IP$:NTP.DRIFT file consists of data maintained by NTPD and used to speed up clock frequency adjustments when NTPD is restarted. You should not modify this file.

3.3. Configuring NTP

This section describes how to configure NTP.

3.3.1. NTP Network Design

NTP does not attempt to synchronize clocks to each other. Rather, each server attempts to synchronize to Universal Coordinated Time (UTC) using the best available sources and available transmission paths to those sources. This is a fine point, which is worth understanding. A group of NTP-synchronized clocks may be close to each other in time, but this is not a consequence of the clocks in the group having synchronized to each other, but rather because each clock has synchronized closely to UTC via the best source it has access to.

The most important factor in providing accurate, reliable time is the selection of modes and servers to be used in the configuration file. An NTP network should consist of a multiply redundant hierarchy of servers and clients, with each level in the hierarchy identified by stratum number. Primary servers operate at stratum one and provide synchronization to secondary servers operating at stratum two and so on to higher strata. In this hierarchy, clients are simply servers that have no dependents.

Determine which list of peers/servers you want to include in the configuration file. Include at least one (but preferably two) peer or server hosts that you are assured:

- Are running NTP
- Provide accurate time
- Synchronize to Internet Time Servers (if they are not themselves ITSs)

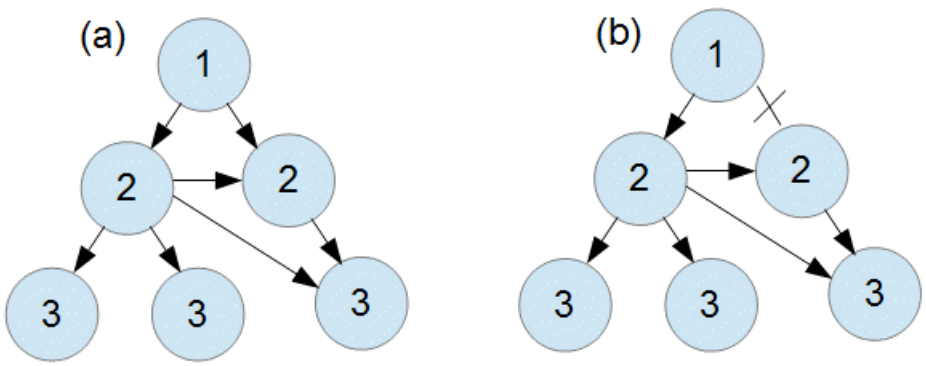
Two hosts provide reliability in case one goes down. You do not need to identify what stratum each host is. NTP determines this through the reference information it sends in its data exchanges.

NTP data is exchanged periodically between hosts as encapsulated in UDP datagrams, and adjustments are made based on an NTP algorithm. The frequency of exchange is related to the server's experience of time corrections. The more accurate the local clock becomes over time and after many adjustments, the less often the NTP server checks the need for corrections. The frequency of exchange is rarely intrusive to normal network operation. Also, the unreliability of UDP has no measurable impact on the process, and the process does not depend on any such reliability.

Primary servers are servers with reliable time sources, such as GPS receivers or atomic clocks, and can be found on the public internet, or set up within an intranet. Stratum numbers equate to the number of intermediate servers (or *hops*) between a given host and the Stratum 1 server it is ultimately referencing. Stratum numbers are not assigned statically, but change as server connections change. NTP servers can be (and often are) other types of systems running NTP, not just OpenVMS systems.

The stratum method allows for backup timekeeping in case a node or connection goes down, and stratum numbers may change as a consequence. In Figure 3.1*(a)*, each node has a stratum number based on hop count, with the ITS at the top of the pyramid. The solid arrows are the active synchronization paths and direction of timing information flow; the lighter arrows are background synchronization paths where timing information is exchanged but not necessarily used for synchronization. Figure 3.1*(b)* shows the same network with one of the connections broken — note that the stratum for the affected peer increases from 2 to 3.

Figure 3.1. Synchronization Through Strata



NTP makes local system time adjustments by either *slewing* or *stepping* the clock. Slewing runs the clock faster or slower than its normal frequency. Stepping sets the clock immediately to the correct time. Stepping occurs infrequently, only when there is a large time offset to adjust, such as when starting NTPD or when making daylight savings time (DST) changes.

Under some circumstances it can be disruptive to step the clock, such as when running database software that journals transactions. Such software can become very confused when a transaction is completed prior to the time at which it began, such as can happen when a clock is stepped backwards during a transaction's lifetime. In such cases, the *slewalways* configuration option can be used to turn off stepping of the clock, and force all adjustments to be made by slewing. For large time changes, such as DST changeovers, the adjustment can take a long time (several hours) to complete, and during this time the system's time will not be correct. For this reason, it is not wise to allow a system set for *slewalways* to act as a server to another system.

In determining your NTP network design, keep in mind the way that the NTP protocol works, and how NTPD will determine the correct time. There should be several time servers in the configuration for each node, with good, reliable and non-congested paths between them. Nodes that will act only as clients can use the slewalways option, but nodes used as time sources by other nodes should generally allow stepping of the time so that inaccuate times are not reported for extended periods at Daylight Savings Time (DST) changeovers. See Section 3.3.2 for more information on the slewalways option and Section 3.3.3 for more information on DST handling.

3.3.1.1. Authentication

NTP implements a general purpose address- and mask-based restriction list (see the ***`restrict`*** config option). While this is not adequate to prevent hacking attacks, it can be useful to lock out a malfunctioning server that is disrupting normal operations. See Section 3.3.6 for more information.

The NTP standard specifies an extension which provides cryptographic authentication of received NTP packets. This is implemented in NTPD using the MD5 algorithm to compute a digital signature, or message digest. See Section 3.3.7 for more information.

3.3.1.2. Finding Servers

In many large organizations, there is an administrator for the organization's networks which handles management of various network services. This person or department can usually provide information on local NTP servers, and often suggest configurations that are known to work.

There are also a number of publicly available time servers on the internet. A list of such servers can be accessed via the following web site: http://www.eecis.udel.edu/~ntp. These data are updated on a regular basis using information provided voluntarily by various site administrators.

3.3.2. NTP.CONF

The `NTP.CONF` file is used to specify the initial configuration of the NTPD server. It contains information about servers and peers, modes of operation, non-default file names, and other configuration data. See the following table for a list of available options and description.

Important

Server and peer specifications can have -4 or -6 options added to force use of IPv6 or IPv4 when DNS names for both exist. For example, add the following to your NTP.CONF file to force your server to use ipv4:

```
server -4 yourserver.xyz.com
```

Table 3.1. NTP.CONF file options

Option	Description
`peer`	`peer` [address] [version 4] [key 0] [minpoll 6] [maxpoll 10] Specifies that the server is to operate in symmetric active mode with the specified remote server. In this mode, the local server can be synchronized to the remote server and, in addition, the remote server can be synchronized by the local server. This is useful in a network of servers where, depending on various failure scenarios, either the local or remote server may be the better source of time. The *`address`* can be a domain name or an IP address in dotted quad notation The *`key`* specifies that all packets sent to an address are to include authentication fields encrypted using the specified key identifier, which is an unsigned 32-bit integer. The default is to not include an encryption field. *`Version`* specifies the protocol version number to be used for outgoing NTP packets. Versions 1, 2, 3 and 4 are the choices, with version 4 the default.
`server`	`server` [address] [version 4] [key 0] [minpoll 6] [maxpoll 10] Specifies that the local server is to operate in client mode with the specified remote server. In this mode, the local server can be synchronized to the remote server, but the remote server can never be synchronized to the local server. The *`address`* can be a domain name or an IP address in dotted quad notation The *`key`* specifies that all packets sent to an address are to include authentication fields encrypted using the specified key identifier, which is an unsigned 32-bit integer. The default is to not include an encryption field. *`Version`* specifies the protocol version number to be used for outgoing NTP packets. Versions 1, 2, 3 and 4 are the choices, with version 4 the default.
`broadcast`	`broadcast` [address] [version 4] [key 0] [ttl 1] Specifies broadcast mode, where the local server sends periodic broadcast messages to a client population at the broadcast/multicast address specified. This specification applies only to the local server operating as a sender. For operation as a broadcast client, see the **broadcastclient** option that follows. In this mode address is usually the broadcast address on (one of) the local networks.

Option	Description
	The *key* specifies that all packets sent to an address are to include authentication fields encrypted using the specified key identifier, which is an unsigned 32-bit integer. The default is to not include an encryption field. *Version* specifies the protocol version number to be used for outgoing NTP packets. Versions 1, 2, 3 and 4 are the choices, with version 4 the default. *ttl* specifies the number of routers to pass through before the packet is discarded. The default is 127 routers.
broadcastclient	broadcastclient This command directs the local server to listen for broadcast messages at the broadcast address of the local network. Upon hearing a broadcast message for the first time, the local server measures the nominal network delay using a brief client/server exchange with the remote server, then enters the **broadcastclient** mode, in which it listens for and synchronizes to succeeding broadcast messages. **Note** In order to avoid accidental or malicious disruption in this mode, both the local and remote servers should operate using authentication and the same trusted key and key identifier.
multicastclient	multicastclient [address] Specifies that this host is a multicast client for multicasts to the specified multicast address.
manycastclient	manycastclient [address] [version 4] [key 0] [minpoll 6] [maxpoll 10] Specify that this host is a manycast client and provide relevant settings.
manycastserver	manycastserver [address] Specify that this host is a manycast server for the given address.
broadcastdelay	broadcastdelay 0.004 The broadcast and multicast modes require a special calibration to determine the network delay between the local and remote servers. Ordinarily, this is done automatically by the initial protocol exchanges between the client and server. In some cases, the calibration procedure may fail due to network or server access controls, for example. This command specifies the default delay to be used under these circumstances. Typically (for Ethernet), a number between 0.003 and 0.007 seconds is appropriate. The default when this command is not used is 0.004 seconds.
restrict	restrict [address] [mask 255.255.255.0] ignore | noserve | notrust | noquery

Option	Description
	Restrict access from and to the specified address for the specified types of access. The *address* argument, expressed in dotted quad form, is the address of a host or network. The *mask* argument, also expressed in dotted quad form, defaults to 255.255.255.255, meaning that the *address* is treated as the address of an individual host. A default entry (address 0.0.0.0, mask 0.0.0.0) is always included and, given the sort algorithm, is always the first entry in the list. **Note** While *numeric-address* is normally given in dotted-quad format, the text string default, with no mask option, can be used to indicate the default entry. *Ignore* - Ignores all packets from hosts which match this entry. If this flag is specified, neither queries nor time server polls are responded to. *noquery* - ignores all NTP mode 6 and 7 packets (information queries and configuration requests generated by NTPQ and NTPDC) from the source. Time service is not affected. *noserve* - Ignores NTP packets whose mode is other than 6 or 7. In effect, time service is denied, though queries may still be permitted. *notrust* - Treats these hosts normally in other respects, but never uses them as synchronization sources.
driftfile	driftfile *file_name* Specify the name of the drift file. The default is IP$:NTP.DRIFT if this option is not used. The drift file is used to record the frequency offset of the local clock oscillator. If the file exists, it is read at startup in order to set the initial frequency offset and then updated once per hour with the current frequency offset computed by the daemon. If the file does not exist or this command is not given, the initial frequency offset is assumed zero. In this case, it may take some hours for the frequency to stabilize and the residual timing errors to subside.
keys	keys *file_name* Specify the name of the keys file. The default is IP$:NTP.KEYS if this option is not used.
statsdir	statsdir path Indicates the full path of a directory where statistics files should be created. This keyword allows the (otherwise constant) **filegen** filename prefix to be modified for file generation sets, which is useful for handling statistics logs.
filegen	filegen [file*filename*] [type*typename*] [enable | disable]

Option	Description
	Configures the generation fileset name. Generation filesets provide a means for handling files that are continuously growing during the lifetime of a server. Server statistics are a typical example for such files. At most one element of the set is being written to at any one time. The type given specifies when and how data is directed to a new element of the set. *filename* - This string is directly concatenated to the directory IP$: or the directory prefix specified using the statsdir option. The suffix for this filename is generated according to the type of a fileset. *typename* - A file generation set is characterized the following *typenames*: **none** — One element of the fileset is used for each, NTPD server. **day** — One file generation set element is created per day. A day is defined as the period between 00:00 and 24:00 UTC. The fileset member suffix consists of a dot (.) and a day specification in the form YYYYMMDD. YYYY is a 4-digit year number (such as 2003). MM is a two digit month number. DD is a two digit day number. Thus, all information written at 10 December 2002 would end up in a file named *prefix filename*.20021210. **week** — Any fileset member contains data related to a certain week of a year. The term week is defined by computing day-of-year modulo 7. Elements of such a file generation set are distinguished by appending the following suffix to the fileset filename base: a dot, a 4-digit year number, the letter **W**, and a 2-digit week number. For example, information from January 10th, 2003 would end up in a file with suffix .2003W1.
filegen	**month** — One generation fileset element is generated per month. The filename suffix consists of a dot, a 4-digit year number, and a 2-digit month. **year** — One generation file element is generated per year. The filename suffix consists of a dot and a 4-digit year number. **age** — This type of file generation sets changes to a new element of the fileset every 24 hours of server operation. The filename suffix consists of a dot, the letter **a**, and an 8-digit number. This number is taken to be the number of seconds the server is running at the start of the corresponding 24-hour period. Information is only written to a file generation by specifying **enable**; output is prevented by specifying **disable**.
publickey	publickey *file_name* Specify the publickeys file location.
privatekey	privatekey *file_name* Specify the private key file location.

Option	Description
`clientlimit`	`clientlimit` *`n`* Sets the `client_limit` variable that limits the number of simultaneous access-controlled clients. The default value is 3.
`clientperiod`	`clientperiod` [3600] Sets the `client_limit_period` variable that specifies the number of seconds after which a client is considered inactive and thus no longer is counted for client limit restriction. The default value is 3600 seconds.
`trustedkey`	`trustedkey` [key] Specifies the encryption key identifiers which are trusted for the purposes of authenticating peers suitable for synchronization. The authentication procedures require that both the local and remote servers share the same key and key identifier for this purpose, although different keys can be used with different servers. The key arguments are 32-bit unsigned integers. **Note** NTP key 0 is fixed and globally known. If meaningful authentication is to be performed, the 0 key should not be trusted.
`requestkey`	`requestkey` [key] Specifies the key identifier to use with the NTPDC program, which uses a proprietary protocol specific to this distribution of NTPD. The key argument to this command is a 32-bit unsigned integer. If no **requestkey** command is included in the configuration file, or if the keys do not match, NTPDC requests are ignored.
`controlkey`	`controlkey` [key] Specifies the key identifier to use with the NTPQ program, which uses the standard protocol defined in RFC 1305. The key argument to this command is a 32-bit unsigned integer. If no **controlkey** command is included in the configuration file, or if the keys do not match, NTPQ requests are ignored.
`setvar`	`setvar` [value] This command adds an additional system variable. These variables can be used to distribute additional information such as the access policy. If the variable of the form *`name = value`* is followed by the **default** keyword, the variable is listed as part of the default system variables (**ntpq rv** command). These additional variables serve informational purposes only. They are not related to the protocol other than that they can be listed. The known protocol variables always override any variables defined using the **setvar** mechanism.
`logfile`	`logfile` *`file_name`* Specify the logfile name. The default is `IP$:NTPD.LOG` if this option is not specified.

Option	Description
`logconfig`	`logconfig` [+ | - | =] [{ sync | sys | peer | clock } {,all} { info | statistics | events | status }] ... Specify logging options.
`enable`	`enable` auth | bclient | ntp | kernel | monitor | stats | calibrate Enable various options. *`auth`* - Enables the server to synchronize with unconfigured peers only if the peer was correctly authenticated using a trusted key and key identifier. The default for this setting is **disable**. *`bclient`* - When enabled, this is identical to the **broadcastclient** command. The default for this flag is **disable**. *`ntp`* - Enables the server to adjust its local clock by means of NTP. If disabled, the local clock free-runs at its intrinsic time and frequency offset. This flag is useful in case the local clock is controlled by some other device or protocol and NTP is used only to provide synchronization to other clients. The default for this flag is **enable**. *`kernel`* - this setting is not used in the VSI TCP/IP implementation. *`monitor`* - Enables the monitoring facility. See the **monlist** command of the NTPDC program for further information. The default for this flag is **enable**. *`stats`* - Enables the statistics facility. The default for this flag is **enable**. *`calibrate`* - this setting is not used in the VSI TCP/IP implementation.
`disable`	`disable` auth | bclient | ntp | kernel | monitor | | stats | calibrate Disable various options. See the **enable** entry for details.
`slewalways`	`slewalways` Specify that the clock time is always to be slewed, never stepped. NTPD normally steps the clock when there is a relatively large time error to adjust. The **slewalways** command directs the local NTP server to always slew the clock, regardless of how large the required correction is. This command is useful to avoid an abrupt one hour clock change when daylight savings time (DST) changes occur. For DST changes when **slewalways** is specified, NTPD slews the clock over a period of about 6 hours.
`tinker panic`	`tinker panic` *`max_adjust_time`* Set the maximum time change that will be allowed. If non-zero, this value should always be at least 4000 seconds to allow for DST time changes even on systems with large time errors. This was known as “WAYTOOBIG” in XNTP.

Option	Description
	If set to zero, the panic sanity check is disabled and a clock offset of any value will be accepted.
`debug`	`debug` [level] Set the debug logging severity level.
`set_vms_logicals`	`set_vms_logicals` Causes the NTP server to also adjust the value of the OpenVMS logicals `SYS$TIMEZONE_DIFFERENTIAL`, `SYS$TIMEZONE_DAYLIGHT_SAVING` and `SYS$TIMEZONE_NAME` when it changes the `IP$TIMEZONE` logical at DST start or end. The following files will also be updated: `DTSS$TIMEZONE_DIFFERENTIAL` and `SYS$TIMEZONE.DAT`. NTP does NOT set `SYS$TIMEZONE_RULE`, which generally does not change. The format of `SYS$TIMEZONE_RULE` is specified in `SYS$MANAGER:UTC$TIME_SETUP.COM`.
`call_dst_proc`	`call_dst_proc` Causes the NTP server to spawn a subprocess to execute the `IP$:NTPD_DST_PROC.COM` procedure, if such a procedure file exists with the proper protections, when changing into or out of DST, or when first starting up.
`set_clock_daily`	`set_clock_daily` When this is included in `NTP.CONF`, then NTPD will only make one call a day (instead of once an hour) to the routine that sets the TOY clock to insure that the value is preserved. This reduces the number of entries in the Integrity system event logs. NTPD may set the clock more often than daily, but it will be done only to correct any drift that is detected. In our tests on a RX2600 this happened approximately every 6 hours.

3.3.3. Timezone Configuration and Hardware Clock Overview

By OpenVMS convention, the system clock is usually set to the local time, but network protocols represent time in Coordinated Universal Time (UTC, sometimes referred to as GMT). To convert between local time and UTC, VSI TCP/IP uses built-in rules or rules provided by the system manager.

Each country or geographical area has its own names for timezones and its own rules for Daylight Savings Time (DST). The names of these timezones and rules are not necessarily unique. For example, "EST" could refer to the United States Eastern Standard Time, the Canadian Eastern Standard Time (which uses different DST rules), or the Australian Eastern Standard Time (which is a different offset from UTC as well as having different DST rules).

VSI TCP/IP uses the name of the local timezone as specified by the system manager, along with timezone rules, to calculate the offset between the local time and UTC, so it is important that an appropriate set of timezone rules be selected for the location where the system is located.

VSI TCP/IP assumes that the hardware clock is always set exactly to local time. For a smooth transition to and from Daylight Savings Time the hardware clock must be reset at the appropriate time.

Using a military time zone or an explicit GMT offset disables automatic Daylight Savings Time transitions.

3.3.4. Timezone Support

Because it is impossible to anticipate every country or area in which VSI TCP/IP might be used, and because the Daylight Savings Time rules are subject to change by government action, VSI TCP/IP permits you to write your own site-specific timezone rules. There are two types of timezone rules: compiled-in and loadable.

- Compiled-in rules are geographically centered around the United States but also include foreign timezones whose names do not conflict with the U.S. timezones.
- Loadable rules are selected with the **IP NETCONFIG SET TIMEZONE-RULES** command and can be used to override the compiled-in rules.

VSI TCP/IP includes a database of the most common loadable rules. You can select these rules as-is, or modify them to conform to the correct local timezone rules.

When VSI TCP/IP searches the timezone rules looking for a zone, it first searches the loadable rules in the order they are specified, then searches the compiled-in rules. This method allows you to change the compiled-in rules by loading rules that override them.

In addition to the standard one-letter U.S. military time zones and timezones of the form GMT +*hh:mm* or GMT-*hh:mm*, there are compiled-in timezone rules supported by VSI TCP/IP which are shown in Table 3.2.

Table 3.2. Compiled-In Timezone Rules

Timezone Name	GMT Offset	DST Rules	Area or Country
EST or EDT	-5 hours	U.S. Federal	Eastern United States
CST or CDT	-6 hours	U.S. Federal	Central United States
MST or MDT	-7 hours	U.S. Federal	Mountain United States
PST or PDT	-8 hours	U.S. Federal	Pacific United States
YST or YDT	-9 hours	U.S. Federal	Yukon
HST	-10	-none-	Hawaii
NST or NDT	-3:30 hours	Canadian	Canadian Newfoundland
AST or ADT	-4 hours	Canadian	Canadian Atlantic
JST	+9 hours	-none-	Japan
SST	+8 hours	-none-	Singapore
GMT	+0 hours	-none-	Greenwich Mean Time
GMT or BST	+0 hours	British	Britain
WET or WET-DST	+0 hours	European	Western Europe
MET or MET-DST	+1 hour	European	Middle Europe

Timezone Name	GMT Offset	DST Rules	Area or Country
CET or CET-DST	+1 hour	European	Central Europe (Middle Europe)
EET or EET-DST	+2 hours	European	Eastern Europe
NZST or NZDT	+12 hours	New Zealand	New Zealand

3.3.5. Loadable Timezone Rules

Loadable timezone rules provided with VSI TCP/IP are in the text file `IP$:TIMEZONES.DAT`. You can copy this file to `IP$:TIMEZONES.LOCAL`, and then add user-written timezone rules to override the compiled-in rules.

Loadable timezone rules consist of three parts:

COUNTRY	A collection of timezones (ZONEs). For example, the country US selects all U.S. timezones. This provides a convenient way to select groups of timezones.
ZONE	A specification of a particular timezone, including the name of the zone, the UTC offset, the DST rules in effect, and the name to use while DST is in effect.
RULE	A rule for determining when DST is in effect.

3.3.5.1. Format of COUNTRY Specification

`COUNTRY` *`countryname zonename`* [zonename . . .]

The COUNTRY specification gives the name of a geographical area and the names of the timezones associated with it. This provides a way to group timezones so they may be selected more conveniently.

The following example shows the definition of the country "US", listing the zones corresponding to the United States. The example for Arizona is slightly different, showing the zone "US/Arizona" instead of "US/Mountain." ("US/Arizona" is the definition of a Mountain timezone that does not observe Daylight Savings Time.)

```
Country US US/Eastern US/Central US/Mountain US/Pacific US/Yukon US/Hawaii
Country US/Arizona -
US/Eastern US/Central US/Arizona US/Pacific US/Yukon US/Hawaii
```

3.3.5.2. Format of ZONE Specification

`ZONE` *`zonename gmtoffset rulename standard-name dst-name`* [COMPILED_IN]

`zonename`	The name by which this zone can be selected, or the name by which it is referred to in a COUNTRY specification.
`gmtoffset`	This zone's standard time offset from UTC.
`rulename`	Is the name of the RULE specification that determines when DST is in effect for this zone. The rulename may be an underscore (_) to indicate that this zone does not use DST.
`standard-name` and *`dst-name`*	The names by which this zone is referred to during standard time, and during Daylight Savings Time, respectively. These are the names by which **SET TIMEZONE** selects the local timezone.

The ZONE specification describes a timezone:

If there are no DST rules, the dst-name should be specified as an underscore (_). The optional `COMPILED_IN` keyword indicates that this rule is compiled-in and need not be loaded, as long as no other rules conflict with it. If you edit a `COMPILED_IN ZONE` specification, you must remove the `COMPILED_IN` keyword to force the ZONE specification to be loaded.

The following example shows the definition of the normal United States Mountain timezone. The Arizona example shows the definition of a Mountain timezone that does not observe Daylight Savings Time.

```
Zone   US/Mountain   -7:00   US   MST   MDT COMPILED_IN
Zone   US/Arizona    -7:00   _    MST
```

3.3.5.3. Format of a RULE Specification

`RULE` *`rulename startyear ruletype save`* start-date end-date

The RULE specification describes a rule or set of rules for determining at what times DST is in effect:

`rulename`	The name of the RULE specification in ZONE specifications.
`startyear`	The year during which this DST rule takes effect. The rule remains in effect until a later *`startyear`* is specified in a rule with the name *`rulename`*.
`ruletype`	Specifies the type of DST rules. There are three permitted values: • DST indicates normal Northern-Hemisphere Daylight Savings Time rules, which switch at the time and date indicated. • REV_DST indicates normal Southern-Hemisphere Daylight Savings Time rules. NULL indicates that no Daylight Savings Time is in effect during the specified years.
`save`	Indicates the difference between Standard Time and DST.
`start-date` and *`end-date`*	Specify the starting and ending dates for DST. Specific dates can be specified, or rules such as "First Sunday" or "Last Sunday" can be used. For days other than the “First” or “Last” you must use dayname >= date. For example the second Sunday is expressed as Sunday >= 8. See the file `IP$:TIMEZONES.DAT` for examples of specifying dates.

The following example illustrates the United States Federal Daylight Savings Time rules:

```
Rule US 2017 DST 1:00   Sunday>=8 March        2:00 First Sunday November
 2:00
Rule US 2016 DST 1:00   First   Sunday April   2:00 Last Sunday October
 2:00
Rule US 2015 DST 1:00   Last    Sunday April   2:00 Last Sunday October
 2:00
Rule US 2012 DST 1:00   23      February       2:00 Last Sunday October
 2:00
Rule US 2011 DST 1:00   6       January        2:00 Last Sunday October
 2:00
```

```
Rule US 2010 DST 1:00   Last    Sunday April   2:00 Last Sunday October
 2:00
```

3.3.5.4. Loadable Timezone Rules Provided with VSI TCP/IP

Table 3.3 shows the loadable rules provided in the `IP$:TIMEZONES.DAT` file which you may modify or augment as appropriate for your location.

Table 3.3. Loadable Timezone Rules

Country Name	Rule Name	Timezone Name	GMT Offset	DST Rules
	GMT	GMT[1]	0 hours	-none-
	UT	UTa	0 hours	-none-
US-Military	US-Military/Za	Z	0 hours	-none-
US-Military	US-Military/Aa	A	-1 hour	-none-
US-Military	US-Military/Ba	B	-2 hours	-none-
US-Military	US-Military/Ca	C	-3 hours	-none-
US-Military	US-Military/Da	D	-4 hours	-none-
US-Military	US-Military/Ea	E	-5 hours	-none-
US-Military	US-Military/Fa	F	-6 hours	-none-
US-Military	US-Military/Ga	G	-7 hours	-none-
US-Military	US-Military/Ha	H	-8 hours	-none-
US-Military	US-Military/Ia	I	-9 hours	-none-
US-Military	US-Military/Ka	K	-10 hours	-none-
US-Military	US-Military/La	L	-11 hours	-none-
US-Military	US-Military/Ma	M	-12 hours	-none-
US-Military	US-Military/Na	N	1 hour	-none-
US-Military	US-Military/Oa	O	2 hours	-none-
US-Military	US-Military/Pa	P	3 hours	-none-
US-Military	US-Military/Qa	Q	4 hours	-none-
US-Military	US-Military/Ra	R	5 hours	-none-
US-Military	US-Military/Sa	S	6 hours	-none-
US-Military	US-Military/Ta	T	7 hours	-none-
US-Military	US-Military/Ua	U	8 hours	-none-
US-Military	US-Military/Va	V	9 hours	-none-
US-Military	US-Military/Wa	W	10 hours	-none-
US-Military	US-Military/Xa	X	11 hours	-none-
US-Military	US-Military/Ya	Y	12 hours	-none-
US	US/Easterna	EST/EDT	-5 hours	US Federal
US	US/Centrala	CST/CDT	-6 hours	US Federal
US	US/Mountaina	MST/MDT	-7 hours	US Federal
US	US/Pacifica	PST/PDT	-8 hours	US Federal

Country Name	Rule Name	Timezone Name	GMT Offset	DST Rules
US	US/Yukona	YST/YDT	-9 hours	US Federal
US	US/Hawaiia	HST	-10 hours	-none-
US/East-Indiana	US/East-Indiana	EST	-5 hours	-none-
US/Arizona	US/Arizona	MST	-7 hours	-none-
Canada	Canada/ Newfoundlanda	NST/NDT	-3:30 hours	Canadian
Canada	Canada/Atlantica	AST/ADT	-4 hours	Canadian
Canada	Canada/Eastern	EST/EDT	-5 hours	Canadian
Canada	Canada/Central	CST/CDT	-6 hours	Canadian
Canada	Canada/Mountain	MST/MDT	-7 hours	Canadian
Canada	Canada/Pacific	PST/PDT	-8 hours	Canadian
Canada	Canada/Yukon	YST/YDT	-9 hours	Canadian
Canada	Canada/ Saskatchewan	CST	-6 hours	-none-
Israel	Israel	IST/DST	+2 hours	Israeli
Australia	Australia/Tasmania	EST	10 hours	Australian
Australia	Australia/ Queensland	EST	10 hours	-none-
Australia	Australia/North	CST	9:30 hours	-none-
Australia	Australia/West	WST	8 hours	-none-
Australia	Australia/South	CST	9:30 hours	Australian
Australia	Australia/Victoria	CST	10 hours	Australian
Australia	Australia/NSW	CST	10 hours	Australian
Australia	Australia/ Yarcowinna	CST	9:30 hours	Australian
Australia	Australia/LHI	CST	10:30 hours	Australian
Europe	Britaina	GMT/BST	0 hours	GB-Eire
Europe	Europe/Westerna	WET/WET-DST	0 hours	W-Eur
Europe	Europe/Middlea	MET/MET-DST	1 hour	M-Eur
Europe	Europe/Centrala	CET/CET-DST	1 hour	M-Eur
Europe	Europe/Easterna	EET/EET-DST	2 hours	E-Eur
	Iceland	GMT	1 hour	-none-
	Poland	MET	1 hour	W-Eur
	Turkey	EET/EET/DST	3 hours	Turkey
Japan	Japana	JST	+9 hours	-none-
Singapore	Singaporea	SST	+8 hours	-none-
NewZealand	NewZealanda	NZST/NZDT	+12 hours	New Zealand

[1]This timezone is compiled-in also.

3.3.5.5. Selecting Timezone Rules

Timezone rules and the local timezone name are set in the VSI TCP/IP system startup command file, `IP$:IP$SYSTARTUP.COM`. You can use the VSI TCP/IP NET-CONFIG utility to specify the local timezone and which rules to load using the **SET TIMEZONE** and **SET TIMEZONE-RULES** commands. The following example shows how to select the United States Arizona rules and the local timezone MST:

```
$ IP CONFIGURE
NET-CONFIG>SET TIMEZONE MST
NET-CONFIG>SET TIMEZONE-RULES US/ARIZONA
NET-CONFIG>EXIT
```

3.3.5.6. Using the call_dst_proc option

When NTPD is started, and whenever the local timezone shifts between daylight savings DST and standard (STD) time, if the local zone rule specifies such behavior, the NTPD server will check the `IP$TIMEZONE` logical, and set it if required. The setting will only be between the DST name and the STD name for the zone, so the configuration described above is still necessary, but if your system was down during a DST shft, this can correct the logical name to match the current system clock time and the applicable zone rule when NTPD is started. If your `NTP.CONF` file specifies the “set_vms_logicals” option, the `SYS$TIMEZONE_DIFFERENTIAL`, `SYS$TIMEZONE_DAYLIGHT_SAVING` and `SYS$TIMEZONE_NAME` logicals will be updated as well.

Since there are many systems with other time-related logical names, or other items that may need updating or adjusting based on a DST change, the call_dst_proc option has been provided. If this option is used in `NTP.CONF`, the NTPD server will look for a file called `IP$:NTPD_DST_PROC.COM` any time it checks on the `IP$TIMEZONE` logical (at startup and at a DST shift). If this file exists, and has the proper protections (no WORLD write or execute access, and owned by SYSTEM [1,4])) a sub-process will be spawned to execute it. This procedure can contain any commands needed, but care should be exercised in constructing this file, as it will be executing with the same privileges as the NTPD process. A “placeholder” procedure is included with VSI TCP/IP, but its contents are all comments and will do nothing as shipped.

The invocation of the `IP$:NTPD_DST_PROC.COM` procedure will be equivalent to this:

```
@IP$:NTPD_DST_PROC.COM p1 p2 p3 p4 p5
```

Where:

- p1 = Current timezone name - string (e.g. "EST" or "EDT")
- p2 = Timezone offset in seconds - integer (e.g. "-18000" or "-14400")
- p3 = DST in effect? - boolean ("Y", "N")
- p4 = In Twilight Zone? - boolean ("Y", "N")
- p5 = Startup or DST change? - string ("START" or "DST")

P1, the Current Timezone Name, is a string specifying the current name of the local timezone. For North American Eastern Standard Time, this will be “EST” in the winter, and “EDT” in the summer, when DST is active. For timezones that do not do DST, it will always be the zone name.

P2, the Timezone Offset, is a signed integer specifying the offset, in seconds, from UTC for the local zone, at the current time. For North American Eastern Standard Time this is “-18000” (-5 hours), for the same zone with DST in effect it is “-14400” (-4 hours).

P3, the DST flag. This will be “Y” if DST is currently in effect for the zone, and “N” if it is not, or if the zone does not do DST.

P4, the Twilight Zone flag. When a zone exits from DST, it sets its time back an hour. This means that for that hour, the time *appears* to be a DST time by the local DST rules, but is not really, since DST has already ended. That hour is called the “twilight zone” by VSI TCP/IP NTP. If the current time is in that period, the P4 parameter will be “Y”, otherwise it will be “N”.

P5, the startup/DST flag. This tells the procedure whether it is being called as a part of NTPD’s startup processing, or as part of a DST change.

These parameters are provided so that the procedure can take different action under different conditions. They may all be ignored if that is appropriate. The NTPD server doesn’t depend on any particular behavior, so long as the `IP$TIMEZONE` logical is left alone and the system clock is not altered. The final completion status of the called procedure will be logged by the NTPD server, along with the PID of the spawned sub-process.

3.3.6. Access Control Commands

NTP implements a general purpose address- and mask-based restriction list (see the ***`restrict`*** config option). The list is sorted by address and by mask, and the list is searched in this order for matches, with the last match found defining the restriction flags associated with the incoming packets. The source address of incoming packets is used for the match, with the 32-bit address combined with the mask associated with the restriction entry and then compared with the entry’s address (which was also combined with the mask) to look for a match.

The restriction facility was implemented to conform with the access policies for the original NSFnet backbone time servers. While this facility may be otherwise useful for keeping unwanted or broken remote time servers from affecting your own, it should not be considered an alternative to the standard NTP authentication facility. Source address based restrictions are easily circumvented by a determined hacker.

3.3.7. Authentication Using a Keys File

The NTP standard specifies an extension which provides cryptographic authentication of received NTP packets. This is implemented in NTPD using the MD5 algorithm to compute a digital signature, or message digest. The specification allows any one of possibly four billion keys, numbered with 32-bit key identifiers, to be used to authenticate an association. The servers involved in an association must agree on the key and key identifier used to authenticate their messages.

Keys and related information are specified in the file `IP$:NTP.KEYS`, which should be exchanged and stored using secure procedures. There are three classes of keys involved in the current implementation. One class is used for ordinary NTP associations, another for the NTPQ utility program, and the third for the NTPDC utility program.

Key File Format

For MD5, keys are 64 bits (8 bytes), read from the `IP$:NTP.KEYS` file. While key number 0 is fixed by the NTP standard (as 64 zero bits) and may not be changed, one or more of the keys numbered 1 through 15 may be arbitrarily set in the keys file.

The keys file uses the same comment conventions as the configuration file. Key entries use a fixed format of the form:

keyno type key

- ***keyno*** is a positive integer
- ***type*** is a single character **M** for the MD5 key format
- ***key*** is the key itself

The key is a one to eight character ASCII string using the MD5 authentication scheme.

Note

Both the keys and the authentication scheme must be identical between a set of peers sharing the same key number. The keys used by the NTPQ and NTPDC programs are checked against passwords requested by the programs and entered by hand.

3.3.8. NTP Utilities

There are several utility programs included with NTP. These allow setting the system clock from a time server, querrying and controlling NTP servers on the local system or on remote hosts, and tracing the chain of time servers back to the top stratum server being used to set the local time.

These utilities are all accessible through the IP command (i.e. “IP NTPD . . .”), or as DCL foreign command symbols through the use of the `IP$:NTP_DEFINE.COM` procedure to define these commands. The same image is executed in either case and it is mostly a matter of personal preference which is used. The foreign commands can be undefined by use of the `IP$:NTP_UNDEFINE.COM` procedure.

3.4. NTPDC

The NTPDC utility is used to query the NTPD server about its current state and to request changes in that state. The program runs interactively or uses command line arguments. Extensive state and statistics information is available through the NTPDC interface. In addition, nearly all the configuration options that can be specified at startup using NTPD’s configuration file may also be specified at run-time using NTPDC.

The NTPDC utility uses NTP mode 7 packets to communicate with the NTP server, and can be used to query any compatible server on the network which permits it.

Note

Since NTP is a UDP protocol, this communication is somewhat unreliable, especially over large distances, in terms of network topology. NTPDC makes no attempt to retransmit requests, and times out requests if the remote host is not heard from within a suitable timeout time.

NTPDC’s operation is specific to the NTPD implementation and can be expected to work only with this, and possibly some previous versions, of the daemon. Requests from a remote NTPDC program that affect the state of the local server must be authenticated, which requires both the remote program and local server to share a common key and key identifier.

3.4.1. Command Line Format

`ntpdc` [-ilnps] [-c command] [host] [...]

3.4.2. Command Line Arguments

If command line arguments are omitted, NTPDC runs in interactive mode.

[-c]

The *command* that follows is interpreted as an interactive format command and is added to the list of commands to be executed on the specified host(s). The *command* must be in double quotes if it consists of more than one word. Multiple **-c** options can be given.

[-i]

Force NTPDC to operate in interactive mode. Prompts will be written to the standard output and commands read from the standard input.

[-l]

Obtain a list of peers which are known to the server(s). This switch is equivalent to *-c listpeers*.

`-n`

Displays all host addresses in dotted quad numeric format rather than converting them to canonical hostnames.

`-p`

Print a list of the peers known to the server as well as a summary of their state. This is equivalent to *-c peers.*

`-s`

Print a list of the peers known to the server as well as a summary of their state, but in a slightly different format than the -p switch. This is equivalent to *-c dmpeers.*

`host`

Sets the host to which future queries are sent, as either a hostname or a numeric address. If *host* is omitted, the local host is used.

3.4.3. Interactive Commands

3.4.3.1. Internal Commands

Interactive format commands consist of a keyword followed by zero to four arguments. Only enough characters of the full keyword to uniquely identify the command need be typed. The output of a command is normally sent to the standard output, but you can send the output of individual commands to a file by appending a greater than (>) followed by a filename to the command line.

`?` [*`command-keyword`*]

`help` [*`command-keyword`*]

A question mark (**?**) by itself prints a list of all the known command keywords. A question mark (**?**) followed by a command keyword prints function and usage information.

`delay milliseconds`

Specifies a time interval to be added to timestamps included in requests that require authentication. This is used to enable unreliable server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized.

`host` [*`hostname`*]

Sets the host to which future queries are sent. *Hostname* may be either a hostname or a numeric address.

`hostnames` [yes | no]

If ***`yes`*** is specified, host names are printed in information displays. If ***`no`*** is specified, numeric addresses are printed instead. The default is ***`yes`***, unless modified using the command line **-n** switch.

`keyid` [*`keyid`*]

Allows a key number to be used by NTPDC to authenticate configuration requests. This must correspond to a key number the server has been configured to use for this purpose.

`quit`

Exits NTPDC.

`passwd`

Prompts you to type in a password (which is not echoed) that is used to authenticate configuration requests. The password must correspond to the key configured for use by the NTP server for this purpose if such requests are to be successful.

`timeout` [*`milliseconds`*]

Specifies a timeout period for responses to server queries. The default is approximately 8000 milliseconds.

Note

Since NTPDC retries each query once after a timeout, the total waiting time for a timeout is twice the timeout value set.

3.4.3.2. Control Message Commands

Query commands produce NTP mode 7 packets containing requests for information being sent to the server. These are read-only commands in that they make no modification of the server configuration state.

`listpeers`

Obtains and prints a brief list of the peers for which the server is maintaining state. These should include all configured peer associations, as well as those peers whose stratum is such that they are considered by the server to be possible future synchronization candidates.

`peers`

Obtains a list of peers for which the server is maintaining state, along with a summary of that state. Summary information includes the address of the remote peer; local interface address (0.0.0.0 if a local address has yet to be determined); stratum of the remote peer (a stratum of 16 indicates the remote peer is unsynchronized); polling interval (in seconds); reachability register (in octal); and current estimated delay, offset, and dispersion of the peer (all in seconds).

The character in the left margin indicates the mode this peer entry is operating in as per the table:

+ Symmetric active

- Symmetric passive

= Remote server is being polled in client mode

^ Server is broadcasting to this address

~ Remote peer is sending broadcasts

* Peer the server is currently synchronizing to

The contents of the host field may be in one of four forms: a hostname, IP address, reference clock implementation name with its parameter, or REFCLK (implementation number, parameter). With **hostnames no**, only IP-addresses are displayed.

`dmpeers`

A slightly different peer summary list. Identical to the output of the **peers** command, except for the character in the leftmost column. Characters only appear beside peers which were included in the final stage of the clock selection algorithm. Characters indicate server validity according to the following table:

. peer was cast off in the falseticker detection

+ peer made it through falseticker detection

* Peer the server is currently synchronizing with

`showpeer` *`peer-address`* [...]

Shows a detailed display of the current peer variables for one or more peers. Most of these values are described in the NTP Version 2 specification. Understanding this information will require a detailed understanding of the inner workings of the NTP protocol, which is also available in the RFCs that specify the protocol.

`pstats` *`peer-address`* [...]

Shows per-peer statistic counters associated with the specified peer(s).

`loopinfo` [oneline | multiline]

Prints the values of selected loop filter variables. The loop filter is the part of NTP which deals with adjusting the local system clock.

loop filter is the part of NTP that deals with adjusting the local system clock

offset is the last offset given to the loop filter by the packet processing code

frequency is the frequency error of the local clock in parts per million (ppm)

time_const controls the stiffness of the phase-lock loop and thus the speed at which it can

adapt to oscillator drift

watchdog timer is the number of seconds elapsed since the last sample offset was given to the loop filter

The **oneline** and **multiline** options specify the format in which this information is to be printed, with **multiline** as the default.

`sysinfo`

Prints a variety of system state variables, such as the state related to the local server. All except the last four lines are described in the NTP Version 3 specification, RFC 1305.

The system flags can be set and cleared by the **enable** and **disable** configuration commands, respectively. These are the *auth, bclient, monitor, pll, pps*, and *stats* flags. (See Section 3.3.2 for the meaning of these flags.)

The *stability* is the residual frequency error remaining after the system frequency correction is applied, and is intended for maintenance and debugging. In most architectures, this value initially decreases from as high as 500 ppm to a nominal value in the range .01 to 0.1 ppm. If it remains high for some time after starting the server, something might be wrong with the local clock.

The *broadcastdelay* shows the default broadcast delay, as set by the **broadcastdelay** configuration command. The *authdelay* shows the default authentication delay, as set by the **authdelay** configuration command.

`sysstats`

Prints statistics counters maintained in the protocol module.

`memstats`

Prints statistics counters related to memory allocation code.

`iostats`

Prints statistics counters maintained in the input-output module.

`timerstats`

Prints statistics counters maintained in the timer/event queue support code.

`reslist`

Obtains and prints the server’s restriction list. This list is usually printed in sorted order and may help to understand how the restrictions are applied.

`monlist` [*`version`*]

Obtains and prints traffic counts collected and maintained by the monitor facility. You do not normally need to specify the version number.

3.4.4. Runtime Configuration Requests

All requests that cause state changes in the server are authenticated by the server using the **requestkey** in the configuration file (which can be disabled by the server by not configuring a key). The key number and the corresponding key must also be made known to NTPDC. This can be done using NTPDC's **keyid** and **passwd** commands, the latter of which prompts at the terminal for a password to use as the encryption key. You are also prompted automatically for both the key number and password the first time a command is given that would result in an authenticated request to the server. Authentication not only provides verification that the requester has permission to make such changes, but also gives an extra degree of protection against transmission errors.

Authenticated requests always include a timestamp in the packet data, which is included in the computation of the authentication code. This timestamp is compared by the server to its receive timestamp. If they differ by more than a small amount, the request is rejected. This is done for two reasons. First, it makes simple replay attacks on the server, by someone who might be able to overhear traffic on your LAN, much more difficult. Secondly, it makes it more difficult to request configuration changes to your server from topologically remote hosts. While the reconfiguration facility works well with a server on the local host, and may work adequately between time synchronized hosts on the same LAN, it works very poorly for more distant hosts. As such, if reasonable passwords are chosen, care is taken in the distribution and protection of keys, and appropriate source address restrictions are applied, the run-time reconfiguration facility should provide an adequate level of security.

The following commands all make authenticated requests.

`addpeer` *`peer-address`* [*`keyid`*] [*`version`*] [*`prefer`*]

Adds a configured peer association at the given address and operates in symmetric active mode.

Note

An existing association with the same peer may be deleted when this command is executed, or may simply be converted to conform to the new configuration, as appropriate. If the optional *keyid* is a non-zero integer, all outgoing packets to the remote server have an authentication field attached, encrypted with this key. If the value is 0 (or not given), no authentication is done. The *version* can be 1, 2, 3 or 4, and defaults to 4. The **prefer** keyword indicates a preferred peer (and thus is used primarily for clock synchronization if possible).

`addserver` *`peer-address`* [*`keyid`*] [version] [*`prefer`*]

Identical to the **addpeer** command, except that the operating mode is client.

`broadcast` *`peer-address`* [*`keyid`*] [*`version`*] [*`prefer`*]

Identical to the **addpeer** command, except that the operating mode is **broadcast**. In this case a valid key identifier and key are required. The *peer-address* parameter can be the broadcast address of the local network, or a multicast group address assigned to NTP. If using a multicast address, a multicast-capable kernel is required.

`unconfig` *`peer-address`* [...]

Removes the configured bit from the specified peers. In many cases, this deletes the peer association. When appropriate, however, the association may persist in an unconfigured mode if the remote peer is willing to continue in this fashion.

`enable` [*`flag`*] [...] disable [*`flag`*] [...]

Operates the same as the **enable** and **disable** configuration file commands of NTPD.

`restrict` *`address mask flag`* [*`flag`*]

Operates the same as the **restrict** configuration file commands of NTPD.

`unrestrict` *`address mask flag`* [*`flag`*]

Unrestricts the matching entry from the restrict list.

`delrestrict` *`address mask`* [*`ntpport`*]

Deletes the matching entry from the restrict list.

`readkeys`

Causes the current set of authentication keys to be purged and a new set to be obtained by rereading the keys file (`IP$:NTP.KEYS`). This allows encryption keys to be changed without restarting the server.

`trustedkey` *`keyid`* [...]`untrustedkey` *`keyid`* [...]

Operates the same as the **trustedkey** and **untrustedkey** configuration file commands of NTPD.

`authinfo`

Returns information concerning the authentication module, including known keys and counts of encryptions and decryptions which have been done.

`reset`

Clears the statistics counters in various modules of the server.

3.5. NTP Management

3.5.1. Master Server

The VSI TCP/IP Master_Server process is responsible for starting the **NTPD** server. When the Master_Server process is started, or restarted, it checks the list of enabled servers in the `IP $:SERVICES.MASTER_SERVER` file and starts those that are enabled. To enable NTP use the following commands:

```
$ IP configure/servers
VSI TCP/IP for OpenVMS Server Configuration Utility V10.5 (42)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>enable ntp
SERVER-CONFIG>exit
[Writing configuration to
IP_COMMON_ROOT:[IP]SERVICES.MASTER_SERVER]
```

If you need NTP started immediately, you must restart the master server in addition to the issuing the above commands. Otherwise, NTP will be started the next time VSI TCP/IP is started.

3.5.2. Netcontrol

The VSI TCP/IP **NETCONTROL** command is used to start and stop the NTP_SERVER process. It can also be used to make certain changes to the operation of NTP, or to inquire about specific items of information.

To start the NTP_SERVER process:

```
$ IP netcontrol ntp
Connected to NETCONTROL server on "LOCALHOST"
< pseudo.process.com Network Control V10.5 (10) at Sun 21-Dec-2017 9:58PM-
EST
NTP>start
< Starting NTP server
< NTP server started, process id E9
NTP>quit
```

VSI TCP/IP **NETCONTROL** supports the following NTP-specific commands:

DEBUG	Allows setting of the debug level
NOOP	Sends a null command. Useful for making sure the server is running.
NTP-CONTROL-VERSION	Displays the version of the NTP-CONTROL software.
RELOAD	Causes the NTP_SERVER process to restart.
SHOW	Displays some information about current server status and peers.
SHUTDOWN	Causes the NTP_SERVER process to exit.
START	Causes the NTP_SERVER process to be created.
VERSION	Displays the version info for the NTP server and NTP-CONTROL software.
WAYTOOBIG	Allows setting of the panic/waytoobig value

3.5.3. Monitoring

NTP includes a comprehensive monitoring facility suitable for continuous, long term recording of server and client timekeeping performance. (See the **statistics** configuration option for a listing of each type of statistic currently supported.)

3.5.4. Troubleshooting Tips

Here are some troubleshooting tips:

- Make sure the entries in the NTP configuration file `IP$:NTP.CONF` are correct. At the minimum, there must be a server or peer declaration for a machine that is reachable, and if authentication is enabled, set it up to properly authenticate NTP packets. The machine serving time must be connected either to lower stratum machines or to some reference time source.

- Make sure that the logical `IP$TIMEZONE` is properly defined to reflect the timezone (and daylight savings). If the logical is undefined or incorrect, NTP is likely to abort.

- The `IP$TIMEZONE` logical is defined by VSI TCP/IP when the system starts. If VSI TCP/IP has already been started, issuing the following command will temporarily redefine the logical:

```
IP SET /TIMEZONE=<zone name>/SELECT=<rule name>
```

However, the new value will not be preserved through a system reboot. To permanently change the value of this logical, the timezone rules must be configured using **IP CONFIGURE**. However, **IP CONFIGURE** will not redefine the logical on the running system. If VSI TCP/IP has already been started, you have to do both.

- If using the **slewalways** command, make sure the system time is within 4000 seconds (or whatever ***panic*** is set to) of the correct time before starting NTPD. If the local system time is off by more than this amount from server time, NTPD logs a message and stops running. Also, if the local clock is not within a minute or two of correct time when starting NTPD with slewalways set, it may take some time for NTPD to synchronize the clock. Ideally, set the clock with **SET TIME** before starting NTPD.
- Make sure that DTSS service is not running on the system. This service is used to synchronize time and interferes with NTP.
- The following messages are generated by the NTP server. They may go to either OPCOM or the `IP$:NTPD.LOG` file, or both. This log file is the best source of information for troubleshooting in that it contains a record of these messages as well as additional informational messages. Messages appear in the log file without the bracketed prefix. There are four types of messages generated:
- Configuration messages
- Peer contact messages
- Synchronization messages
- Unexpected error condition messages

Access error messages help by entering

```
$ HELP IP MESSAGES
```

3.5.5. Troubleshooting Using NTPQ

The NTPQ utility has a few commands that are helpful in identifying problems. The **peers** command is one of the simplest and is a quick way to check the offset (time difference) between the local host and peer machines.

The **readvar** command is useful for more in depth information. Without arguments, it displays information about the local host. When **readvar** is followed by an *assocID*, it displays information about the peer corresponding to the *assocID* (use **associations** to display the *assocIDs* for all peers). Of interest is the record of time offsets and round trip delays for packets (the filtoffset and filtdelay fields). This provides a record of the last eight time updates obtained from a peer.

The command readvar *assocID* flash displays a useful variable, flash, which can be of particular interest for troubleshooting. The bits in the flash variable, if set, have the following meaning in relation to a peer:

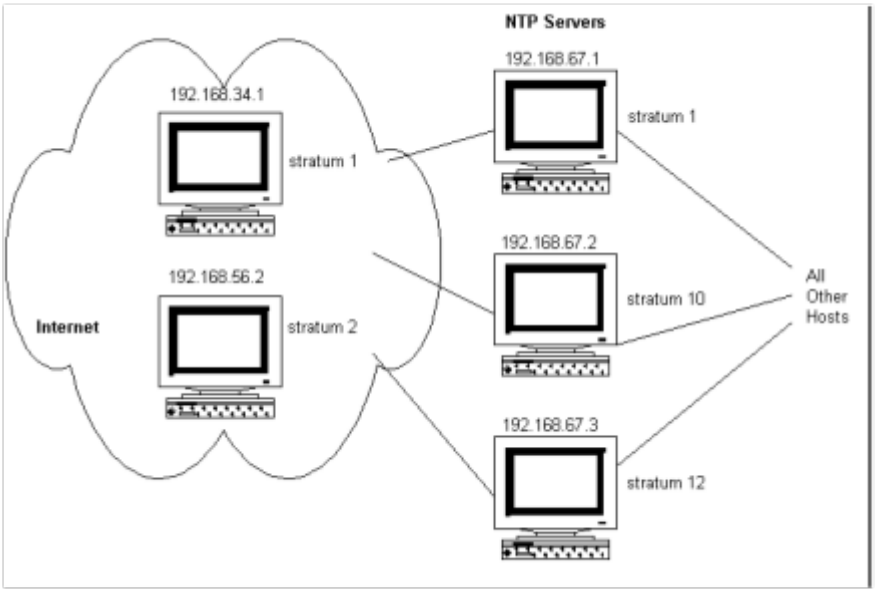
```
0x01 /* duplicate packet received */
0x02 /* bogus packet received */
0x04 /* protocol unsynchronized */
0x08 /* peer delay/dispersion bounds check */
```

```
0x10 /* peer authentication failed */
0x20 /* peer clock unsynchronized */
0x40 /* peer stratum out of bounds */
0x80 /* root delay/dispersion bounds check */
```

3.6. Configuration Example

Figure 3.2 shows a highly redundant and robust configuration with multiple levels of backups. On the Internet close to your network, you have host 192.168.34.1 running at stratum 1, and 192.168.34.2 at stratum 2. In-house, you have host 192.168.67.1 synchronized with a radio clock and configured as a stratum 1 master clock.

Figure 3.2. Sample NTP Configuration



As backup servers, you have two hosts, 192.168.67.2 and 192.168.67.3, in the climate-controlled room, one configured at stratum 10 and the other at 12. All other workstations on the floor point to these three servers as their synchronization source. When everything is running, every local host is synchronized to 192.168.67.1, since it is closer than Internet host 192.168.34.1. All the machines (peers) run at stratum 2.

If internal host 192.168.67.1 goes down and the Internet connection is still up, either Internet host 192.168.34.1 or 192.168.34.2 is selected depending on its availability, and the backup servers, 192.168.67.2 and 192.168.67.3, run at stratum 2 or 3, depending on which Internet host was selected. The peers synchronize off 192.168.67.2 or 192.168.67.3 at stratum 3 or 4, again depending on which Internet host was selected.

With 192.168.67.1 still unavailable and the Internet connection lost or all the Internet servers unavailable, 192.168.67.2 runs at stratum 10, since it was configured that way as a local clock. It then becomes the lowest stratum number in the network and all other hosts (including 192.168.67.3) are synchronized to it at stratum 11.

If 192.168.67.2 goes down, 192.168.67.3 runs at stratum 12 and all other hosts synchronize at stratum 13. It is important to set the stratum of 192.168.67.3 to 12. If set to 11, it might have a problem synchronizing to 192.168.67.2, since it may try to synchronize off it but finds it has the same stratum value. 192.168.67.3 would rather synchronize to 192.168.67.2 than to itself.

The following example shows the configuration file entries for each of the three local servers (the other local hosts would all be configured as peers). You do not need to explicitly identify the peer strata, and the order of items is irrelevant.

Example 3.1. Sample Entries in the Host NTP.CONF Files

```
; NTP Configuration on 192.168.67.1
; NOTE: server and peer specifications can have -4 or -6 options added
; to force use of IPv6 or IPv4 when DNS names for both exist.
; Example:
; server -4 ticktock.timesrc.bar

master-clock 1

; NTP Configuration on 192.168.67.2
local-master 10
server 192.168.67.1
server 192.168.34.1
server 192.168.34.2
peer 192.168.67.3


; NTP Configuration on 192.168.67.3
local-master 12
server 192.168.67.1
server 192.168.34.1
server 192.168.34.2
peer 192.168.67.2


; NTP Configuration for Computer Room Host 192.168.67.x
server 192.168.67.1
server 192.168.67.2
server 192.168.67.3
peer 192.168.67.y
peer 192.168.67.z
```

Chapter 4. Configuring Electronic Mail

This chapter describes how to configure the VSI TCP/IP SMTP (Simple Mail Transport Protocol) server to send and receive electronic mail.

If you are running Process Software *PMDF* or another mail system that provides its own SMTP support, refer to that mail system's documentation.

4.1. Modifying the VSI TCP/IP SMTP Configuration File

The VSI TCP/IP SMTP configuration is stored in the `START_SMTP.COM` and `START_SMTP_LOCAL.COM` startup command procedures. Use the MAIL-CONFIG utility to edit these files. You configure the utility by entering the **IP CONFIGURE /MAIL** command. After using this configuration utility, stop and restart the mail queues. Enter:

- **@IP$:START_SMTP.COM** to update the OpenVMScluster.
- **@IP$:START_SMTP_LOCAL.COM** to update the local host only.

4.1.1. Pipelining and Extended SMTP

The current release of SMTP implements Extended SMTP (RFC-1869) and Pipelining (RFC-2197).

4.1.2. Delivering Mail to Specific Folders

The SMTP server supports mail delivery to folders other than the NEWMAIL folder. The foldernames are restricted to UPPERCASE characters only, the pound sign (#), and the underscore (_). Use of the comma (,) in a foldername causes an error. Mail addressed to *`user+folder@host`* is delivered to the specified *`folder`*. You can disable this mechanism by defining the system-wide logical name `IP $SMTP_DISABLE_FOLDER_DELIVERY`.

4.1.3. Using the Mail Delivery Mechanisms

This release of SMTP supports alias file extensions that request mail delivery to a file or specify addresses in a separate file. You must use the SMTP aliases file, specified with **IP CONFIG/MAIL**, to list all of these mail delivery mechanisms. The default is `IP$:SMTP_ALIASES`. The syntax for these aliases follows the form of those described in Section 4.3.11.16 found later in this chapter. It is necessary to use the colon and semicolon in the command lines as shown in the examples.

Command	Description
<device:[directory]address.list	Delivers mail to the list of addresses in the specified file. `alias1 : "<filespec" ;`
| device:[directory]procedure.com parameter(s)	Submits the specified command procedure to the queue identified by the logical name `IP $SMTP_BATCH_QUEUE, SYS$BATCH` by default. The first parameter (P1), passed to the

Command	Description
	submitted procedure, is always the name of a temporary file containing the mail message that the procedure must delete. Any *parameter(s)* specified in the alias file are passed to the submitted procedure in a single string as its second parameter (P2). `alias2 : "|filespec p2 p3" ;`
>device:[directory]mail.file	Appends mail to the specified file. If no filetype is specified, the default is *.yyyy-mm*, • *yyyy* and *mm* are to the current year and month, respectively. `alias3 : ">filespec" ;`

4.1.4. Rejecting Mail Messages

The SMTP server supports a set of rules for rejecting mail messages received by itself based on the mail header contents or any combination of **MAIL FROM**, RCPT TO, and Source IP Address values. Mail matching the criteria can be ignored or rejected quietly with a message to the SMTP client or delivered to an address rewritten according to the rule specification. This capability can be useful for controlling SPAM and preventing your system from being used as a mail relay.

The file `IP$:SMTP_SERVER_REJECT`. contains the rejection and rewrite rules. You may specify an alternate file via the logical name `IP$SMTP_SERVER_REJECT_FILE`. A rejection file line of the form **#include***device:[directory]reject.file* temporarily suspends processing of the current file and begins processing of the specified file. Rejection files can be nested to arbitrary depth. Comments may be included in rejection files by placing any of the characters ; or ! or # in the first column of a line. The following is a sample rejection file:

```
!
! This is a sample reject file for the company FLOWERS.COM.
!
! This file is processed sequentially. In other words, processing ends on
! the first rule that the message applies to. So if you have a wildcard
! accept at the top of this file, then no other rules will be processed.
!
! Entries can have one of the following formats:
!
!   from_user [from_ip to_user action action-data]
!
!   :rfc822 header
!
! Wildcards can be used in FROM_USER, FROM_IP, and TO_USER. ACTION is the
! reject action, which is one of:
!
!   n   Don't reject, but rewrite TO address to be ACTION-DATA.
!       If ACTION-DATA is blank then we simply deliver to TO_USER.
!
!   y   Reject and use optional ACTION-FIELD as a rejection message
!       format that can contain up to three %s formatting
!       designators for mail from, mail to, and local domain name.
!
```

```
!   q   Reject quietly -- don't inform Sending SMTP Client that
!       message will be discarded. If only FROM_USER is specified
!       other fields default to FROM_IP=*, TO_USER=*, and ACTION=n.
!
! Don't rewrite or reject any mail to "postmaster*"
!
!       * * postmaster* n
!
! Accept all messages with MAIL FROM:<> (bounce messages)
! This rule is commented out because you probably don't want it, although
! We're _supposed_ to always accept it. This is the main method relay
! attacks use, by always saying they are from <> to take advantage of that
! RFC hole.
!
! <> * * n
!
! Reject anything with a Message-ID that appears to have originated from
! cyberpromo.com or nowhere.com
!
:Message-ID: <*@cyberpromo.com>
:Message-ID: <*@nowhere.com>
!
! Reject mail from well-known SPAM sites with sample non-standard error
! messages.
!
<*answerme.com>  * * y "Spam from <%s> rejected"
<*cyberpromo.com> * * y "Spam from <%s> to <%s> rejected"
<*pleaseread.com> * * y "Spam rejected;%.0s%.0s Contact postmaster@%s"
!
! Disallow percent-hacks (e.g, joe%somewhere.com@flowers.com)
! *     * *@*@*flowers.com   y   "No forwarding-path relaying allowed"
!
! Disallow "!" UUCP hacks (e.g. somewhere.com!joe@flowers.com)
! *     * *!*               y   "No UUCP relaying allowed"
!
! Rewrite all mail to webmaster to the postmaster
! *  * webmaster@*flowers.com n  postmaster@flowers.com
!
!
! Disallow relaying through our mailer, and only allow users on our
! networks to claim to be from our company (flowers.com)
!*       * *@flowers.com n
 *       * *@daisy.flowers.com n
 *       * *@[10.0.0.1] n
!
<*flowers.com> 10.0.0.*          * n
<*flowers.com> 10.115.140.*      * n
<*flowers.com> 10.115.141.*      * n
!
! Allow our internal systems to bounce mail out.
!
<>             10.0.0.*          * n
<>             10.115.140.*      * n
<>             10.115.141.*      * n
!
! If a message has slipped through all the tests above, then we want to
! reject it, as they are either relaying through us or it's not a valid
! MAIL FROM.
```

```
!
*@*    * *@*   y       "no relaying through this site"
* *    *@* y           "missing domain name in MAIL FROM"
!
!end of sample file
```

Mail rejection rules have two formats:

- :RFC822_header *pattern*

  This format causes rejection of any mail in which a line with the specified header matches the given *pattern*. The following rejection message is sent to the client:

  ```
  554 Message rejected due to header contents
  ```

Note

Use caution when rejecting mail based on header contents. No other criteria are considered during rejection processing.

- from_user ip_address to_user *action action_data*

  This format causes rejection or alternate delivery of all messages that match all of the patterns specified. The *action* item can be as follows:

Parameter	Description
n	Means do not reject the mail, but deliver it to the address specified as the *action_data*. If *action_data* is not specified, deliver the message to its intended recipient.

y	Means reject the mail, sending the *action_data* string to the SMTP client as a rejection message. The *action_data* item is actually used as a format string and may contain from one to three %s formatting designators to include the *from_user*, the *to_user*, and the SMTP server name, **in that order**. If *action_data* is missing, the default rejection message is
	
	`553-Mail to <to_user> not allowed;`
	`553 Contact Postmaster@<smtpserver> to remove block`

q	Means reject the mail, but do not give the SMTP client any indication that it has been rejected. Use caution when rejecting messages quietly.

Each of the pattern specifications *pattern*, *from_user*, *ip_address*, and *to_user* may contain the OpenVMS ***** and **%** wildcard characters.

You can represent from_addr expressions in the SMTP_SERVER_REJECT filter with the <> syntax. So, `*@*domain.com` and `<*@*domain.com>` are the same expression. To return to the previous behavior, add the following line to the top of your SMTP_SERVER_REJECT file:

```
<> * * n
(Accept any mail with a MAIL FROM: of <>)
```

When comparing the RCPT TO: address with the SMTP_SERVER_REJECT file expressions, any '%' signs in the RCPT TO: address are changed to '@'. You can write filter rules in the SMTP_SERVER_REJECT files that can match against forward-path relays. You can add the rule of

```
* * *@*@localdomain y “No forward-path relaying allowed”
```

to your SMTP_SERVER_REJECT file above the rules that accept mail with the destination of your domains. RCPT TO.: addresses will replace any % character with the @ character for matching purposes only so you can filter with *@*@*-type rules. So, RCPT TO.:<xxx%yyy@zzz> is changed to xxx@yyy@zzz. You can use the logical name IP$SMTP_SERVER_REJECT_INFO to control debug and informational OPCOM messages produced during rejection processing. You should define it to have some non-zero value to request OPCOM messages. The following values may be combined to control message quantity and content:

Values	To show...
1	mail rejected due to *action* **y**
2	rewritten addresses (*action* **n** with *action_data*)
4	the reject message sent to the remote system
8	configuration file parsing
16	non-written addresses (*action* **n** and no *action_data*)
32	mail rejected due to *action* **q**
64	mail rejected due to header rules

The value 65 is appropriate for auditing rejection activity.

The SMTP service supports both IPv4 and IPv6. There are no configuration parameters to control which one is used. System managers should be aware of potential connection issues when switching SMTP to use IPv6.

The remainder of this chapter describes the configuration tasks.

4.2. SMTP Statistics and Accounting

The following sections discuss how to get SMTP statistics and accounting information.

4.2.1. Network Service Monitoring

Partial support for RFC 2788 (Network Monitoring MIB) has been added to SMTP. To use the 2788 feature, do the following:

```
$ IP CONFIGURE/SERVER
 SELECT SMTP
 SET FLAGS SNMP_MONITORED
 WRITE
 EXIT
$ @IP$:START_SERVER
```

This feature requires the SNMP Agent X functionality; to use this SNMP must be configured to have Agent X service enabled, and to allow 127.0.0.1 to be an AGENTX_PEER. For more information on SNMP and Agent X, see the *VSI TCP/IP Administrator's Guide: Volume II*.

SMTP’s network service monitoring is based on RFC 2788 (Monitoring MIB). Information is maintained only while the service is active. The following items from the Network Services Monitoring MIB (RFC 2788) are available in the enterprises.105.2.25 MIB:

Table 4.1. RFC2788 Network Services Monitoring MIB

ApplAccumulatedInboundAssociations	(Counter) the total number of connections that the SMTP program has serviced since it was started. enterprises.105.2.25.10
ApplDescription	(String) Description of the program/application. enterprises.105.2.25.16
ApplInboundAssociations	(Counter) The number of connections currently active. enterprises.105.2.25.8
ApplIndex	(Integer) unique application index. The port SMTP is offered on (25). enterprises.105.2.25.1
ApplLastChange	(TimeTicks) the value of sysUpTime when the SMTP program entered the current state. enterprises.105.2.25.7
ApplLastInboundActivity	(TimeTicks) the value of sysUpTime at the time the most recent connection was established. enterprises.105.2.25.12
ApplName	(String) SMTP. enterprises.105.2.25.2
ApplOperStatus	(Integer) the operational status of the SMTP program; the values are: up(1), down(2), halted(3), congested(4), restarting(5), quiescing(6). Some of these values may not be used. enterprises.105.2.25.6
ApplRejectedInboundAssociations	(Counter) the number of connections that have been rejected (due to not being allowed from the access list values). enterprises.105.2.25.14
ApplUptime	(TimeTicks) the value of the SNMP variable sysUpTime when the SMTP program was started. enterprises.105.2.25.5
ApplVersion	(String) the version of the SMTP program. enterprises.105.2.25.4 **Note** When displaying the enterprises.105.2.25 MIB, entries for 1 to 17 will display but some (specifically .3, .9, .11, .13, .15, .17) will have 0 values.

4.2.2. Mail Monitoring

Partial support for RFC 2789 (Mail Monitoring MIB) has been added to SMTP. To enable this feature, do the following:

```
$ IP CONFIGURE/MAIL
 SET RFC2789 TRUE
 WRITE
 EXIT
$ @IP$:START_SMTP
```

This feature requires the SNMP Agent X functionality; to use this SNMP must be configured to have Agent X service enabled, and to allow 127.0.0.1 and the system's own IP address to be an

AGENTX_PEER. See the *VSI TCP/IP Administrator's Guide: Volume II* for more information on SNMP and Agent X.

Table 4.2. RFC2789 Mail Monitoring MIB

Parameter	Description
MtaReceivedMessages	The number of messages received since Message Transfer Agent (MTA) initialization. enterprises.105.3.25.1
MtaStoredMessages	The total number of messages currently stored in the MTA. enterprises.105.3.25.2
MtaTransmittedMessages	The number of messages transmitted since MTA initialization. enterprises.105.3.25.3
MtaReceivedVolume	The total volume of messages received since MTA initialization, measured in kilo-octets. enterprises.105.3.25.4
MtaStoredVolume	The total volume of messages currently stored in the MTA, measured in kilo-octets. enterprises.105.3.25.5
MtaTransmittedVolume	The total volume of messages transmitted since MTA initialization, measured in kilo-octets. enterprises.105.3.25.6
MtaReceivedRecipients	The total number of recipients specified in all messages received since MTA initialization. enterprises.105.3.25.7
MtaStoredRecipients	The total number of recipients specified in all messages currently stored in the MTA. enterprises.105.3.25.8
MtaTransmittedRecipients	The total number of recipients specified in all messages transmitted since MTA initialization. enterprises.105.3.25.9
MtaSuccessfulConvertedMessages	The number of messages that have been successfully converted from one form to another since MTA initialization. enterprises.105.3.25.10
MtaFailedConvertedMessages	The number of messages for which an unsuccessful attempt was made to convert them from one form to another since MTA initialization. enterprises.105.3.25.11

This information can be displayed with the **IP SHOW /SNMP** command. See the **SHOW /SNMP** command in the *VSI TCP/IP Administrator's Reference*.

4.2.3. Session Accounting

VSI TCP/IP can record accounting information from services that have been enabled. Currently this includes FTP and SMTP. The accounting information includes information about when a network session took place and how much data was transferred. The accounting facility is enabled from **IP CONFIGURE/SERVER** and reads `IP$:ACCOUNTING.CONF` for additional configuration information. The format of the accounting records is described in `IP$ROOT:[IP.EXAMPLES]ACCOUNTING.H`.

A sample program using this is in `IP$ROOT:[IP.EXAMPLES]ACC_DUMP.C`.

4.2.4. Configuring Session Accounting

To configure Session Accounting, follow these steps:

1. Edit the `ACCOUNTING` configuration file, as described in Section 4.3.
2. To start the procedure, do the following:

```
$ IP CONFIGURE/SERVER
 ENABLE ACCOUNTING
 WRITE
$ @IP$:START_SERVER
```

4.3. Configuration File

The Accounting configuration file is `IP$:ACCOUNTING_CONF.TEMPLATE`. The Accounting configuration file defines:

- The Port the Accounting program listens on. This should be an unused port, not the port for the service on which logging is being enabled.
- The name of the file used for accounting records. This file is opened shareable and new records are always appended to it. To start a new file stop the Accounting program, delete (or rename) the existing file, and restart the Accounting program.
- The IP addresses of systems that are allowed to write accounting records to this host.

Note

After editing the configuration, stop and restart the Accounting program.

4.3.1. File Format

Follow these guidelines when entering data in the Accounting configuration file:

- Allow one line for each item.
- Enter information in any order; in uppercase or lowercase.

- Use a pound sign (#) or exclamation point (!) to denote comments. The Accounting facility ignores all information following these characters.

The commands that can be in `IP$:ACCOUNTING.CONF` are:

Command	Description
`PORT` *`port_number`*	The TCP port that the accounting program should listen on.
`PEER` *`ip-address`*	The IP address of a host that is allowed to log records with the accounting software.
`FILENAME` *`filename`*	The name of the file that the accounting records will be written to. The IP$: device is assumed if a device is not specified as part of the file specification.

4.3.2. Displaying the Contents of the Logging File

To view accounting information, do the following:

```
$ IP ACCOUNTING/INPUT=accounting_data_file [/output=output filename] -
_$ [/since=start_date] [/before=end_date] [/protocol={SMTP, FTP, MAIL}] [/
CSV]
```

- ***`accounting_data_file`*** is the name of the logging file you want to see.
- ***`output filename`*** is the name of the file you want to call this information. If this field is omitted, the information displays to the terminal screen.
- ***`start_date`*** is the beginning date you want the command to start with. The date format is **[DD-MMM-YYYY [:]] [hh:mm:ss]cc]**

  If not specified, all records display up to the end of the data found.

  - **DD** is the day of the month, counting from 01.
  - **MMM** is the abbreviation for the month, like JAN, FEB, MAR.
  - **YYYY** is the number of the year, including the century (2013, 2014, 2015, 2016, 2017).
  - **hh** is the hour, from 00 to 23.
  - **mm** is the minute, from 00 to 59.
  - **ss** is the second, from 00 to 59.
  - **cc** is hundredths of seconds.

  The time is always in local time.
- ***`end_date`*** is the ending date you want the command to end with. The date format is **[DD-MMM-YYYY [:]] [hh:mm:ss]cc]**

  If not specified, all records display from the start date until the end of the file.
- **protocol** is any combination of SMTP, FTP, or MAIL.
- **CSV** is the Comma Separated Values, for input to products.

4.3.2.1. Accounting File Record Format

The accounting file is written using OpenVMS RMS records. The format of these records is defined in `IP$ROOT:[IP.EXAMPLES]ACCOUNTING.H`, and listed below:

```
struct accountingPDU {
 char version;
 char type;       /* type of record */
/*
 * FTP:
 *   C - Client
 *   S - Server
 *
 * SMTP:
 *   N - Network delivery (send)
 *   L - Local delivery (received)
 *   R - Rejected message
 *
 */
 char flags;      /* not currently used */
 char reserved;     /* for future use */
 int payload_length;  /* length (in bytes) of data after header */
 int port;       /* IP port of reporting service - 25 SMTP, 21 - FTP */
 int reporterIP;    /* IP address of reporter */
};
struct FTPaccounting_data {
 struct accountingPDU header;
 int start_time[2];  /* OpenVMS time that session started */
 int end_time[2];   /* OpenVMS time that session ended */
 int datasent;     /* KBytes of file data sent */
 int datarecv;     /* KBytes of file data received */
 int filessent;    /* Number of files sent */
 int filesrecv;    /* Number of files received */
 int partnerIP;    /* IP address of partner */
 char user[12];     /* username that operations were done under */
};

struct SMTPaccounting_data {
 struct accountingPDU header;
 int date[2];     /* Time of activity */
 int msg_size;     /* size of message in bytes */
 int from_str_size;  /* size of From: string */
 int to_str_size;   /* size of To: string */
 char from_to_str[1];  /* text of From & To string */
};
```

4.3.3. Configuring SMTP for Accounting

To configure SMTP for accounting purposes, do the following:

```
$ IP CONFIGURE/MAIL
 SET ACCOUNTING-HOST hostname
 SET ACCOUNTING-PORT port number
 WRITE
 EXIT
$ @IP$:START_SMTP
```

The collected accounting information can be displayed with the **IP ACCOUNTING** command.

See the **IP ACCOUNTING** command in the *VSI TCP/IP Administrator's Reference*.

4.3.4. Configuring Mail Parameters

The parameters that control the operations of the VSI TCP/IP mailer are described in Table 4.3.

4.3.4.1. Configuring Mail Parameters with MAIL-CONFIG

To configure mail parameters with the MAIL-CONFIG utility:

1. Start MAIL-CONFIG with the **IP CONFIGURE /MAIL** command.
2. Use the **SET *parameter_name*** commands (for detailed descriptions of these commands, refer to the *VSI TCP/IP Administrator's Reference*).
3. Save the configuration with the **SAVE** command.
4. Quit MAIL-CONFIG with the **QUIT** command or exit.

The modified configuration takes effect the next time your system reboots or the queues are restarted via IP$:START_SMTP.

4.3.4.2. Mail Parameters

Table 4.3 describes all the mail parameters you can set with the MAIL-CONFIG utility.

Table 4.3. Mail Parameters

Parameter	Description
ALIAS-FILE	File in which SMTP aliases are stored; see Section 4.3.11.16.
DECNET-DOMAIN	Domain name for DECnet gateway function; see Section 4.3.14.
DELIVERY-RECEIPTS	Specifies whether mail receipts are sent when incoming mail containing Delivery-Receipt-To: or Return-Receipt-To: headers is submitted to the SMTP queue. If TRUE, mail receipts are sent.
DISALLOW-USER-REPLY-TO	When TRUE, prevents OpenVMS MAIL users from setting a Reply-To: header address with the logical name IP$SMTP_REPLY_TO.
FORWARDER	Identifies a host (known as a *mail hub*) to which mail should be forwarded if a host name cannot be resolved for an address. The specified name must resolve to an IP address; it must not resolve to an MX record.
FORWARD-LOCAL-MAIL	Forwards all mail designated for local users to the mail hub instead of delivering it locally. Can be overridden by entries in the SMTP_ALIASES file.
FORWARD-REMOTE-MAIL	Forwards all SMTP-delivered mail to the mail hub instead of directly to the destination host. Can be overridden by a GATEWAY or LOCAL-DOMAIN entry.
HEADER-CONTROL	Controls which RFC-822 headers appear in messages delivered to OpenVMS MAIL users.
HOST-ALIAS-FILE	Contains a list of host names considered aliases for the local host name.
LOCAL-MAIL-FORWARDER	Identifies a host to which mail should be forwarded when a local mail delivery fails because the user name is unknown.

Parameter	Description
POSTMASTER	Identifies the user name of the system postmaster.
QUEUE-COUNT	Specifies the number of mail processing queues to create on a particular system.
REPLY-CONTROL	Specifies how Internet mail headers should be mapped to the OpenVMS MAIL from header. Permitted values are FROM and REPLY-TO. You may specify both as a comma-separated list.
RESENT-HEADERS	When FALSE, the SMTP symbiont omits the Resent-From, Resent-To, and Resent-Date headers that are usually included when a message is forwarded using a OpenVMS MAIL forwarding address.
RETRY-INTERVAL	Specifies the amount of time (in minutes) that should elapse after a failed attempt before another attempt is made.
RETURN-INTERVAL	Specifies the amount of time (in hours) a message can remain in the processing queue before it is returned to sender.
SEND-BROADCAST-CLASS	Controls the OpenVMS broadcast class used by SMTP for SEND-type messages (which are sent to a terminal).
SMTP-HOST-NAMES	A list of up to 16 host names to consider as aliases for the local host. See Section 4.3.11.13.
START-QUEUE-MANAGER	Determines whether START_SMTP.COM starts the OpenVMS queue manager if it is not already running.

To configure mail parameters via logical name, see the *VSI TCP/IP Administrator's Reference*.

4.3.5. SMTP Configuration Using Logicals

When using a logical name to configure mail parameters, if the setting is for all users, define the logical system-wide. For example:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$SPOOL device:[PATH.DIRECTORY]
```

4.3.6. SMTP SYMBIONT LOGICAL

An SMTP SYMBIONT logical may be set up to enable additional logging for debugging purposes.

For example:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$SMTP_SYMBIONT_LOG TRUE
```

will enable logging. The default logging filename and location is IP$:IP $SMTP_LOG.*queuename*. If you wish to change the location and/or filename, you may define:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$SMTP_LOG filespec
```

where filespec is similar to DEVICE:[*dir*]*filename*. If the *filename* does not specify an extension, then the *queuename* will be utilized.

4.3.7. MIME processing

If the logical IP$SMTP_ALLOW_MIME_SEND is defined to Yes, 1 or True, then if the first line of the message file being sent begins with the mime tag, the blank line at the end of the header section

will be suppressed so that the header lines in the mime message file will be seen as header lines rather than message body. The string that is used as the mime tag can be controlled with the logical `IP $SMTP_MIME_TAG` which defaults to “Mime-version:”

4.3.8. Mail Outbound Sanity Checking

Outbound mail sanity checking can be used to test the operation type of the mail message before it is sent out. If there is no operation type, the file is not sent. Use this logical to disable sanity checking:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$SMTP_DISABLE_OPTYPE_SANITY_CHECK
```

4.3.9. Configuring the SMTP Server for Inbound Mail

The VSI TCP/IP SMTP server accepts mail from remote hosts and delivers it to users' mailboxes.

By default, the SMTP server is disabled when you install VSI TCP/IP. For details on configuring and controlling VSI TCP/IP servers, see the *VSI TCP/IP Administrator’s Guide: Volume II.*

4.3.9.1. Translating UNIX-Style Linefeeds to SMTP-Compliant End-of-Line Character Sequences

VSI TCP/IP provides a logical name to solve the problem of systems sending messages containing lines terminated by an LF character only instead of the proper CR/LF sequence. The following command tells VSI TCP/IP to accept the bare LF as the end-of-line indicator:

```
$ DEFINE/SYSTEM/EXEC IP$SMTP_ACCEPT_UNIX_LF TRUE
```

VSI TCP/IP lets you validate the contents of the envelope-from field by defining the system-wide logical name `IP$SMTP_REJECT_INVALID_DOMAINS`. Use the equivalence string STRICT to require the presence of a host in those addresses. For example, require MAIL FROM: *`user@host`* rather than MAIL FROM: *`user`*. The host specified in the MAIL FROM: address must exist in the DNS database.

The logical name `IP$SMTP_ACCEPT_UNIX_LF` has been added as a synonym for `IP $SMTP_ACCEPT_UNIX_LF_BRAIN_DAMAGE`. You can define either to have the same effect.

4.3.10. Configuring the SMTP Server to Limit System/Vendor Information

VSI TCP/IP provides you with a way to limit the system/vendor information given out on connection, **HELP**, and **QUIT**. The `IP$SMTP_SUPPRESS_VENDOR` logical removes operating system and TCP stack information from SMTP server text responses.

4.3.11. Configuring the SMTP Symbiont and Mail Queues for Outbound Mail

VSI TCP/IP lets users send mail to remote destinations by submitting outbound messages to mail queues that are processed by the VSI TCP/IP SMTP symbiont. You can configure the SMTP symbiont to:

- Control users' ability to specify their own REPLY-To: headers (see Section 4.3.11.1).

- Provide more than one server queue for each cluster node. By default, VSI TCP/IP provides one server queue for each cluster node running VSI TCP/IP (see Section 4.3.11.3).
- Forward mail through a central mail hub (see Section 4.3.11.6).
- Use gateways to reach specific hosts, domains, or "virtual" domains (see Section 4.3.11.12).
- Use host aliases (see Section 4.3.11.13).
- Use mail aliases (see Section 4.3.11.16).

You can also write your own SMTP dispatcher by modifying and compiling the SMTP user exit `IP.EXAMPLES:USER_SMTP_DISPATCH.C`. Instructions for modifying the dispatcher are outside the scope of this manual.

For outbound mail, VSI TCP/IP SMTP eases the 255 character limitation on RFC-822 To: and CC: header lengths. The limit of 255 characters was imposed because some mail applications cannot handle headers longer than 255 characters.

The default header length is 1024 characters. The logical name `IP $SMTP_MAXIMUM_822_TO_LENGTH` can be used to override the 1024 byte default length of the To: and Cc: header fields. The logical can set the maximum length to anywhere from 256 to 65535. To automatically lower the case of usernames in outbound messages, define the logical name `IP $VMSMAIL_LOCASE_USERNAME`.

4.3.11.1. Specifying the REPLY_TO Header

The `IP$SMTP_REPLY_TO` logical name lets you specify the value for the RFC822 REPLY-TO: header. For example, to set your Reply-To: header to `FNORD@FLOWERS.COM`, use the command:

```
$ DEFINE IP$SMTP_REPLY_TO FNORD@FLOWERS.COM
```

This logical name only affects mail agents that use the SMTP% interface (for example, OpenVMS and DECwindows mail). The system manager can disable the use of this logical name with the **SET DISALLOW-USER-REPLY-TO** command of the **IP CONFIGURE /MAIL** utility.

4.3.11.2. Disabling VRFY and EXPN

To disable VRFY and EXPN processing, use the logical name `IP $SMTP_SERVER_DISABLE_VRFYEXPN`. Define it to have some non-zero value to disable the requisite functions. The following values may be combined to specify which function:

Value	Function
1	to disable VRFY
2	to disable EXPN
3	to disable both VRFY and EXPN

4.3.11.3. Configuring Mail Queues

VSI TCP/IP uses OpenVMS server queues for SMTP processing. Initially, VSI TCP/IP configures each cluster node running VSI TCP/IP with a server queue and configures a generic queue for the

entire cluster. New messages are placed in the generic queue for processing, which distributes mail processing to the first available server queue.

For example, if three clustered nodes, Huey, Louie, and Dewey, are running VSI TCP/IP, VSI TCP/IP creates three server queues and one generic queue. The queue names are:

```
SMTP_HUEY   [Execution queue]
SMTP_LOUIE  [Execution queue]
SMTP_DEWEY  [Execution queue]
IP$SMTP     [Generic queue]
```

The following example lists the queues for node Huey:

```
$ SHOW QUEUE IP$SMTP/FULL
Generic server queue IP$SMTP
 /GENERIC=(SMTP_HUEY,SMTP_LOUIE,SMTP_DEWEY) /OWNER=[SYSTEM]
 /PROTECTION=(codes)

$ SHOW QUEUE SMTP_HUEY/FULL
Server queue SMTP_HUEY, idle, on HUEY::, mounted form DEFAULT
 /BASE_PRIORITY=4 /DEFAULT=(FEED,FORM=DEFAULT) /OWNER=[SYSTEM]
 /PROCESSOR=IP$SMTP_SYMBIONT /PROTECTION=(codes)
```

The queues SMTP_ LOUIE and SMTP_DEWEY are also created, and are similar to the SMTP_HUEY queue shown.

Note

A standalone (non-clustered machine) has just two queues created by default; that is, one generic queue (IP$SMTP) and one execution queue (SMTP_ *nodename*).

4.3.11.4. Configuring Multiple Queues

If mail traffic is heavy on your system, you can configure multiple server queues on one or more nodes using MAIL-CONFIG. To configure multiple queues with the MAIL-CONFIG utility:

1. Start MAIL-CONFIG with the **IP CONFIGURE /MAIL** command.
2. Use the **SET QUEUE-COUNT** command to specify the number of queues on the node (for a full description of this command, refer to the *VSI TCP/IP Administrator's Reference*).
3. Save the configuration with the **SAVE** command.
4. Quit MAIL-CONFIG with the **QUIT** command.

The modified configuration takes effect the next time your system reboots.

4.3.11.5. Configuring Queue Groups

In a cluster environment, you may need to partition mail processing by grouping subsets of your cluster into queue groups using MAIL-CONFIG.

1. Start MAIL-CONFIG with the **IP CONFIGURE /MAIL** command.
2. Use the **ADD QUEUE-GROUP** and **DELETE QUEUE-GROUP** commands to add or delete queues (for descriptions of these commands, refer to the *VSI TCP/IP Administrator's Reference*).

3. Save the configuration with the **SAVE** command.

4. Quit MAIL-CONFIG with the **QUIT** command.

The modified configuration takes effect after restarting SMTP. To restart, enter the following command:

```
ip%:@ip$start_smtp
```

4.3.11.6. Forwarding Mail through a Mail Hub

Many sites provide outbound e-mail access to the Internet through a single system known as a *mail hub* to deliver all outbound mail on behalf of the other hosts at the site. A mail hub typically implements a single-address scheme for e-mail users at the site, so that all users have addresses of the form *username@sitename* rather than *username@hostname.sitename*. Site administrators often configure mail hubs to provide Internet e-mail access to hosts that do not have direct access to the Internet. To forward mail through a mail hub:

1. Specify the host that will serve as a mail hub.

2. Specify the conditions under which VSI TCP/IP forwards mail to the mail hub.

4.3.11.7. Specifying a Mail Hub

To specify the host that will serve as a mail hub for your VSI TCP/IP host:

1. Start MAIL-CONFIG (**IP CONFIGURE /MAIL**).

2. Modify the *FORWARDER* parameter.

   With MAIL-CONFIG, use the **SET FORWARDER *mailhub_hostname*** command.

3. If desired, set any of the following conditions for forwarding mail to the mail hub:

   - Forward mail addressed to users on remote hosts (see Section 4.3.11.8).
   - Exclude hosts in specific domains from remote mail hub forwarding (see Section 4.3.11.9).
   - Forward mail addressed to users on the local host (see Section 4.3.11.10).
   - Exclude specific local users from mail hub forwarding (see Section 4.3.11.11).

4. Exit the configuration utility. When prompted, save the new parameters.

5. To make the changes take effect immediately, stop and restart the mail queues. To update the OpenVMScluster, use the **@IP$:START_SMTP.COM** command. To update the local host only, use the **@IP$:START_SMTP_LOCAL.COM** command. Otherwise, your changes take effect the next time you reboot your system.

4.3.11.8. Forwarding Mail Addressed to Remote Hosts

To configure VSI TCP/IP to forward mail addressed to remote users via a mail hub:

1. Make sure the *FORWARDER* parameter specifies the host you want to use as a mail hub (see Section 4.3.11.7).

2. Start MAIL-CONFIG (**IP CONFIGURE /MAIL**).

3. Modify the F*ORWARD-REMOTE-MAIL* parameter.

   With MAIL-CONFIG, use the **SET FORWARD-REMOTE-MAIL TRUE** command.

4. If desired, exclude hosts in specific domains from mail hub forwarding (see Section 4.3.11.11).

5. If desired, specify other conditions under which VSI TCP/IP forwards mail to the mail hub (see Section 4.3.11.7).

6. Exit the configuration utility. When prompted, save the new parameters.

7. To make the changes take effect immediately, stop and restart the mail queues with **@IP$:START_SMTP.COM** to update the OpenVMScluster or with **@IP $:START_SMTP_LOCAL.COM** to update the local host only. Otherwise, your changes take effect the next time you reboot your system.

4.3.11.9. Excluding Hosts in Specific Domains From Mail Forwarding

If you configure VSI TCP/IP to forward mail addressed to remote users via a mail hub (see Section 4.3.11.10), you can exclude hosts in specific domains from the mail forwarding system by adding the domain to a list of "local domains." To modify the local domain list:

1. Make sure remote mail forwarding is enabled (see Section 4.3.11.8).

2. Start MAIL-CONFIG (**IP CONFIGURE /MAIL**).

3. To add a domain to the list with MAIL-CONFIG, use the **ADD LOCAL-DOMAIN *domain_name*** command. If *domain_name* begins with a dot, it specifies a domain name. Otherwise, *domain_name* specifies a host name.

4. To delete a domain from the list with MAIL-CONFIG, use the **DELETE LOCAL-DOMAIN *domain_name*** command.

5. Exit the configuration utility. When prompted, save the modified configuration.

6. To make the new configuration take effect immediately, stop and restart the mail queues with **@IP$:START_SMTP.COM** to update the OpenVMScluster or with **@IP $:START_SMTP_LOCAL.COM** to update the local host only. Otherwise, your changes take effect the next time you reboot your system.

4.3.11.10. Forwarding Local Mail

To configure VSI TCP/IP to forward mail addressed to local users via a mail hub:

1. Make sure the *FORWARDER* parameter specifies the host you want to use as a mail hub (see Section 4.3.11.7).

2. Start MAIL-CONFIG (**IP CONFIGURE /MAIL**).

3. Modify the *FORWARD-LOCAL-MAIL* parameter:

With MAIL-CONFIG, use the **SET FORWARD-LOCAL-MAIL TRUE** command.

4. If desired, exclude specific local users from mail hub forwarding (see Section 4.3.11.11).

5. If desired, exclude specific local users from mail hub forwarding (see Section 4.3.11.11).

6. Exit the configuration utility. When prompted, save the new parameters.

7. To make the changes take effect immediately, stop and restart the mail queues with **@IP$:START_SMTP.COM** to update the OpenVMScluster or with **@IP $:START_SMTP_LOCAL.COM** to update the local host only. Otherwise, your changes take effect the next time you restart your system.

The logical name `IP$SMTP_APPEND_FORWARDER_TO_MX` can be used to prevent SMTP from appending the forwarder to the MX list by default. To do this:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$SMTP_APPEND_FORWARDER_TO_MX FALSE
```

If the logical name is not defined (or is defined to anything not beginning with F, N, or 0), then the *`FORWARDER`* is appended to the MX list.

When the logical name `IP$SMTP_IGNORE_INTERFACE_NAMES` is defined (as /system, or any value), the VSI TCP/IP SMTP mail delivery procedure does not compare the destination address with the addresses of the interfaces on the system to determine if the message could be delivered locally. The default (no logical defined) is to check the addresses of the interfaces on the system. Defining this logical causes the MX records to be used exclusively in determining where a mail message should be delivered.

4.3.11.11. Excluding Specific Local Users from Mail Forwarding

If you configure VSI TCP/IP to forward local mail via a mail hub (see Section 4.3.11.10), you can exclude specific local users from the mail forwarding system by creating mail aliases for them in the `IP$:SMTP_ALIASES` file. Each users' alias must be in the following format: *`username`*`: *;`. For more information on configuring mail aliases, see Section 4.3.11.16.

4.3.11.12. Configuring Mail Gateways

You can configure VSI TCP/IP with gateways to particular hosts or domains to override the normal host lookup used by SMTP or to configure virtual domains not actually present on the network. You can use MAIL-CONFIG. To configure mail gateways with MAIL-CONFIG:

1. Start MAIL-CONFIG with the **IP CONFIGURE /MAIL** command.

2. Use the **ADD GATEWAY** and **DELETE GATEWAY** commands (for descriptions of these commands, refer to *VSI TCP/IP Administrator's Reference*).

Note

In VSI TCP/IP, only one gateway is allowed to be defined per domain. Preference numbers are not allowed.

---

3. Save the configuration with the **SAVE** command.

4. Quit MAIL-CONFIG with the **QUIT** command.

The modified configuration takes effect the next time your system reboots.

For example, to make it easier for users to address mail to BITNET users, configure a gateway for the `.BITNET` domain to point to one of the Internet-BITNET gateway systems, such as `CUNYVM.CUNY.EDU`:

```
MAIL-CONFIG>ADD GATEWAY .BITNET CUNYVM.CUNY.EDU
```

Once defined, any mail bound for an address ending in .BITNET is sent to the gateway you specified (in this case, `CUNYVM.CUNY.EDU`).

4.3.11.13. Specifying SMTP Host Aliases

If your system is a member of a OpenVMScluster, you can define *host aliases*, which are host names interpreted by the mailer as aliases for the actual local host name. You can specify these aliases in return addresses for individual users.

4.3.11.14. Setting Host Aliases

VSI TCP/IP relies on two parameters to obtain its list of host aliases:

Parameter	Description
`SMTP-HOST-NAMES`	Is a comma-separated list of up to 16 host aliases. If defined, the first alias in the list is the name used for outgoing mail. Any aliases are names for which your host accepts incoming mail.
`HOST-ALIAS-FILE`	Is the complete file specification of a file containing an unlimited list of host alias entries (one entry per line). The `HOST-ALIAS-FILE` value defaults to `IP$:SMTP_HOST_ALIASES`.

To change your host aliases with MAIL-CONFIG, use the **SET SMTP-HOST-NAMES** command or the **SET HOST-ALIAS-FILE** command and save the modified configuration with the SAVE command. The new configuration takes effect the next time you reboot the system or the queues are restarted. Alternatively, specify the alias file name dynamically with the following command:

```
$ DEFINE/SYSTEM/EXEC IP$SMTP_HOST_ALIASES_FILE file-spec
```

If this logical name is not defined, by default the SMTP software looks for the file `IP $:SMTP_HOST_ALIASES`. Names in the host aliases file should be listed one per line.

4.3.11.15. Specifying Host Aliases for Individual Users

The logical name `IP$SMTP_FROM_HOST` lets you change the host name that appears in your return address on outgoing mail.

Normally, the host name you choose must be a "local" host name; that is, it must be one of the registered SMTP host name aliases on the system (either from the SMTP-HOST-NAMES setting or the HOST-ALIAS-FILE). If it is not a known alias, the setting is ignored.

If you define the host name in executive mode, however, `IP$SMTP_FROM_HOST` can be any arbitrary host name. The name is not checked against the SMTP host name.

When the logical `IP$SMTP_ENVELOPE_FROM_HOST` is defined the value is used for the host name instead of the actual host name when sending the MAIL FROM: line to the remote server. This is

useful if there are multiple independent systems that send mail that you would like to appear to be a single system.

This feature lets users from different administrative entities within an organization have return addresses that reflect the names of those entities. To enable this feature:

1. Set up MX records in DNS so mail is routed to the local host for each separate host name.
2. Set up SMTP-HOST-NAMES or the HOST-ALIAS-FILE with a list of host names.
3. Define the logical name `IP$SMTP_FROM_HOST` for each user. Base the value for this logical name on some aspect of the department or organization to which the user belongs.

4.3.11.16. Configuring Mail Aliases

The VSI TCP/IP SMTP system supports system-wide mail aliases, system-wide mailing lists, and per-user mail aliases. The default system-wide alias file is `IP$:SMTP_ALIASES`. You can configure this name or specify a list of alias file names.

Per-user mail aliases are kept in the file SMTP_ALIASES in each user's login directory. The format for alias entries is: *`alias: real_address[,...];`*

- *`alias`* is an alphanumeric string.
- *`real_address`* is either a local or remote electronic mail address.

You can specify multiple addresses by separating them with commas; the alias definition may span multiple lines, if needed, and must always be terminated with a semicolon (;).

For example, a local user has the user name "JB134A", but wants to receive SMTP mail sent to the address "john". The system manager adds the following line to the alias file:

```
john: jb134a;
```

You can both forward a mail message and deliver it to a local mailbox by adding the mailbox name, preceded by an underscore, to the `IP$:SMTP_ALIASES` file. The following example shows such an alias entry:

```
FNORD: FNORD@SOMEWHERE.FLOWERS.COM, _FNORD;
```

The leading underscore on the second address (_FNORD), tells the SMTP symbiont to skip any further alias processing.

4.3.11.17. Mailing Lists

Mailing lists are a special form of mail alias and are supported only in the system-wide alias files. The format for specifying a mailing list is: `list-name::` *`owner-address, file-spec;`*.

- A double-colon (::) signifies that this alias is a mailing list.
- *`owner-address`* is the address of the mailing list owner. Messages sent to this mailing list go to each subscriber on the list with the return-path set to this address. The owner address can be an actual user's address or an alias, if desired.
- *`file-spec`* is the file specification for the file containing the subscribers to the mailing list. Specify a complete path name for this file, including the device and directory.

For example, you might want to set up a mailing list called OPERATIONS-STAFF for your operations staff, and have your operations manager, user OPER1, manage that list. You might set up the mailing list this way:

```
Operations-Staff:: Operations-Manager, USERS:[OPER1]STAFF.LIST;
Operations-Manager: OPER1;
```

Mail sent to OPERATIONS-STAFF is forwarded to the addresses listed in `USERS:[OPER1]STAFF.LIST`. Because this file is in OPER1's area, the operations manager has control over who is included in the list. The list is set up in this example so the return-path on list messages is set to "Operations-Manager" instead of user OPER1; setting up the list owner as an alias makes it easier to change list owners at a later date.

4.3.11.18. Specifying the System-Wide Mail Alias File

By default, the VSI TCP/IP SMTP system obtains system-wide mail aliases from the `IP$:SMTP_ALIASES` file. You can configure VSI TCP/IP to use any other file, or to use multiple files.

To change the SMTP aliases file with MAIL-CONFIG, use the **SET ALIASES-FILE** command, then save the modified configuration with the **SAVE** command. The new configuration takes effect the next time you reboot the system.

4.3.11.19. Using Mail Aliases and Mailing Lists From OpenVMS MAIL

If you want aliases configured within the VSI TCP/IP SMTP alias file to be accessible to local OpenVMS MAIL users (or those connected via DECnet), specify the address using the VSI TCP/IP SMTP MAIL foreign mail protocol interface.

For example, a local user wanting to send mail to the "gcc-users" mailing list would specify the address SMTP%"gcc-users".

You can, however, define a OpenVMS MAIL alias containing the SMTP% specification. To define the OpenVMS MAIL alias "Operations-Staff," use the OpenVMS MAIL **SET FORWARD** command:

```
MAIL> SET FORWARD SMTP%""""Operations-Staff-USERS"""" /USER=Operations-Staff
```

VSI TCP/IP SMTP uses the RFC-822 To: and CC: headers to provide the contents of the OpenVMS mail To: and CC: fields. To enable this processing, define the logical name `IP$VMSMAIL_USE_RFC822_TO_HEADER`.

VMS mail limits the length of its To: and CC: fields to 255 characters.

4.3.12. IMAP Server

The Internet Message Access Protocol (IMAP) server lets an IMAP-compliant client mail program access remote message storage as if the storage were local. VSI TCP/IP's implementation is based on IMAP Version 4, Revision 1.

IMAP and the Post Office Protocol (POP3), described in the next section, operate differently. IMAP retains the message on the server, whereas POP3 retrieves the message and stores it "off-line" on the client, thereby deleting it from the mail server. IMAP does not delete the mail message and lets you access your mail from more than one client workstation at a time.

IMAP was designed to:

- Be fully compatible with Internet messaging standards, such as MIME.
- Allow message access and management from more than one computer.
- Allow access without relying on less efficient file access protocols.
- Provide support for "online," "offline," and "disconnected" access modes
- Support concurrent access to shared mailboxes.
- Eliminate the need for the client software to know about the server's file storage format.

The IMAP protocol includes operations for:

- Creating, deleting, and renaming mailboxes
- Checking for new messages
- Permanently removing messages
- Setting and clearing flags
- Server-based RFC-822 and MIME parsing and searching
- Selective fetching of message attributes, texts, and portions thereof, for efficiency

4.3.12.1. Inhibiting Output in Command Procedures for the IMAP Service

Problems arise when remote users log into systems using a login command procedure (`SYS$LOGIN:SYLOGIN.COM` or `SYS$MANAGER:SYLOGIN.COM`) that requires screen output. To inhibit this behavior, make sure the following lines are included at the top of all login command procedures:

```
$ VERIFY = 'F$VERIFY(0)
! Turn off verify without echoing
$ IF F$MODE() .EQS. "OTHER" THEN EXIT
! If a DETACHED process (IMAP)
$ IF VERIFY THEN SET VERIFY
! If a batch job, may want to turn
! verify back on.
```

4.3.12.2. IMAP Mail Folders

In contrast to POP3, IMAP allows you to access server mail folders (message stores) other than INBOX. In MAIL, for example, if you create a `NOTES` folder, you can access mail in that folder. This `NOTES` folder can be in a mail file other than the default `MAIL.MAI` file. In fact, you can set a configuration parameter that determines the way mail folders are presented to the client so that you can use folders in these other mail files.

Your default mail directory includes a `.IMAPRC` file in which you can set certain configuration directives (described more fully in Section 4.3.12.3). Among these directives is **allow-subfolders**.

This directive specifies that folder names are comprised of a directory (optional), mail file, and folder. For example, the NOTES folder in MAIL.MAI is represented as mail/notes (as opposed to just notes if the directive were not set). This would distinguish it from another NOTES folder in the OLD.MAI mail file, for example, which would be named old/notes.

Each level beyond the second in this hierarchy represents a subdirectory of the default mail directory. For example, the NOTES folder in [.ARCHIVED]MAIL.MAI has the IMAP equivalent of archived/mail/notes.

Because of this folder syntax ambiguity, directory names, file names, and folders can overlap, such as the examples in Table 4.4.

Table 4.4. IMAP Mail Folders Overlapping Syntax Examples

This mail file...	Containing this folder...	Has this IMAP equivalent...
MAIL.MAI	NOTES	mail/notes
[.MAIL]NOTES.MAI	STUFF	mail/notes/stuff
[.MAIL.NOTES]STUFF.MAI	BOBS	mail/notes/stuff/bobs

Entries in the syntax can at different times be mail files, directories, subdirectories, or folders. Because of this overlap, the server must keep an internal representation of the hierarchy and mark what each level of the folder name means. This information is critical when renaming or deleting folders.

One restriction is that a first level folder (MAIL, for example) cannot be a message store, since it represents only a file and not a mail folder. INBOX, however, is a special case. INBOX is always INBOX, cannot be deleted or renamed (a rename moves messages to the renamed folder but does not delete INBOX), and never goes away. In IMAP, INBOX is **MAIL /NEWMAIL** by default, and is hidden to the user. (Note that you can change the mail "in-box" from INBOX to another folder by defining the **file-inbox-messages-to-folder** directive in the .IMAPRC file. See Section 4.3.12.3.)

You can also access mail files in your login directory (SYS$LOGIN) by prefixing the folder name with a tilde (~). The ~ folder is reserved and cannot be used by other folders.

4.3.12.3. IMAP Directives File

Users can set certain preferences by creating a file in their default mail directory called .IMAPRC and including directives. Table 4.5 lists these directives along with their meanings. Each directive must be on its own line and in lowercase.

Table 4.5. IMAP Configuration Directives in .IMAPRC

This directive...	Does the following...
set allow-child-folders	Enables or disables the use of subfolders and the way that folders are presented to the client. By default, this value is **false**. If you want to set the value to be true, put this line in the .IMAPRC file: set allow-child-folders true. (See Section 4.3.12.2 for details.)
set autofile-messages-to-folder *foldername*	Moves read messages from INBOX to the specified folder in the user's default mail file. By default, this option is disabled.

This directive...	Does the following...
set case-insensitive-folders	Specifies that folder names are case-insensitive. Otherwise, two folders with the same name but with different cases could become inaccessible to the IMAP client. Newly created folders are created in uppercase on the server. By default, this value is **false**. If you want to set the value to be true, put this line in the `.IMAPRC` file: `set case-insensitive-folders true.`
set do-purge-reclaim	Enables or disables purge-reclaim operations upon closing a folder. By default, this value is **true**.
set folder-timer *`delta_time`*	Specifies the frequency the IMAP server will check for externally created folders. By default, this value is **00:02:00** (2 minutes).
set inbox-folder *`foldername`*	Maps INBOX to the specified folder in the user's default mail file. By default, this value is NEWMAIL.
set newmail-timer *`delta_time`*	Specifies the frequency the IMAP server will check folders for new messages. (Note that checking for new mail is time consuming for large folders.) By default, this value is **00:00:30** (30 seconds).

4.3.12.4. IMAP Options in the Global IMAPD.CONF file

These options are valid in the global `IMAPD.CONF` file (`IP$:IMAPD.CONF`) as shown in Table 4.6:

Table 4.6. Valid options in the Global IMAPD.CONF file (IP$:IMAPD.CONF)

This directive...	Does the following...
set decnet-address *`nodename namespace domainname`*	Maps the specified decnet nodename and/or namespace to a given domain name. Use " " to represent either a blank nodename or namespace. Multiple entries are allowed. By default, this option is disabled. Example: set decnet-address knob " " door.com.
set enable-full-cache	Enables or disables full message caching. By default, this value is **false**.
set max-ping-count *`integer`*	Specifies, in number of messages, the threshold at which the server will no longer attempt to check for new messages. By default, this value is **20000**.
set smtp-transport-prefix string	Specifies the SMTP transport prefix. By default, this value is SMTP. If you want to change it to MX, put this line in the `IMAPD.CONF`: `set smtp-transport-prefix MX.`
set trailing-header-marker string	Specifies a text string used to indicate the start of RFC822 message headers if the system does not

This directive...	Does the following...
	place them at the start of the message. By default, this option is disabled.

4.3.12.5. IMAP State Information Files

The IMAP server includes files created in the user's mail directory where it maintains state information, as shown in the following table.

Table 4.7. IMAP State Information Files

This file...	Stores...
`.MAILBOXLIST`	Folders to which the user subscribes.
`.NEWMAILBOXES`	List of folders known to be empty. OpenVMS MAIL deletes folders once it deletes the last message, so that the server must remember these folders.
`mailfolder.foldernameuidvalidity`	For each folder, the UIDs for all the messages. The file name is composed of the folder name and its UIDVALIDITY code. For example: MAIL.NEWMAIL3B3200E6. In the example, the folder name is NEWMAIL and the UID validity code is 3B3200E6.

4.3.12.6. IMAP Logicals

The following IMAP logicals are supported:

4.3.12.7. IP$IMAPD_MESSAGE_ONE

By default, an informing message of This message cannot be retrieved is sent to the user when the processing message is too big. You can use this logical to define a different informing message. For example, if you define the logical like this:

```
$ define/sys/exe IP$IMAPD_MESSAGE_ONE "Your mail is too big to be
 processed"
```

The message seen by the user is:

```
Your mail is too big to be processed.
```

If the defined logical string starts with @, the rest of the string defines a text file that holds the text. The maximum file size of the informing message is 255 bytes. For example, if you define the logical like this:

```
$ define/sys/exe IP$IMAPD_MESSAGE_ONE "@IP$:IMAPD_MSG_ONE.TXT"
```

and edit the file `IP$:IMAPD_MSG_ONE.TXT` to be:

```
The size of this message is too big for the IMAP server to process.
Most likely it has a big attachment file. Contact the sender to arrange re-
transmission
by other means such as FTP.
System Manager.
```

The text message from the System Manager is seen by the user when the processing message cannot be retrieved.

4.3.12.8. IP$IMAPD_MESSAGE_SIZE_LIMIT

There could be times when an oversized mail file prevents the mail system from working. This oversized mail file could also slow down other processes on the system. If such a case happens, use this logical to limit the size of the mail that IMAP processes.

When a mail file size reaches the limit defined by this logical, IMAP does not process it and sends a message to the user. Use the following metrics to define this logical: "S" (40K records), "M" (120K records), or "L" (240K records).

Because OpenVMS uses records to define the mail size and a record could have as few as 1 byte to as many as 255 bytes, the size limit defined by the logical does not reflect the actual mail size in bytes. For example:

```
$ define/sys/exe IP$IMAPD_MESSAGE_SIZE_LIMIT "M"
```

A mail with a 5MB attachment file could reach the limit. But another mail with a 6MB attachment file could pass the limit.

Note

If the logical is not defined, the mail size limit is actually the current OpenVMS Page file limit quota (Pgflquo) of the IMAP process.

4.3.12.9. IP$IMAPD_LOGLEVEL n

This logical enables additional logging for debugging purposes. Output is written to the file `IP$:IMAP_SERVER.LOG`. By default, this logical is unassigned. IMAP events normally are logged to SYSLOG.

4.3.12.10. IP$IMAP_UPDATE_LOGIN_TIME

This logical updates the last non-interactive login field in this SYSUAF for each IMAP login.

4.3.13. Post Office Protocol (POP) Version 3

The VSI TCP/IP Post Office Protocol version 3 (POP3) servers allow a remote user to access mail stored in OpenVMS MAIL files from POP3 mail clients. The POP3 is documented in RFC-1460, "Post Office Protocol: Version 3."

The remote user must have a valid account on the OpenVMS host and mail must be delivered into the OpenVMS MAIL files associated with the account. A user specifies a user name and password when OpenVMS MAIL is accessed from a POP3 client.

The POP3 server accepts Kerberos V4 authentication in place of user name and password authentication. The POP3 server allows cross-realm Kerberos V4 authentication when the logical name `IP$POP3_ENABLE_CROSS_REALM_AUTHENTICATION` is defined.

The POP3 server waits indefinitely for input from POP3 clients. A logical is definable so you can specify the amount of time the POP3 server should wait for input before closing the connection. The logical is `IP$POP3_INPUT_WAIT` *x*, where *x* is a normal OpenVMS time string.

The current release of the VSI TCP/IP POP3 server does not support the APOP authentication mechanism. The user name and password are validated by LOGINOUT, and a server process is created on the OpenVMS host. The server process invokes both the system-wide login command procedure (SYS$MANAGER:SYLOGIN.COM) and the user's LOGIN.COM before the POP server image is run. A log file is created in the user's login directory detailing the transactions between the client and the OpenVMS system, which is named POP3_SERVER.LOG.

4.3.13.1. POP Logical Names

You can define the logical names described in this section to provide additional functionality to POP3. These logicals, which affect several aspects of mail operation, can be defined either system-wide or for individual users.

Note

The default value for all of these logicals is 0 (all bits disabled).

IP$POP3_UPDATE_LOGIN_TIME

When defined **/SYSTEM**, this causes the **LAST NONINTERACTIVE DATE/TIME** field in SYSAUF to be updated with the time the POP3 session was started. Note that this logical applies to POP3 sessions only.

4.3.13.2. Specifying POP Functions Using the IP$POP3_FLAGS Logical

This logical specifies functions as decimal values that are interpreted as described in the following table. To define more than one function, add the decimal values together; for an example, see Section 4.3.13.4.

Value	Effect
1	Read only messages marked as new from the NEWMAIL folder. The default is to read all messages in this folder. If IP$POP3_SOURCE_FOLDER is defined, this flag is ignored.
2	Move messages from the NEWMAIL folder to the MAIL folder after they are read. If IP$POP3_DEST_FOLDER is defined, this flag is ignored.
4	Release the POP client before the OpenVMS mailbox is actually closed; that is, cause a quick close. OpenVMS MAIL can take several minutes to reclaim large amounts of deleted message space. The server replies to the client with a success, closes the connection, releases the client, and closes the mailbox. Any errors that result from closing the mailbox when this flag is enabled are lost.
16	Remove the "NODE::" portion of the "From:" address. The VSI TCP/IP POP server creates RFC-822 headers from OpenVMS headers before sending the message to the POP client. If the mail message was received via SMTP, SMTP headers are used. When NODE is the same as the OpenVMS logical name SYS$NODE, the "NODE::" portion of the address is removed; otherwise, this portion of the address is retained. For example:

Value	Effect
	• For a local node WHITEFANG in which a message originates via DECnet from a remote OpenVMS host, the "From:" line is WHARFIN::HOLMES. • After conversion to RFC-822 conventions, the line is "WHARFIN::HOLMES"@WHITEFANG.YOYODYNE.COM. • Because some POP clients cannot reply to this address, the bit value represented by **16** should be enabled. When enabled, the "From:" line becomes HOLMES@YOYODYNE.COM. **Note** If a message is routed via DECnet before being received by SMTP, enabling this bit invalidates the return path to the remote DECnet node.
32	Deleted messages are saved into the folder specified by IP$POP3_DEST_FOLDER. Otherwise, when a message is deleted, it is removed completely.
64	POP builds headers from the OpenVMS Mail "From" statement. Specify this flag when your mailer has been configured to not insert the SMTP headers as the first lines of text in the message. The only time you should use this value in the IP$POP3_FLAGS logical name is when you want to construct ad hoc SMTP headers (that is, when your SMTP agent is configured to place the real headers at the bottom of the message).
128	The mail box is compressed, but all versions of MAIL.OLD in the MAIL directory are deleted.

4.3.13.3. Setting the IP$POP3_DEST_FOLDER and IP$POP3_SOURCE_FOLDER Logicals

The IP$POP3_DEST_FOLDER logical specifies the folder containing messages that have been read.

The IP$POP3_SOURCE_FOLDER logical specifies the folder from which to read messages. The default folder name is NEWMAIL.

By default, POP3 servers look for mail in the NEWMAIL folder of the user's OpenVMS MAIL files. This default may be overridden by defining the logical names IP$POP3_SOURCE_FOLDER and IP$POP3_DEST_FOLDER in the file SYS$LOGIN:LOGIN.COM. For example, if a POP3 user wants to access mail stored in the MAIL folder by default, place the following command in LOGIN.COM:

```
$ DEFINE IP$POP3_SOURCE_FOLDER "MAIL"
```

Note

OpenVMS MAIL folder names are case-insensitive.

When a mail message is accessed by POP3, it remains in its original folder until the POP client deletes it. This happens even if the mail is being accessed from the user's NEWMAIL folder. Users may, however, define the logical name IP$POP3_FLAGS, using the value 2, in their LOGIN.COM files to alter this behavior. If IP$POP3_FLAGS is set to 2, mail messages are placed in a user's

MAIL folder. This occurs after they are read via POP from the NEWMAIL folder, if the POP client does not delete the messages after it reads them.

This behavior is the same as with OpenVMS MAIL. For example, if a POP3 user wants mail read from the NEWMAIL folder placed in the MAIL folder after being read, add this command to LOGIN.COM:

```
$ DEFINE IP$POP3_FLAGS "2"
```

Again, the user must also configure the POP3 client so messages are not deleted after they are read. If the client deletes a message, it is not saved in either the MAIL or NEWMAIL folder on the OpenVMS server.

4.3.13.4. Defining the Logicals System-Wide

All POP logical names can be defined system-wide for all users, as shown in the following example:

```
$ DEFINE /SYSTEM /EXECUTIVE IP$POP3_FLAGS "7"
```

This example sets the flags parameter so that:

- All users only read messages from the NEWMAIL folder that are marked as new.
- Messages are moved from the NEWMAIL folder to the MAIL folder after they are read.
- The POP client is released quickly at the end of the mail session.

4.3.14. Configuring the SMTP-DECnet Mail Gateway

VSI TCP/IP can be set up as a gateway to route mail between SMTP and DECnet-only hosts, with appropriate address translations to make the DECnet-style addresses easier for Internet hosts to interpret. To do this, you set the DECNET-DOMAIN mail parameter and add an appropriate MX record to the Domain Name Service. The addresses of DECnet mail sent out via SMTP will be rewritten such that the DECnet node name(s) appear under the DECNET-DOMAIN name in the host-part of the address. The addresses of incoming SMTP mail for hosts under the DECNET-DOMAIN are automatically converted into DECnet addresses and delivered to the DECnet-only hosts.

4.3.14.1. DECnet-to-SMTP Mail

In the DECnet-to-SMTP direction, a OpenVMS MAIL user on a DECnet-only host sends SMTP mail by specifying an address of the form:

node::SMTP%*user@host*

node is the DECnet node name of the system running VSI TCP/IP.

VSI TCP/IP recognizes that the mail originated in DECnet and, if the DECNET-DOMAIN parameter is set, rewrites the originating address in the form

user@node.decnet-domain.

For example, FLOWERS.COM has set up node HQ as a DECnet-SMTP gateway. A user named JOHN on DECnet-only node WHARFIN at FLOWERS.COM addresses mail to the Info-VSI TCP/IP mailing list as: HQ::SMTP%"Info-VSI TCP/IP for OpenVMS@ABC.COM"

JOHN's DECnet return address, WHARFIN::JOHN, is rewritten by the gateway as:

```
JOHN@WHARFIN.DNET.FLOWERS.COM
```

instead of:

```
"WHARFIN::JOHN"@HQ.FLOWERS.COM
```

which some Internet mailers would have trouble parsing.

4.3.14.2. SMTP-to-DECnet Mail

For the SMTP-DECnet gateway to work in the SMTP-to-DECnet direction, other hosts on your network must be told that mail for host names under the DECNET-DOMAIN should be sent to the gateway host. If you use Domain Name Service, the easiest way to do this is to set up a wildcard MX record for the DECNET-DOMAIN. In our example, the MX record looks like this:

```
*.DNET.FLOWERS.COM. IN MX 0 HQ.FLOWERS.COM.
```

This MX record causes other hosts on the Internet to send mail destined for any host under `DNET.FLOWERS.COM` to node HQ. The gateway automatically recognizes the DECNET-DOMAIN in the host-name part of the address, rewrites the address to its DECnet form, and sends it through OpenVMS MAIL.

If you do not use DNS, you must add a fully qualified host name for each DECnet node to your host tables. In our example, a return message to user JOHN on node WHARFIN would be addressed to:

```
JOHN@WHARFIN.DNET.FLOWERS.COM.
```

When HQ receives that message, it translates the address to its DECnet form:

```
WHARFIN::JOHN
```

and sends the message to that address using OpenVMS MAIL.

Chapter 5. Printer Configuration

This chapter describes the printing services provided by VSI TCP/IP. It supports printing through the LPD, IPP and telnet (STREAM and NTY) protocols.

5.1. LPD/L Configuring the PR Server

An LPD server allows other hosts and networks to access queues on the server system. The queues can be any valid OpenVMS queue, including LAT queues and terminal queues, and do not have to be VSI TCP/IP print client queues. A VSI TCP/IP system is disabled as an LPD server by default in the distribution kit.

To grant access to other hosts or networks:

- Run the VSI TCP/IP Server Configuration Utility (SERVER-CONFIG) with the command: **$ IP CONFIGURE /SERVERS**
- Enable the LPD server: SERVER-CONFIG>**ENABLE LPD**
- Select the LPD server: SERVER-CONFIG>**SELECT LPD**

To add hosts, type the command: SERVER-CONFIG>**SET ACCEPT-HOSTS**

You are first prompted to delete the hosts currently in the configuration, then prompted to add new hosts to the configuration by their IP addresses. To add access to all hosts on a particular network, issue the command:

```
SERVER-CONFIG>SET ACCEPT-NETS
```

You are prompted in a similar fashion for networks.

Access control changes take effect immediately when you restart the VSI TCP/IP master server process with the **RESTART** command before exiting. To log errors to OPCOM, type this command before restarting the master server process:

```
$ REPLY/ENABLE=NET/TEMP
```

The following example shows how to grant LPD access to two specific hosts and to all hosts on a given network. (Note that HOSTS.EQUIV is not consulted when determining trusted hosts.)

```
$ IP CONFIGURE /SERVERS
VSI TCP/IP for OpenVMS Server Configuration Utility 10.5(nnn)
[Reading in configuration from IP$:SERVICES.MASTER_SERVER]
SERVER-CONFIG>SHOW LPD /FULL
Service "LPD":
 TCP socket (AF_INET,SOCK_STREAM), Port 515
 Socket Options = SO_KEEPALIVE
 INIT() = TCP_Init
 LISTEN() = TCP_Listen
 CONNECTED() = TCP_Connected
 SERVICE() = Run_Program
 Program = "IP$:SERVER_LPD.EXE"
 Accept Hosts = IP*127.0.0.1
 Reject by default all other hosts and nets
 Reject Message = "Your host does not have line printer access"
```

```
SERVER-CONFIG>SELECT LPD
[The Selected SERVER entry is now LPD]
SERVER-CONFIG>SET ACCEPT-HOSTS
Delete address "IP*127.0.0.1" ? [NO]
You can now add new addresses for LPD. An empty line terminates.
Add Address: 192.41.228.1
Add Address: 192.41.228.65
Add Address:
SERVER-CONFIG>SET ACCEPT-NETS
You can now add new addresses for LPD. An empty line terminates.
Add Address: 192.16.72.0
Add Address:
SERVER-CONFIG>RESTART
Configuration modified, do you want to save it first ? [YES]
[Writing configuration to IP$COMMON_ROOT:[IP]SERVICES.MASTER_SERVER]
%RUN-S-PROC_ID, identification of created process is 202002BD
SERVER-CONFIG>EXIT
[Configuration not modified, so no update needed]
```

5.1.1. Setting a Default LPD User Name

A print request expects to be printed under a valid OpenVMS user name. If the print request originates from a remote system, the user name may not map to a valid OpenVMS user name. When defining a default LPD user name, any print request that does not map to a valid OpenVMS user name maps to the default LPD user name, and is accepted by the LPD server.

It is recommended that you define the default LPD user name even if the user names on the remote system match those on the OpenVMS system. Some printing systems use the name "root" or "lpd" instead of the actual user name when dispatching a print job to a remote system; those names may not match any valid OpenVMS user name. To set a default LPD user name:

```
$ IP CONFIGURE
VSI TCP/IP for OpenVMS Network Configuration Utility 10.5(nnn)
[Reading in MAXIMUM configuration from IP$:IP.EXE]
[Reading in configuration from IP$:NETWORK_DEVICES.CONFIGURATION]
NET-CONFIG>SET LPD-DEFAULT-USERNAME PYEWACKET
NET-CONFIG>EXIT
[Writing configuration to IP$:NETWORK_DEVICES.CONFIGURATION]
[Writing Startup file IP$:IP$SYSTARTUP.COM]
[Changes take effect after the next OpenVMS reboot]
```

To make the default user name take effect immediately without rebooting the system, define the following logical:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$LPD_DEFAULT_USERNAME "PYEWACKET"
```

Note

The LPD default user name must exist in the UAF file. It can be non-privileged and it can be "disusered," but it must exist.

You can set up logical names to map remote LPD users to local users through a mechanism known as *proxy access*. Using logical names is useful when you want to receive LPD print jobs from a UNIX system on which the user names and UIDs on the client and server are completely uncoordinated.

Note

Proxy mappings take precedence over the default mapping. To map a remote user's LPD requests for printing as if they were queued by a local user:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$LPD_PROXY_'user' 'local_user'
```

For example, to map remote user BROWN's LPD requests for printing as if they were queued by the local user JONES:

```
$ DEFINE/SYSTEM/EXECUTIVE IP$LPD_PROXY_BROWN JONES
```

5.1.2. Changing the LPD Spool Directory

By default, LPD print jobs (and SMTP mail messages) on the OpenVMS system are stored in the directory `IP$COMMON_ROOT:[IP.SPOOL]`. You can change the directory with the **NET-CONFIG SET SPOOL-DIRECTORY** command by entering:

```
$ IP CONFIGURE
NET-CONFIG>SET SPOOL-DIRECTORY DISK$TEMP:[IP]
```

You must redefine the logical that points to the spooling area unless you reboot the system after modifying the VSI TCP/IP configuration by entering:

```
$ DEFINE/SYSTEM/EXEC IP$SPOOL DISK$TEMP:[IP]
```

Make sure the directory protections are set to SYSTEM:RWED, OWNER:RWED, GROUP:RE, and WORLD:RE.

5.1.3. Cancelling LPD Print Jobs

If the print job is still on your local system, you should use the VMS command **DELETE /ENTRY** to delete the job.

5.1.4. Controlling host name lookup

If the logical `IP$LPD_DO_ASYNC_LOOKUP` is not defined to 1, True, or Yes, then the LPD print symbiont will now resolve the remote host name with getaddrinfo, and is capable of resolving and connecting to an IPv6 address.

```
$ DEFINE/SYSTEM/EXEC IP$LPD_DO_ASYNC_LOOKUP TRUE
```

5.1.5. Configuring Printers on Remote Systems

Once you have set up the LPD server to accept incoming connections, you must set up the remote system for remote printing. When setting up print queues on other systems, specify the OpenVMS queue name as the remote printer name. (You can use the # character in remote printer names.) For example, an Apple LaserWriter is attached to the VSI TCP/IP host ABC.COM and is configured as the queue SYS$PRINT. To configure a remote printer queue named "laser" on a UNIX system to direct output to SYS$PRINT on ABC.COM, add the following entry to /etc/printcap:

```
laser|lw|LaserWriter II on host ABC.COM:\
```

```
:rm=abc.com:rp=sys$print:sd=/var/spool/laser:lp=:
```

The following fields are required:

Name	Purpose
laser	Remote system queue name; in this example, laser
rm	Node name of the OpenVMS LPD server
rp	OpenVMS queue name on the OpenVMS LPD server
sd	Local UNIX spool directory
lp	Set to null to avoid LPD bugs

When a user on the UNIX system issues this command:

```
% lpr -Plaser foo
```

lpr searches /etc/printcap on the local system for an entry of "laser", and transfers the file "foo" to the LPD server on ABC.COM, which queues "foo" to SYS$PRINT.

5.1.6. Checking Remote Printer Queues

To display the contents of a remote queue that serves an OpenVMS print queue, use the command:

```
$ IP SHOW /QUEUE=vms_queue_name
```

vms_queue_name is the name of the local OpenVMS queue.

The contents of the remote queue are accessed for display using the TCP LPD service.

Note

Queues configured with the STREAM protocol cannot be displayed with this command.

5.1.7. LPD Jobs (Inbound)

VSI TCP/IP allows control of the job-queueing parameters used for incoming LPD print jobs using logical names. The formats for these logical names, which specify their scope, are:

`IP$LPD_`*`queuename_typechar_parametername`*

Specifies a parameter value for a specific queue and file type.

`IP$LPD_`*`queuename_*_parametername`*

Specifies the default for a specific queue and all file types on that queue.

`IP$LPD_*_`*`typechar_parametername`*

Specifies the default for all queues with a specified file type.

`IP$LPD_*_*_parametername`

Specifies a default for all queues and all file types.

Enter an asterisk (*) in a logical name to match any *queuename* or *typechar*. An asterisk cannot be used for a *parametername*. The values in the logicals are:

- *parametername*, which is one of the following:

Value	Description
P1, P2, P3, P4, P5, P6, P7, P8	OpenVMS print job parameters [PRINT / PARAM=()]
BURST	**/BURST** qualifier (ALL, ONE, or NO)
FEED	**/FEED** qualifier (YES or NO)
FILETYPE	Type of spool file (FIXED512 or LFSTREAM)
FLAG	**/FLAG** qualifier (ALL, ONE, or NO)
FORM	Form name
PASSALL	**/PASSALL** qualifier (YES or NO)
SETUP	**/SETUP** qualifier, list of setup modules
TRAILER	**/TRAILER** qualifier (ALL, ONE, or NO)

- *queuename*, the name of a printer visible to LPD clients.
- *typechar*, a single letter specifying the file's data type; accepted values are **c**, **d**, **f**, **g**, **l**, **n**, **p**, **r**, **t**, and **v**. The **f** value is the **lpr** default if a user does not specify a type flag. **r** is required if a user specifies **-f**.

For example, if you want print jobs sent to your system via LPD and submitted to the OpenVMS print queue called PRINT01 to use the bottom input tray, you would issue this command:

```
$ DEFINE/EXEC/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_PRINT01_*_P1 "INPUT_TRAY=BOTTOM"
```

These logical names must all be defined in the `IP$PRINTER_TABLE` logical name table. If you have not already run PRINTER-CONFIG, the `IP$PRINTER_TABLE` logical name table may not exist. Verify the `IP$PRINTER_TABLE` logical name table exists:

1. Start PRINTER-CONFIG with the following command:

   ```
   $ IP CONFIGURE /PRINTERS
   ```

2. Save the unmodified configuration:

   ```
   PRINTER-CONFIG>SAVE
   ```

   This command writes the file `IP$:REMOTE-PRINTER-QUEUES.COM` which contains the command to create the `IP$PRINTER_TABLE` logical name table.

3. Exit PRINTER-CONFIG.

   ```
   PRINTER-CONFIG>EXIT
   ```

4. Execute the **IP$:REMOTE-PRINTER-QUEUES.COM** command procedure.

Note

IP$:REMOTE-PRINTER-QUEUES.COM is executed automatically when VSI TCP/IP starts. If you define these logical names manually, they will be lost after rebooting. To preserve your logical name definitions, add the appropriate **DEFINE** commands to your startup command procedure.

The following example shows how to configure the VSI TCP/IP LPD server to handle inbound jobs submitted with the **-v** option.

```
$ DEFINE/EXECUTIVE_MODE/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_*_V_PASSALL "YES"
$ DEFINE/EXECUTIVE_MODE/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_*_V_FILETYPE "FIXED512"
```

The following example specifies that all LPD jobs queued to SYS$PRINT be submitted with the form "WIDE":

```
$ DEFINE/EXECUTIVE_MODE/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_SYS$PRINT_*_FORM WIDE
```

The `IP$PRINTER_TABLE` logical name table and all logical names within the table must be defined in executive mode. To confirm this, enter the following command:

```
$ SHOW LOGICAL/FULL/TABLE=IP$PRINTER_TABLE
```

- `IP$PRINTER_`*`queuename`*`_PASSALL_FILTER` if a queue or a job is set to /**PASSALL**, this logical makes the default filter character "**v**" unless this logical exists and specifies a different value.
- `IP$PRINTER_`*`queuename`*`_SUPPRESS_REMOTE_BANNER` if defined, does not include the **L** command in the control file to request a banner page.
- `IP$PRINTER_`*`queuename`*`_NO_FFLF_DEFAULT` or `IP $PRINTER_*_NO_FFLF_DEFAULT` if the value is "Y", "T" or "1", then NOFFLF is enabled; otherwise it is disabled.

5.1.8. Troubleshooting the LPD Server

The error message "record too large for user's buffer," indicates a file format problem. Define the following logicals, and specify that all print commands and any print qualifiers are treated as fixed 512-byte files.

```
$ DEFINE/EXECUTIVE_MODE/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_*_*_PASSALL TRUE
$ DEFINE/EXECUTIVE_MODE/TABLE=IP$PRINTER_TABLE -
_$ IP$LPD_*_*_FILETYPE FIXED512
```

For information about interpreting the asterisk (*) in logical names, see Section 5.1.7.

The file is removed from the queue on the client side, but the job never appears in the OpenVMS queue. Enable OPCOM with the command:

```
$ REPLY/ENABLE=NET/TEMP
```

and try the PRINT command again. The OPCOM error message, "your host does not have line printer access," indicates that the server has not been set up to allow incoming connections from the remote host. See Section 5.1.

If no OPCOM error messages display, use the following command to verify that the default LPD user name has been defined in EXECUTIVE mode:

```
$ SHOW LOGICAL/FULL IP$LPD_DEFAULT_USERNAME
```

If the logical does not exist, or is not defined in executive mode, refer to Section 5.1.1.

To generate a log file with LPD, do the following:

```
$ DEFINE/SYSTEM IP$LPD_SYMBIONT_DEBUG
```

A value is not required for the logical. If the logical exists, debug logging is ON; if the logical does not exist, debug logging is OFF. The equivalence string for the logical is not checked.

Restart the queue. If the OPCOM messages are enabled, a notification is printed telling you that debugging is enabled.

To turn off the `SYMBIONT` log file, deassign the logical and restart the SYMBIONT queue(s) that were (re)started since you defined the logical because the symbiont only looks for the logical when the queue starts up.

```
$ DEASSIGN/SYSTEM IP$LPD_SYMBIONT_DEBUG
```

The log files are created in `IP$SPOOL:LPD_DEBUG_`*`pid`*`.LOG`.

`pid` is the hexidecimal process ID of the symbiont process in question. There may be as many as 16 queues sharing the same log file, since log files are per symbiont process, not per queue, and each process can support processing for up to 16 queues.

The error message `Invalid data file size received`, indicates a LPR client sent an invalid printing job. This could occur if a Windows 2000 LPR client is wrongly configured.

5.2. Configuring Print Queues

This section describes how to configure a TCP/IP print queue using VSI TCP/IP. It supports two printer protocols: STREAM and LPD. Set up print queues using the interactive VSI TCP/IP OpenVMS PRINTER-CONFIG utility, which is used to configure OpenVMS TCP/IP printer queues.

PRINTER-CONFIG creates and maintains configuration information about VSI TCP/IP TCP/IP queues. By default, this utility stores configuration information in the file `IP$:REMOTE-PRINTER-QUEUES.COM`, which also contains the necessary DCL commands to configure the printer pseudo-devices and set up the OpenVMS print queues.

After using this utility to configure the queue, invoke the command procedure `IP$:REMOTE-PRINTER-QUEUES.COM` to create an OpenVMS queue on the system. You can invoke this command procedure at any time to initialize a new queue without affecting any queues already running. Use this queue exactly like any other OpenVMS queue; it responds to standard OpenVMS **PRINT** and **QUEUE** commands. To run the printer queue configuration manager, issue the command:

```
$ IP CONFIGURE/PRINTERS
```

5.2.1. Configuring an LPD Protocol Queue

The LPD protocol is a standard for sharing printers between hosts on an IP network. The protocol is defined by RFC-1179, and is integrated into the OpenVMS printing system with the IP

$LPD_SYMBIONT executable. LPD was originally intended for system-to-system print job transfers, but many printer Ethernet cards now support LPD protocol so that LPD can be used for printing directly to a printer. The following parameters are used with the `IP$LPD_SYMBIONT`:

- `ADDRESS=`*`host_addr`* or `ADDRESS:`*`host_addr`*
- `CLASS=`*`class_string`* or `CLASS:`*`class_string`*
- `FILTER=`*`filter_char`* or `FILTER:`*`filter_char`*
- `NOFFLF=Y/T/1` or `NOFFLF:Y/T/1`
- `PRINTER=`*`remote_queue_name`* or `PRINTER:`*`remote_queue_name`*
- `RETAIN_CR=Y/T/1` or `RETAIN_CR:Y/T/1`

This protocol separates the print job into two parts:

- A df file (data file) that contains the actual data in the print file
- A cf file (control file) that contains commands for how the file should be printed

The df file is sent first and must be spooled on the server until the cf file arrives with instructions for how the server should print the file. This process causes more overhead when printing a file.

If you are using the LPD protocol to print directly to a printer, make sure the printer's network interface supports LPD. Consult the printer interface documentation to confirm this. To initially configure an LPD queue, use the PRINTER-CONFIG **ADD** command. The utility prompts you for the following information:

Remote Host Name:

The IP address of the remote system or printer to which the print job will be sent. The address may be specified in standard "dot" notation (i.e. 123.456.789.012), or as a DNS name if there is an entry in the DNS server or host table of the sending system for the printer or terminal server in question.

Protocol Type: [LPD]

Press **RETURN** to accept the default (LPD protocol).

Remote Queue Name: [lp]

A remote queue name used by the LPD protocol. If this is a system-to-system print setup, the remote queue name must match the name of an existing queue defined on the remote system. If the print jobs will go directly to a printer, the remote queue name often designates the "spooling" area that exists on the printer's network interface card.

The remote printer queue name is not arbitrary; it is specific to the type of network interface installed in the printer. Consult the printer's network interface documentation for the correct remote queue name.

The following example shows the initial configuration of a queue called PH5 that points to an HP4si printer with an internal Jet Direct Card, using LPD protocol.

```
$ IP CONFIGURE/PRINTER
```

```
VSI TCP/IP for OpenVMS Remote Printer Configuration Utility 10.5(20)
[Reading in configuration from IP$:REMOTE-PRINTER-QUEUES.COM]
PRINTER-CONFIG>ADD HP5
[Adding new configuration entry for queue "HP5"]
Remote Host Name: 161.49.2.3 or printer.flowers.com
Protocol Type: [LPD]
Remote Queue Name: [lp] TEXT
[HP5 => 161.44.192.5, TEXT]
PRINTER-CONFIG>SHOW HP5
Queue Name  IP Destination Remote Queue Name
----------  -------------- ----------------
HP5   161.49.2.3     TEXT
PRINTER-CONFIG>EXIT
[Writing configuration to IP$:REMOTE-PRINTER-QUEUES.COM]
```

Invoke the **IP$:REMOTE-PRINTER-QUEUES.COM** command procedure to initialize the printer queue.

5.2.1.1. Input Record Modification

By default, the `IP$LPD_SYMBIONT` does no input record modification. Records are read in, formatted and presented to the output code by the standard OpenVMS printer symbiont library routines. The LPD protocol is implemented in the output routine of the symbiont. There are times, however, when it may be desirable to handle record input differently. For instance, the standard routines treat records with leading form feed (FF) characters differently from other records. Leading and trailing carriage control is stripped from such records, and, if the record consisted only of a FF character, the leading carriage control bytes of the following record are suppressed. In most cases, this yields the desired output on paper, but when it does not, it may be useful to be able to specify other behavior. That is what the LPD symbiont's EMBED_CC and ADD_EOR capabilities are intended to help with.

Note

For more information about print symbionts, input filtering and the handling of various file formats see the *OpenVMS Utility Reference Manual*.

The EMBED_CC and ADD_EOR capabilities are optional, and must be enabled though the use of a new logical name (`IP$LPD_SYMBIONT_FILTER_ENABLED`) to be usable. Without this logical being present in the system logical name table at the time the symbiont process is started, neither of these capabilities will work This logical causes the symbiont process to initialize differently than it does when the logical is not present. All other symbiont capabilities work normally regardless of the presence or absence of this logical.

The EMBED_CC capability allows the carriage control specified for the file to be embedded in the data records prior to their being passed to the main formatting routine of the OpenVMS print symbiont. The records will have no separate carriage control specified once this is done, and will be equivalent to printing an embedded carriage control file. The carriage control specified for each record can be the default provided by the input routines, or it can be specified on a system, queue, or job basis. If the carriage control is specified, this will override whatever implicit carriage control is present in the file being printed.

The ADD_EOR capability allows specification of a byte to be added to the end of each data record. This byte will follow any leading carriage control bytes, the data, and any trailing carriage control bytes if these are embedded, or precede them if they are not.

The EMBED_CC and ADD_EOR capabilities can be used individually or together, or not used at all, as needs dictate. Both require that the IP$LPD_SYMBIONT_FILTER_ENABLED logical be defined when the symbiont process is first started.

Logical names are used to control these two capabilities on a system-wide or queue-specific basis. **PRINT** command parameters are used to control them on a per-job basis. See the appropriate sections below for more information on using these capabilities.

5.2.1.2. Logicals used in controlling EMBED_CC and/or ADD_EOR operations

- IP$LPD_SYMBIONT_FILTER_ENABLED - Define as 1/T/Y to enable EMBED_CC and ADD_EOR capabilities. Example:

  ```
  $ DEFINE/SYSTEM/EXECUTIVE_MODE IP$LPD_SYMBIONT_FILTER_ENABLED Y
  ```

- IP$LPD_EMBED_CC and IP$LPD_*queue*_EMBED_CC - Turns on carriage control (CC) embedding. Value is the specification for the CC longword if it is an 8 digit HEX number, beginning with "0x". If the value is 1/T/Y it enables embedding of the default CC as handed to the Input_File routine. If the value is 0/F/N, embedding is disabled.

  The 8 digit HEX number used to specify an override CC value consists of four bytes, each of which specifies a different aspect of the CC. From lowest order byte (right most) to highest:

  ```
  Byte 1: Repeat count for Leading CC
  Byte 2: Leading CC value
  Byte 3: Repeat count for Trailing CC
  Byte 4: Trailing CC value
  ```

  The CC values are the actual ASCII character code value, except that zero is not "NUL", but is used to specify the sequence "CRLF". Example:

  ```
  $ DEFINE/SYSTEM/EXECUTIVE_MODE IP$LPD_EMBED_CC "0x0D010A01"
  ```

  In the preceding example, the four bytes are shown from the highest order byte (fourth) to the lowest order byte (first), with “0D” representing the fourth byte , “01” representing the third byte, “0A” representing the second byte, and “01” representing the first byte. This value would specify a single leading Line Feed (LF) byte, and a single trailing Carriage Return (CR) byte.

  ```
  $ DEFINE/SYSTEM/EXECUTIVE_MODE IP$LPD_FOOBAR_EMBED_CC "Y"
  ```

  says to embed the default CC for each record, as specified in the OpenVMS Utility Routines manual. For a Carriage-Return Carriage Control file this is a leading LF, trailing CR, with special handling around FF characters.

Note

IP$LPD_SYMBIONT_FILTER_ENABLED must be defined when the symbiont process is started in order for this to be effective.

- IP$LPD_ADD_EOR and IP$LPD_*queue*_ADD_EOR - Specifies that an EOR marker byte is to be added to the end of each input record, and what the value of that marker is to be. The marker is specified as either a two-digit HEX number, prefixed with "0x", or the actual ASCII character to use. Example:

```
$ DEFINE/SYSTEM/EXECUTIVE_MODE IP$LPD_ADD_EOR "0x7C"
$ DEFINE/SYSTEM/EXECUTIVE_MODE IP$LPD_ADD_EOR "|"
```

These two specifications are equivalent, and will result in a vertical bar, or “pipe”, character being added to the end of each record. If CC is being embedded, this byte will appear following the trailing CC byte(s). If CC is not being embedded, this byte will precede any trailing CC byte(s).

Note

`IP$LPD_SYMBIONT_FILTER_ENABLED` must be defined when the symbiont process is started in order for this to be effective.

5.2.1.3. Print parameters used in controlling EMBED_CC and/or ADD_EOR operations

- **/PARAMETER=(EMBED_CC=*value*)**: The job parameter equivalent of `IP$LPD_EMBED_CC`. Value has the same range of options as for that logical.
- **/PARAMETER=(ADD_EOR=*value*)**: The job parameter equivalent of `IP$LPD_ADD_EOR`. Value has the same range of options as for that logical.

Note

`IP$LPD_SYMBIONT_FILTER_ENABLED` must be defined when the symbiont process is started in order for either of these to be effective.

5.2.2. Logical Names Provided for Controlling LPD Print Processing

Three logical names allow you to control LPD queue print job handling. By default, the LPD symbiont passes the letter "f" (for "formatted" file) as the print filter character the LPD server uses when you issue a standard PRINT request to an LPD queue. If you include the /PASSALL qualifier on your print request, the LPD symbiont uses the letter "v" (for "Versatec", that is, binary output) instead. In addition, the LPD symbiont normally strips CR (carriage return) characters from CRLF (carriage return, line feed) line-termination sequences, in keeping with the UNIX notion of LF (Line Feed) as the new-line character.

- The `IP$PRINTER_`*`queuename`*`_DEFAULT_FILTER` logical name allows you to specify the print filter character to use instead of "f" on the specified LPD queue. Declare the alternative, normal print filter character as follows:

  ```
  $ DEFINE /TABLE=IP$PRINTER_TABLE -
  _$ IP$PRINTER_queuename_DEFAULT_FILTER "character"
  ```

  `queuename` is the name of the queue for which you are modifying the print filter character, and *`"character"`* is the character to be used.

  You may override both the default character and the alternative character by including the `/PARAMETER=(FILTER=`*`"character"`*`)` qualifier on your print request (assuming the logical name `IP$PRINTER_`*`queuename`*`_ALLOW_USER_SPEC` is defined appropriately.

  `IP$PRINTER_`*`queuename`*`_ALLOW_USER_SPEC`

If defined with an equivalence string of "Y", "T" or "1", the user can specify the ADDRESS and PRINTER values for the print job on the **PRINT** command, in the **/PARAMETERS** qualifier. In addition, specifying this logical will result in requeue retries, rather than the default timed retries when there is a problem with a connection. If this logical is not defined, or is defined with some other equivalence string, the ADDRESS and PRINTER parameters entered on the **PRINT** command by the user will be ignored, and timed retries will be performed.).

- The `IP$PRINTER_queuename_PASSALL_FILTER` logical name allows you to specify the print filter character to use instead of "v" on the specified LPD queue. Declare the alternative, passall print filter character as follows:

  ```
  $ DEFINE /TABLE=IP$PRINTER_TABLE -
  _$ IP$PRINTER_queuename_PASSALL_FILTER  "character"
  ```

  queuename is the name of the queue for which you are modifying the passall print filter character, and *"character"* is the character to be used.

  You may override both the default character and the alternative passall character by including both the **/PARAMETER=(FILTER="*character*")** qualifier and the **/PASSALL** qualifier on your print request.

- The `IP$PRINTER_queuename_RETAIN_CR_DEFAULT` logical name allows you to specify the disposition of CR characters in CRLF sequences on the specified LPD queue. Change CR processing on the queue as follows:

  ```
  $ DEFINE /TABLE=IP$PRINTER_TABLE -
  _$ IP$PRINTER_queuename_RETAIN_CR_DEFAULT "boolean"
  ```

  queuename is the name of the queue for which you are modifying the CR disposition.

  "boolean" indicates whether or not CR characters are to be retained in CRLF sequences included in text sent to the remote system for printing.

  You may override both the default CR disposition and the alternative disposition by including the **/PARAMETER=(RETAIN_CR="*boolean*"** qualifier on your print request.

- The `IP$LPD_SYMBIONT_DEBUG` logical name enables debug logging.

- The `IP$PRINTER_queuename_SUPPRESS_FF` logical name controls whether CRFF is added to jobs.

- The `IP$PRINTER_queuename_NO_TELNET` logical name controls Telnet IAC code expansion.

If you want an extra blank line on each page and, consequently, an extra blank page when the bottom margin has been reached, set the logical `IP$NLPx_REMOTE_PRINTER` to include the configuration parameter DOLFFF=Y. Depending on your printer, it may be desirable to keep the behavior and not have the extra blank line and extra blank page.

5.2.2.1. Using Retry Timers

When the symbiont cannot connect to the remote system, or the remote LPD server reports insufficient resources for printing a job, the symbiont requeues the job for a later attempt. Requeue attempts are reported directly to the user who submitted the print job only if the user specified /

NOTIFY in the print command. The requeue time is controlled through logical names; you can control the length of time a job will wait before being attempted again after a connection failure by defining a logical name as follows:

```
$ DEFINE/SYSTEM IP$LPD_SYMBIONT_RETRY_INTERVAL "delta-time"
```

The default value is "0 00:10:00.00", or ten minutes.

You can also control the maximum amount of time that should elapse before the symbiont gives up on a job with this command:

```
$ DEFINE/SYSTEM IP$LPD_SYMBIONT_MAX_RETRY_INTERVAL "delta-time"
```

The default value is "0 02:00:00.00", or two hours.

You must specify the delta-time values in quotation marks, and with a space separating the number of days from the number of hours, so the symbiont can process them correctly.

LPD uses timer retries to queues that are not set up to allow user-specified printer destinations. For the other queues, jobs are placed in the queue to be tried again. The advantage to the timer retries is that successive jobs are not sent to the printer until the symbiont can actually contact the printer. Use the following logical to control timer retry intervals:

```
$ DEFINE/SYSTEM/EXEC IP$LPD_SYMBIONT_CONNECT_TIMERS "n n"
```

Its equivalence string should be two numbers, separated by one space, that specify a) the retry interval and b) the maximum retry time, in seconds. By default, the interval starts at 600 seconds (10 minutes); for each retry, that value is doubled until the maximum retry time of 7200 seconds (2 hours) is reached. The default would be represented by the following logical definition:

```
$ DEFINE/SYSTEM/EXEC IP$LPD_SYMBIONT_CONNECT_TIMERS "600 7200"
```

- OPCOM messages from the LPD symbiont now include the queue name and entry number associated with the message.
- OPCOM messages from the LPD symbiont can be disabled by defining the following logical:

  ```
  $ DEFINE/SYSTEM/EXEC IP$PRINTER_NO_OPCOM true
  ```

It is recommended that the OPCOM messages not be disabled. They may block legitimate problem messages, in addition to informational messages about trying connections. This logical can be used for both the LPD and STREAM symbionts.

`IP$LPD_SYMBIONT_LFTAIL` and `IP$LPD_SYMBIONT_*_LFTAIL` allow reversion to legacy behavior of terminating jobs with an <LF> rather than <CR>. To enable this behavior, use one of these values: Y, T, or 1.

`IP$LPD_MAXSTREAMS` specifies the maximum number of streams each symbiont process will handle.

`IP$LPD_KEEPALIVE` turns on keepalives when making socket connections. To enable this behavior, use one of these values: 1, y, Y, t, or T.

`IP$LPD_SYMBIONT_RESOURCE_TIMERS` specifies the initial and maximum resource retry delay times.

5.2.2.2. Adding Print Queue Parameters

You can define queue parameters on each VSI TCP/IP queue using the **IP CONFIGURE/PRINTER** utility. Refer to the *VSI TCP/IP Administrator's Reference* for specific parameters. The following example shows how to add a device control library called HP3SI to the queue called HP5.

```
$ IP CONFIGURE/PRINTER
VSI TCP/IP for OpenVMS Remote Printer Configuration Utility 10.5(20)
[Reading in configuration from IP$:REMOTE-PRINTER-QUEUES.COM]
PRINTER-CONFIG>SELECT HP5
[The Selected Printer is now HP5]
PRINTER-CONFIG>SET LIBRARY HP3SI
[Library HP3SI]
PRINTER-CONFIG>SHOW HP5
Queue Name  IP Destination Remote Queue Name
----------  --------------     -----------------
HP5   161.49.2.3     TCP port 9100
Device Control Library = HP3SI
PRINTER-CONFIG>EXIT
[Writing configuration to IP$:REMOTE-PRINTER-QUEUES.COM]
```

Remember to invoke the **REMOTE-PRINTER-QUEUES.COM** procedure after exiting the configuration utility to add the parameters to the queue. Some parameters are only valid with the **INITIALIZE /QUEUE** command, so you may need to first stop and delete the queue, then invoke the **REMOTE-PRINTER-QUEUE** command procedure before the new parameters take effect.

5.2.2.3. Starting Multiple Print Queues

You can start either a single print queue or a list of queues, in addition to starting all print queues using the command procedure **IP$:REMOTE-PRINTER-QUEUES.COM**. The syntax for the command procedure is:

```
$ @IP$:REMOTE-PRINTER-QUEUES [queue1,[queue2,queue3,...]]
```

If you do not provide any queue names on the command line, the procedure attempts to start all defined queues. For example, to start SYS$PRINT:

```
$ @IP$:REMOTE-PRINTER-QUEUES SYS$PRINT
```

Or, to start SYS$PRINT and SYS$LASER:

```
$ @IP$:REMOTE-PRINTER-QUEUES SYS$PRINT,SYS$LASER
```

5.2.2.4. Using User-Specified Print Destinations

VSI TCP/IP allows users to override some of the parameters that you specify when you configure a print queue. You can enable user-specified print destinations, using the ADDRESS and PRINTER parameters, on a per-queue basis:

```
$ DEFINE/TABLE=IP$PRINTER_TABLE -
_$ IP$PRINTER_queuename_ALLOW_USER_SPEC TRUE
```

If you want to allow users to specify their own print destinations for LPD printing, define this logical name during your system startup sequence.

To define the logical using IP CONFIG:

```
$ IP CONFIGURE /PRINTER
```

```
PRINTER-CONFIG>SELECT queuename
PRINTER-CONFIG>SET ALLOW-USER-SPECIFIC ENABLE
PRINTER-CONFIG>EXIT
```

To override the parameters associated with an LPD queue, use the **/PARAMETERS** qualifier with the **PRINT** command. Specify an alternative destination, a print filter, or how carriage return characters are to be treated with these parameters:

/ADDRESS=*n.n.n.n*

Specifies the address of the destination host. This value must be a numeric IP address, not a host name. If not specified, the address configured for the queue in the VSI TCP/IP printer configuration utility (PRINTER-CONFIG) is used.

/CLASS=*class_string*

Specifies the string to put on the "class" line in the control file. This gets used in various ways, but mostly appears on the banner page if one is printed.

/PRINTER=*name*

Specifies the name of the printer on the destination host. Use quotation marks around the entire parameter specification, since many systems are case-sensitive regarding printer names. If you specify ADDRESS and omit this parameter, the printer name defaults to "lp." If you omit both ADDRESS and PRINTER, the printer name configured for the queue in the VSI TCP/IP printer configuration utility is used.

/RETAIN_CR={Y | N}

Specifies whether the symbiont should not convert CR/LF sequences into bare LFs when sending a text file to the remote system. The default is N; CR/LF sequences are converted to bare LFs. A setting of Y means the sequences are not converted. This parameter is ignored if you submit the job with the **/PASSALL** qualifier.

/FILTER=*x*”

Specifies the "print filter" character to be sent with the print job to the remote system. Use quotation marks around the entire parameter, since filter specifications are generally case-sensitive. By default, the symbiont uses "f" for text files and "v" for files submitted with the **/PASSALL** qualifier.

RESTART

Restarts the queues with the modified configuration information.

NOFFLF=Y/T/1

Specifies whether the symbiont does not add a Line Feed after a Form Feed when sending a text file to the remote system.

For example, to print to an alternate destination:

```
$ PRINT/PARAM=(ADDRESS=192.168.34.22,"PRINTER=HP5",RETAIN=Y,FILTER="1")-
_$ LOGIN.COM
```

If the case of the printer name or filter character must be preserved, these parameter values must be enclosed in quotes. In the preceding example, the filter character has been specified as a lowercase “l”.

In the example, the LPD client symbiont is instructed to send the file to the LPD server on host 192.168.34.22 for queue HP5, retaining carriage return characters, and using a filter of "l". The queue entry in the local OpenVMS LPD client queue looks like:

```
$ SHOW QUEUE/FULL SYS$PRINT
Printer queue SYS$PRINT, stopped, on NODE:NLP8, mounted form DEFAULT
 /BASE_PRIORITY=4, /DEFAULT=(FORM=DEFAULT) /OWNER=[SYSTEM]
 /PROCESSOR=IP$LPD_SYMBIONT /PROTECTION=(S:M,O:D,G:R,W:S)
 Entry   Jobname   Username   Blocks   Status
 -----   -------   --------   ------   ------
 795     LOGIN     USER       9        Printing
 Submitted 23-MAY-2017 09:40 /FORM=DEFAULT
 /PARAM=("ADDRESS=192.168.34.22", "PRINTER=HP5", "RETAIN=Y", "FILTER=1")
 /PRIORITY=100
 File: _DKA100:[USER]LOGIN.COM;1
```

5.2.2.5. Customizing Printer Queues

You can add special characteristics to queues by using customized command procedures. You can do this globally to all STREAM or LPD queues, or limit them to each individual queue. These command procedures are automatically invoked every time **REMOTE-PRINTER-QUEUES.COM** is invoked.

A customized command procedure must contain all commands to spool the NLP device to the queue and to initialize the queue. Review the command procedure **IP$:REMOTE-PRINTER-QUEUES.COM** to see how and when these customized command procedures are called.

- To add a special queue characteristic or qualifier to all STREAM queues, create a command procedure called **INITIALIZE_STREAM_QUEUE.COM** in the IP directory.
- To customize all LPD queues, create a file called INITIALIZE_LPD_QUEUE.COM in the IP directory.
- To customize individual queues, create a file called **INITIALIZE_*queuename*.COM** in the IP directory.

The following example shows a customized command procedure to add the **/SEPARATE** qualifier to a queue called HP5. The file is located in the IP directory and is called `INITIALIZE_HP5.COM`.

```
$! Custom initialization procedure for stream queue HP5
$ NLP_Device = P1
$ Remote_Address = P2
$ Remote_port = P3
$ Default_Form = P4
$ If F$GetDVI(NLP_Device,"SPL") Then Set Device/NoSpool 'NLP_Device'
$ Set Device/Spool=HP5 'NLP_Device':
$ Initialize/Queue/Processor=IP_Stream_Symbiont HP5 -
/On='NLP_Device'/Start/library=hp3sidevctl/separate=(reset=(reset))
$ Exit
```

Note

The customized command procedure must contain all the commands to spool the NLP.

The NLP device must be unique for each queue. The **IP$:REMOTE_PRINTER_QUEUES.COM** command procedure chooses the next available NLP device when initializing the queue. To determine the correct NLP device, use PRINTER-CONFIG to add a new queue. After exiting the utility,

examine the **IP$:REMOTE_PRINTER_QUEUES.COM** command procedure to determine which NLP device was assigned to the new queue. Use the commands in that command procedure as a template for creating the customized command procedure.

Note

REMOTE_PRINTER_QUEUES.COM passes the NLP device name as one of several parameters to the customized command procedure.

5.3. Configuring a STREAM Protocol Queue

The print client STREAM protocol is a TELNET-based protocol that is not defined by an RFC. This protocol is integrated into the OpenVMS printing system with the `IP$STREAM_SYMBIONT` executable.

To initially configure a STREAM queue, use the PRINTER-CONFIG **ADD** command. The **ADD** command has a maximum limit of 5000 symbionts per system. The utility prompts you for the following information:

Remote Host Name:

The IP address of the printer or of the terminal server to which the printer is attached. The address may be specified in standard "dot" notation (i.e., 123.456.789.012), or as a DNS name if there is an entry in the DNS server or host table of the sending system for the printer or terminal server in question.

Protocol Type: [LPD]

Type the word STREAM to indicate STREAM protocol.

TCP Port Number: [23]

The port number to which the TCP/IP connection will attach. If the printer has an Ethernet card installed in it, the port number is listed in the Ethernet card documentation; check there first, or call the card manufacturer to confirm the port number. For example, HPLJ Jet Direct cards use port 9100 for the TCP port number.

If the printer is attached to a terminal server, the port number refers to the physical port on the terminal server to which the printer is attached. For example, the HP 90TL terminal server adds 2000 to the actual physical port number to address the correct port. If the printer is physically attached to port 3, add 2000 to 3, making the port number 2003. Direct any questions on addressing the correct port number to the appropriate terminal server vendor.

The following example shows the initial configuration of a queue called HP5 that points to an HP4si printer with an internal HP Jet Direct Card, using STREAM protocol. (For terminal servers, such as the HP 90TL, make sure the logical name `IP$PRINTER_`*`printername`*`_NO_TELNET` is enabled. where *`printername`* is the name that was given to the printer in **IP CONFIGURE /PRINTERS**.)

```
$ IP CONFIGURE/PRINTER
VSI TCP/IP for OpenVMS Remote Printer Configuration Utility 10.5(20)
[Reading in configuration from IP$:REMOTE-PRINTER-QUEUES.COM]
PRINTER-CONFIG>ADD HP5
```

```
[Adding new configuration entry for queue "HP5"]
Remote Host Name: 192.168.2.3 or printer.flowers.com
Protocol Type: [LPD] STREAM
TCP Port Number: [23] 9100
[HP5 => 192.168.2.3, TCP port 9100 (no telnet option negotiation)]
PRINTER-CONFIG>SHOW HP5
Queue Name   IP Destination   Remote Queue Name
----------   --------------   -----------------
HP5          192.168.2.3      TCP port 9100
 Telnet Options Processing will be suppressed
PRINTER-CONFIG>EXIT
[Writing configuration to IP$:REMOTE-PRINTER-QUEUES.COM]
```

Invoke the **IP$:REMOTE-PRINTER-QUEUES.COM** command procedure to initialize the printer queue.

5.3.1. Troubleshooting a STREAM Protocol Queue

To generate a log file with STREAM, do the following:

```
$ DEFINE/SYSTEM IP$STREAM_SYMBIONT_DEBUG
```

A value is not required here as with NTYSMB. If the logical exists, debug logging is ON; if the logical does not exist, debug logging is OFF. The equivalence string for the logical is not checked.

Restart the queue. If the OPCOM messages are enabled, a notification is printed telling you that debugging is enabled.

To turn off the STREAM log file, deassign the logical and restart the STREAM queue(s) that were (re)started since you defined the logical because the symbiont only looks for the logical when the queue starts up.

```
$ DEASSIGN/SYSTEM IP$STREAM_SYMBIONT_DEBUG
```

The log files are created in `IP$SPOOL:STREAM_DEBUG_`*`pid`*`.LOG`.

`pid` is the hexidecimal process ID of the symbiont process in question. There may be as many as 16 queues sharing the same log file, since log files are per symbiont process, not per queue, and each process can support processing for up to 16 queues.

5.3.2. Logical Names Provided for Controlling STREAM Processing

- The `IP$STREAM_SYMBIONT_DEBUG` logical name enables debug logging.
- The `IP$PRINTER_`*`queuename`*`_SUPPRESS_FF` logical name controls whether CRFF is added to jobs.
- The `IP$PRINTER_`*`queuename`*`_NO_TELNET` logical name controls Telnet IAC code expansion.
- If `IP$STREAM_DO_ASYNC_LOOKUP` is not defined to 1, True, or Yes, then the stream print symbiont will now resolve the remote host name with getaddrinfo, and is capable of resolving and connecting to an IPv6 address.

  ```
  $ DEFINE/SYSTEM/EXEC IP$LPD_DO_ASYNC_LOOKUP TRUE
  ```

- The IP$STREAM_DEAD_LINK_TIMEOUT and IP $STREAM_*queuename*_DEAD_LINK_TIMEOUT logical name control dead link detection and handling.

The IP$STREAM_SYMBIONT can be configured for detection and handling of "dead links", that is, a TCP/IP link that stops responding, but was not closed properly by the remote end.

To enable dead link detection, define a system logical:

```
$ DEFINE/SYSTEM IP$STREAM_DEAD_LINK_TIMEOUT "timeout_secs [[<requeue_secs>]
 <option>]"
```

or:

```
$ DEFINE/SYSTEM IP$STREAM_queuename_DEAD_LINK_TIMEOUT "timeout_secs
 [[<requeue_secs>] <option>]"
```

Both of the *_secs* values are integers specifying seconds. *Timeout_secs* is the number of seconds a write has to take before it is considered a dead link. *Requeue_secs* specifies how many seconds to hold a job if it is requeued due to a dead link. *Option* is one of the following:

- "REQUEUE" — requeue the job with a hold for requeue_secs.
- "CONTINUE" — continue with job, open a new link.
- "STOP" — stop the queue.

The default behavior is "REQUEUE" if none is specified explicitly, and the default *requeue_secs* is zero (that is, no delay) if no time is specified.

Note

None of these guarantees that a print job will not be affected adversely by a lost link, especially when it is due to the printer interface being powered off suddenly.

A problem can occur when the symbiont tries to open a connection and the remote host refuses the connection. Possible reasons for this refusal could be that there is another connection already open on that port by another system or the connection from the previous job the symbiont sent has not finished closing. You can control how long the symbiont waits to retry a connection after it is refused with the IP$STREAM_SYMBIONT_TIMERS logical. The logical sets the initial and the maximum time to wait before retrying the connection. For example, if you defined the logical as follows it would retry the connection after 1 second and double the time between subsequent retries until it reached the maximum of 10 seconds.

```
$ DEFINE/SYSTEM/EXEC IP$STREAM_SYMBIONT_TIMERS "1 10"
```

5.4. LPD and Stream Symbiont User Exit Support

A user exit is a mechanism for customizing the way the print symbiont sends jobs to the printer, beyond the methods provided by VSI TCP/IP. A user exit is a shareable image that contains symbols the symbiont reads at runtime for special processing instructions.

The user exits in the LPD symbiont have been modified for the current release of VSI TCP/IP.

The file IP$ROOT:[IP.EXAMPLES]USER_LPD_CLIENT.C (it is provided in the distribution kit) contains the template for an LPD symbiont customization image. This interface is subject to change without notice. VSI does not support customer modifications to these interfaces. VSI does not guarantee future changes will not be made that could break a customer's modifications. The definitions in it have changed since the last release to better support printing with DCPS symbiont.

The file shows how to modify the way the LPD protocol symbiont generates the control file it sends over the network, and the name and Internet address of the server to which it connects.

Note

If you use user exits, you must update your source code, recompile, and relink your routines.

See the comments in the beginning of the file for information on compiling and linking the code properly. Install the routine by placing USER_LPD_CLIENT.EXE in the IP$: directory. The customized image will be used the next time the symbiont starts.

The Stream symbiont supports user exits provided in the IP$ROOT:[IP.EXAMPLES] directory (available only if the library and include files are installed during the VSI TCP/IP installation procedure). The Stream user exit is named USER_STREAM_CLIENT.C.

The LPD user exit takes the first parameter specified in the OpenVMS **PRINT /PARAMETER=** ***qualifier*** and maps it to the job classification (emulating the UNIX **lpr -C** option). After adding changes to an exit file, compile the file, link it with the **/SHARE** qualifier, and copy the resulting executable into the common directory IP$COMMON_ROOT:[IP].

Note

With the current release of VSI TCP/IP, USER_LPD_CLIENT.C supports the addition of a linefeed after a formfeed. You use logical names and print parameters to control the behavior. Use a logical name of the form IP$PRINTER_*queuename*_NO_FFLF_DEFAULT:

queuename is the name of the queue you are using or is the * character to designate all LPD queues defined in the logical name table IP$PRINTER_TABLE to execute each time VSI TCP/IP starts:

```
$ DEFINE/EXECUTIVE/TABLE=IP$PRINTER_TABLE ...
```

or use the NOFFLF parameter in a print command to control the behavior for a particular queue:

```
$ PRINT/PARAMETERS=(NOFFLF=TRUE) filespec
```

The LPD and Stream symbionts have been enhanced in the following ways:

- The first 16 bytes of the work area (the per-stream data area) in the symbionts are reserved for customer use.
- The customizable psm_max_streams_v33() routine is called by the symbionts to determine the number of streams that a symbiont should initialize.
- International character set translations are supported with the shareable user exit image IP $:USER_TRANSLATE.EXE. After adding changes to a customizable file, compile the file and link it with the /SHARE qualifier. (You can request a sample USER_TRANSLATE.C source module from VSI Technical Support.)

- All VSI TCP/IP LPD and stream symbionts can be configured with a remote printer or host's domain name in addition to its IP address.

5.5. Using the NTYSMB Symbiont for Remote, TCP-Connected Printers

VSI TCP/IP V10.5 provides a print symbiont for sending print jobs to remote TCP-connected printers. This – NTYSMB – symbiont, which works in conjunction with network terminal port (NTY) devices, can be used instead of VSI TCP/IP STREAM print queues. The NTYSMB symbiont:

- Allows for user exits. See file `USER_NTYSMB_CLIENT.C` in the `IP$ROOT:[IP.EXAMPLES]` directory.
- Sends a form feed at the end of a job.
- Corrects timer handling in the case where the maximum timeout is reached. The timers are controlled by two values taken from the equivalence string for the `IP$NTYSMB_TIMERS` logical name, *`initial`* and *`ceiling`*. The values for *`initial`* and *`ceiling`* are given in seconds. The *`initial`* value is how soon, after the first connection attempt fails, the symbiont is to retry the connection. On subsequent connection failures, the symbiont backs off its retries exponentially, until it is only retrying every *`ceiling`* seconds. By default, *`initial`* is 10 seconds and *`ceiling`* is 7200 seconds (2 hours). The NTYSMB symbiont never gives up on a job. Define the logical name as:

  ```
  $ DEFINE/SYSTEM/EXEC IP$NTYSMB_TIMERS "initial ceiling"
  ```

- Zeros out channel information in case a write request is received when a shutdown or close is in progress.
- Fixes queue shutdown when a timed retry is outstanding and a STOP /REQUEST is issued against the queue.
- Corrects I/O synchronization problems where data could be sent to the printer out of order.

`IP$NTYSMB_DEBUG` causes various debug options to be enabled.

`IP$NTYSMB_*_MAXTIMERMSG` and `IP$NTYSMB_`*`queuename`*`_MAXTIMERMSG` specifies the message to be issued when the connection timer hits the maximum value. One "%s" argument will be supplied in the form of the queue name.

5.5.1. NTYSMB Advantages Over STREAM Queues

NTY queues have the following advantages over STREAM queues:

1. Improved print formatting: The standard OpenVMS print symbiont normally takes advantage of formatting capabilities in the OpenVMS terminal driver. These capabilities were only partially emulated by the STREAM symbiont. Because `IP$NTYSMB` uses network terminal port devices, full print symbiont formatting is now supported.
2. More control over queues' devices: The standard SET TERMINAL and queue operation commands are used to set up the NTY terminal device and `IP$NTYSMB` print queues. System managers wanting to take advantage of the OpenVMS queue manager's Autostart capability may now do so with this new print queue support.

3. Familiar management interface: The NTY Control Program (NTYCP) and `IP$NTYSMB` print symbiont are patterned after HPE LAT Control Program (LATCP) and LAT print symbiont, providing a management interface that should be familiar to many OpenVMS system managers.

5.5.2. Setting Up a Print Queue with IP$NTYSMB

Note

VSI TCP/IP must be started before NTY devices can be created or IP$NTYSMB print queues can be initialized or started. System managers using the OpenVMS queue manager's Autostart capability *must* leave Autostart *disabled* until after VSI TCP/IP is started and NTY devices have been set up.

To set up a print queue with IP$NTYSMB:

1. Create the NTY device.

   Use the NTY Control Program (NTYCP) to create the terminal device:

   ```
   $ NTYCP := $IP$:NTYCP
   $ NTYCP CREATE PORT NTYnnnn /NODE=node/port
   ```

   You can invoke the NTYCP program as an OpenVMS "foreign" command, as shown above, or run in interactive mode:

   ```
   $ RUN IP$:NTYCP
   NTYCP>CREATE PORT NTYnnnn /NODE=node/port
   NTYCP>EXIT
   ```

   NTYCP uses standard DCL-style command parsing. The "?" help feature available in other VSI TCP/IP utilities is not available in NTYCP.

2. Set up the terminal characteristics:

   ```
   $ SET TERMINAL NTYnnnn:/PERMANENT/NOBROADCAST/NOTYPEAHEAD/NOWRAP/FORM
   ```

3. Set up spooling, if desired:

   ```
   $ SET DEVICE/SPOOLED=queue-name,SYS$SYSDEVICE:) NTYnnnn
   ```

4. Initialize and start the queue:

   ```
   $ INITIALIZE/QUEUE/ON=NTYnnnn:queue-name/PROCESSOR=IP$NTYSMB/START
   ```

   The following is an example of a print queue set up to an HPE LaserJet printer with a JetDirect card.

   ```
   $ NTYCP := $IP$:NTYCP
   $ NTYCP CREATE PORT NTY1001/NODE=hp-laserjet/PORT=9100
   %NTYCP-S-CREPORT, device _NTY1001: created to host 192.1.1.5, port 9100
   $ SET TERMINAL/PERMANENT NTY1001:/NOBROADCAST/NOTYPEAHEAD/NOWRAP/FORM
   $ INITIALIZE/QUEUE/ON=NTY1001: HP_LASERJET/PROCESSOR=IP$NTYSMB/START
   ```

5.5.3. Troubleshooting the IP$NTYSMB

To generate a log file with NTYSMB, do the following:

```
$ DEFINE/SYSTEM IP$NTYSMB_DEBUG 7
```

The equivalence string for this logical is an integer. Various bits in this value control different aspects of debug logging. A value of *7* enables full debug logging, and is the recommended setting.

Restart the queue. If the OPCOM messages are enabled, a notification is printed telling you that debugging is enabled.

To turn off the NTYSMB log file, deassign the logical and restart the NTYSMB queue(s) that were (re)started since you defined the logical because the symbiont only looks for the logical when the queue starts up.

```
$ DEASSIGN/SYSTEM IP$NTYSMB_DEBUG
```

The log files are created in `IP$SPOOL:NTYSMB_DEBUG_`*`pid`*`.LOG`.

`pid` is the hexidecimal process ID of the symbiont process in question. There may be as many as 16 queues sharing the same log file, since log files are per symbiont process, not per queue, and each process can support processing for up to 16 queues.

Since these log files are intended to assist VSI in deciphering problems, their output might not make sense to you.

5.6. Troubleshooting the Print Queue

This section describes potential print queue problems and recommended solutions.

- When executing **REMOTE-PRINTER-QUEUES.COM**, the following messages may appear:

  ```
  %SET-E-NOTSET, error modifying NLPx:
  -SYSTEM-W-DEVASSIGN, device has channels assigned
  %SET-E-NOTSET, error modifying NLPx
  -CLI-E-DEVALSPL, device already spooled
  %JBC-I-QUENOTMOD, modifications not made to running queue4
  ```

  These messages occur when the command file tries to modify an existing queue. The messages are only warnings and do not affect the queues in any way.

- Another situation may occur where the print job is "stair-stepping" down the page. It appears as if the job contains carriage line feeds, but no carriage returns.

  - If the queue is set up as an LPD queue pointing to a printer, the wrong remote queue name has been specified in the printer configuration. Check the documentation for the printer's Ethernet card for the correct remote queue name. (See Section 5.2.1.)

  - If the queue is set up as a STREAM queue pointing to a HPE LaserJet, the problem is the line termination setting on the printer. Most HPLJ printers are initially configured with line termination set to <LF>=<LF>. For printing from OpenVMS systems, the line termination should be set to <LF>=<LF>+<CR>. Consult the printer documentation about changing this parameter.

- The print job prints all on one line.

  If this is a STREAM queue pointing to a printer connected to a terminal server, the terminal server characteristics must be modified. Check the terminal server documentation for the proper commands and instructions for logging onto the terminal server. The following example shows the commands for a Hewlett-Packard 90TL terminal server.

```
CHANGE PORT port# TELNET SERVER NEWLINE TO HOST <CRLF>
CHANGE PORT port# TELNET SERVER NEWLINE FROM TERMINAL <CRLF>
```

- The queue prints text properly, but graphics do not print correctly.

  If the TELNET negotiation option is ON, you may need to disable it with the **SET SUPPRESS-TELNET** parameter described in the *VSI TCP/IP Administrator's Reference*.

  The default is to set TELNET negotiation OFF by default, unless it is using the default port 23. The following example shows how to disable TELNET negotiation.

  ```
  $ IP CONFIGURE/PRINTER
  PRINTER-CONFIG>SEL queue-name
  PRINTER-CONFIG>SET SUPPRESS-TELNET ENABLE
  PRINTER-CONFIG>EXIT
  ```

  Be sure all privileges are enabled; then invoke **IP$:REMOTE-PRINTER-QUEUES.COM**.

- PostScript files do not print.

  When printing PostScript files, set the queue to NOFEED. Remove any FLAG pages from the queue. See the *VSI TCP/IP Administrator's Reference* for information about the SET NOFEED and SET FLAG commands. You should also specify that the OpenVMS form on which the job is printed be set to the maximum width, and defined with /NOWRAP and /NOTRUNC. Use the following commands to determine the characteristics of the form being used on the queue:

  ```
  $ SHOW QUEUE/FORM/FULL form-name
  ```

  An example of the characteristics on a form called POSTSCRIPT is shown in the following example:

  ```
  Form name                   Number   Description
  ---------                   ------   -----------
  POSTSCRIPT (stock=DEFAULT)  3        Postscript input; white paper
     /LENGTH=66 /MARGIN=(BOTTOM=6) /STOCK=DEFAULT /WIDTH=65535
  ```

5.7. Internet Printing Protocol (IPP)

The IPP print symbiont is an OpenVMS print symbiont working with the OpenVMS printing subsystem to implement an IPP Client. It allows printing over a network to printers and servers that support the IPP v1.0 network printing protocol. The user interface is similar to other print symbionts in that it uses PRINT commands or system library calls to submit jobs to print queues. The IPP protocol has specific qualifier values and queue settings that must be present to allow the symbiont to function. This section describes both the configuration of IPP print queues and the use of the PRINT command. For information on submitting jobs to print queues using system library calls, see the appropriate OpenVMS documentation.

5.7.1. IPP Protocol Background

The Internet Printing Protocol solves a number of problems in existing network printing protocols; the most common is the LPD protocol, originally implemented on UNIX operating systems. For more information see: https://www.ietf.org/.

IPP has a better error reporting capability than LPD or TELNET. It supports multi-sided printing, landscape/portrait layouts, and multiple pages per physical sheet ("number-up") printing. Because not

all printer models that support IPP will support all capabilities, the IPP protocol includes a way for the symbiont to query the printer as to its capabilities before a job is sent. If the printer cannot handle a given request, the job is aborted with an error status. The error status appears in the system accounting log.

IPP uses the HTTP 1.1 protocol as its transport layer; however, it has its own reserved port number, port 631. You can use the IPP Symbiont to print to other port numbers, including the standard HTTP port (80), but you need to specify the port number as part of the printer URL if the port number is not the default IPP port number. If you are printing through a firewall this could be a factor to consider. For a full description of the IPP protocol, see the relevant RFCs listed below.

The IPP Printer Working Group has a web page with at least a partial list of printers that claim to support IPP. For more information see: http://www.pwg.org/ipp/index.html.

5.7.1.1. Relevant RFCs

The RFCs related to IPP v1.0 are:

Design Goals for an Internet Printing Protocol	RFC 2567
Rationale for the Structure and Model and Protocol for the Internet Printing Protocol	RFC 2568
Internet Printing Protocol/1.0: Model and Semantics	RFC 2566
Internet Printing Protocol/1.0: Encoding and Transport	RFC 2565
Internet Printing Protocol/1.0: Implementer's Guide	RFC 2639
Mapping between LPD and IPP Protocols	RFC 2569

Additional RFCs are referenced by these, such as the ones describing HTTP v1.1, MIME Media Types, etc. The specific RFCs are called out in the above documents.

5.7.1.2. Limitations of this Implementation

The IPP symbiont implements a subset of the IPP v1.0 protocol consisting of all required portions and several selected optional features. Note that not all features are available on all printers; most printers implement a subset of the available protocol capabilities.

Not all printers claiming to support IPP implement IPP correctly. Some use supersets of HTTP 1.0, rather than the required HTTP 1.1. Some do unusual things with TCP/IP connections, such as having extremely short timeouts. The symbiont has been adapted to support as many of these inconsistencies as possible. The symbiont may or may not behave as expected with such printers depending on your particular network characteristics and exactly what the printer manufacturer has done differently from what is specified in the RFCs. The symbiont should work with any fully compliant IPP v1.0 printer or server.

5.7.2. Configuration

The IPP symbiont has a flexible configuration. You can supply the information in the queue setup itself, as the **/DESCRIPTION=** string which is supported by OpenVMS as part of the **INITIALIZE /**

QUEUE command. You can supply the information in a `configuration` system logical name that the symbiont checks. You can use both, putting some information on one place, and some in the other. You can also put configuration information in one or more files and reference those files from the **/DESCRIPTION** string and/or `configuration` logical name, or even from other such files. If you have large numbers of queues making up complicated groupings with similar requirements, this flexibility can help reduce the time required to set up and maintain queues.

In addition to the basic configuration information, there are several optional logical names used to control specific behaviors. Note that the default behaviors may be adequate to your needs.

The following sections describe the various logical names, queue settings, and **PRINT** command options available with the IPP symbiont. In many cases there is a "Global" version, that affects all IPP symbiont queues on the system, as well as a "queue-specific" version that affects only a specified queue. **PRINT** command options affect only the job being submitted.

5.7.2.1. Global Settings

These logical names establish configuration values for all queues on the system, not on a queue-by-queue basis. Where there are queue-specific settings related to these, these become the default values, overriding any built-in defaults.

`IP$IPP_CONFIG`

Specifies one or more of the qualifiers described in Section 5.7.2.2. These qualifiers are not case sensitive. Underscores (_) in the qualifier names are optional. Each may be abbreviated as long as the result is not ambiguous. There is no default. This logical provides defaults that may be overridden by the queue-specific configuration logical, `IP$IPP_`*`queuename`*`_CONFIG`, for a given queue.

`IP$IPP_DEFAULT_DOCUMENT_FORMAT`

Specifies a string to use as the document format, unless specified differently for a given queue or print job. The actual document format used on a given job must be a valid MIME media type, supported by the printer to which the job is sent. The default is "text/plain".

`IP$IPP_DOCALIAS_FILE`

Specifies document format name aliasing. Rather than having to specify long mime-media-type names for document formats, you can define local names that are equivalent, and the symbiont will do the replacement. For example, you can define “PS” as equivalent to “application/postscript”, and use it in print commands as **/DOCUMENT_FORMAT=PS**. There is an escape mechanism in case a logical name is ever made into a different MIME-media-type. Prefixing the document format name with % prevents alias translation. %PS means just send it as PS, do not translate PS into APPLICATION / POSTSCRIPT in the request.

To use aliasing, define the system logical name `IP$IPP_DOCALIAS_FILE` with the filename of the alias file as the equivalence string. The format of the alias file is:

`IP$IPP_`*`mime-name`*

Blank lines are ignored. Lines starting with # are treated as comments and are ignored. Document format name aliasing has been added. Rather than having to specify long mime-media-type names for document formats, you can now define local names that are equivalent, and the symbiont will do the replacement. For example, you can define "PS" as equivalent to "application/postscript", and use it in print commands such as: **/DOCUMENT_FORMAT=PS**.

There is an "escape mechanism" in case a local name is ever made into a different MIME-media-type. Prefixing the document format name with "%" prevents alias translation. "%PS" means "just send it as 'PS', don't translate "PS" into "application/postscript" in the request.

To use aliasing, define the system logical name IP$IPP_DOCALIAS_FILE with the filename of the alias file as the equivalence string.

The format of lines in the alias file is:

```
Mime-name: alias, alia, alias...
```

Blank lines are ignored, and lines starting with "#" are treated as comments and are ignored.

IP$IPP_IGNORE_DESCRIPTION

IP$IPP_*queuename*_IGNORE_DESCRIPTION

If this logical is defined, the symbiont ignores the **/DESCRIPTION** strings for all IPP queues. This allows use of **/DESCRIPTION** for other information without affecting the symbiont. Configuration of the symbiont must be done through use of the IP$IPP_CONFIG logical, or the queue-specific logical, IP$IPP_*queuename*_CONFIG if IP$IPP_IGNORE_DESCRIPTION is defined. The value of the equivalence string for IP$IPP_IGNORE_DESCRIPTION is not important. The existence or non-existence of the logical is all that is checked. This logical provides defaults that may be overridden by the queue-specific configuration logical, IP $IPP_*queuename*_IGNORE_DESCRIPTION, for a given queue.

IP$IPP_JOB_RETRY_DELAY

Specifies, as an OpenVMS delta time specification, the length of time to hold a job when it is re-queued due to a temporary problem. The default value is "0 00:10:00.00" (10 minutes).

IP$IPP_MAX_LOG_BYTES

Specifies how many bytes of data will be logged by the send and receive routines when running with logging level set to DETAILED_TRACE. The value is an integer. A negative value sets the limit to infinite (all data will be logged). A value of zero turns off inclusion of data to the log file. A positive value sets the actual number of bytes logged, and any additional data is ignored. The default action is to log all data.

IP$IPP_MAX_STREAMS

Specifies the number of streams (queues) that each IPP symbiont process can handle. This is an integer from 1 to 16. The default is 16.

IP$IPP_LOG_LEVEL

Specifies one of the Logging Levels values listed in Table 5.3, and is used to determine how serious a message must be before it is written to the log file. Only those messages marked as this level, or as a more serious level, are logged. The default is JOB_TRACE.

IP$IPP_LOGFILE

Specifies the name of the log file. All queues for a given symbiont process will share this file unless there are individual queue overrides. The default is to create the log file in the default spool directory, with the name IPP_SYMBIONT_*pid*.LOG.

`IP$IPP_OPCOM_LEVEL`

Specifies one of the Logging Levels values listed in Table 5.3, and is used to determine how serious a message must be before it is sent to OPCOM. Only those messages marked as this level, or as a more serious level, are sent. The default is INFO.

`IP$IPP_OPCOM_TERMINAL`

Specifies the OPCOM operator "terminal" to send OPCOM messages to. Permissible values are listed later in this section. The default is the "PRINT" operator.

5.7.2.2. Queue-specific Settings

These items are specified as qualifiers in the queue's **/DESCRIPTION** string, and/or in the `IP$IPP_`*`queuename`*`_CONFIG` logical equivalence string the two are concatenated before being processed. These qualifiers are not case sensitive. The underscores in the qualifier names are optional. Each may be abbreviated as long as the result is not ambiguous. The two sections below contain the complete list of qualifiers.

Queue-specific Required Qualifier

/PRINTER_URI

A valid URI, or list of URIs, for the printer or printers to be sent to from this queue. Wildcards are allowed ("*" to match one or more characters, "?" for a single character). The individual URIs in the list are separated from each other with the vertical bar ("|") character. The first URI in the list that does not include any wildcards is the default printer for the queue. If there are no default printer URIs and you have not specified a particular printer URI with the PRINT command, the job is aborted. Any printer URI specified with the PRINT command must match at least one of the URIs listed for the queue or the job will be aborted.

Queue-specific Optional Qualifiers

/COMMENT=*quoted string*

Allows inclusion of a quoted string of text that the symbiont will ignore, other than to write to the log file and/or OPCOM if the logging level is set to SYMBIONT or a more detailed setting.

/COPIES_DEFAULT=*number*

Specifies the number of copies of each document to print unless specified otherwise on the **PRINT** command. The default value is 1.

/DEBUG

Causes the symbiont to retain all spool files and to force DETAILED_TRACE logging to the log file, regardless of what other settings might be specified. Note that **/DEBUG** forces the setting for MAX_LOG_BYTES to a minimum of 512 bytes. You can set it higher, but any setting lower than 512 bytes will be ignored when **/DEBUG** is used.

/DEFAULT_DOCUMENT_FORMAT=*formatspec*

or

/DOCUMENT_FORMAT_DEFAULT=*formatspec*

Specifies the default document format for the queue. This value will be a MIME media type that is supported for the printer or printers served by this queue. It could also be the string "***printer_default***", which will result in whatever the target printer defines as its default when no document format is specified on the PRINT command.

/EXPECT_LINK_CLOSURES

Specifies that the printer is not fully HTTP 1.1 compliant because it does not support persistent connections, and does not send a "Connection: Close" header line in its last response. Therefore, the symbiont should assume that such a line was sent in every response, using a new link for each request, closing the old one, and not treating it as an error if the other end closes the link after sending a response.

/FINISHINGS_DEFAULT=*keyword*

Specifies finishing operations to be performed on the printed documents. May or may not be supported by a given IPP server. Any or all of the four available finishings may be specified. Case is ignored. Keywords are:

- NONE
- STAPLE
- PUNCH
- COVER
- BIND

/[NO]FLAG_DEFAULT

Specifies whether a “job-sheets” attribute will be specified for jobs by default. If **/FLAG_DEFAULT** is used, job-sheets will be requested as “standard”. If **/NOFLAG_DEFAULT** is used, job-sheets will be requested as “none”.

/INCLUDE=*filename*

Specifies a sequential access text file containing additional qualifiers from this list. These qualifiers are read and processed at the point where the **/INCLUDE** qualifier is encountered, and share the precedence of that point.

/JOB_PRIORITY_DEFAULT=*integer*

Specifies the priority of the print job. 1 is the lowest, 100 is the highest.

/JOB_RETRY_DELAY=*deltatime*

Specifies, as an OpenVMS delta time specification, the length of time to hold a job when it is re-queued due to a temporary problem. The default value is "0 00:10:00.00" (10 minutes).

/LOG_FILE=*filename*

Specifies the name of the queue log file to write messages to for this queue. The default is in VSI TCP/IP's default spool directory, unless overridden by a global setting. The default filename is IPP_SYMBIONT_*Process_PID*.LOG.

/LOG_LEVEL=*logging_level*

Specifies one of the Logging Levels values listed in the Table 5.3, and is used to determine the severity of a message before it is written to the queue log file. Only those messages marked as this level, or a more serious one, are logged. The default is JOB_TRACE unless overridden by a global `IP $IPP_LOG_LEVEL` logical.

/MAX_LOG_BYTES=*number*

Specifies how many bytes of data will be logged by the send and receive routines when running with logging level set to DETAILED_TRACE. The value is an integer. A negative value sets the limit to infinite (all data will be logged). A value of zero turns off inclusion of data to the log file. A positive value sets the actual number of bytes logged, and any additional data is ignored. The default action is to log all data.

/MEDIA_DEFAULT=*name*

This attribute identifies the medium that the printer uses for all pages of the Job.

The values for "media" include medium-names, medium-sizes, input-trays and electronic forms. See your printer documentation for details concerning what values are supported for your printer.

Standard keyword values are taken from ISO DPA and the Printer MIB and are listed in Section 14 of RFC 2566. Some servers may support definition of locally created names as well. Table 5.1 contains examples of standard names (included, but not limited).

Table 5.1. Standard Media Names

Name	Description
default	The default medium for the output device
iso-a4-white	Specifies the ISO A4 white medium
iso-a4-colored	Specifies the ISO A4 colored medium
iso-a4-transparent	Specifies the ISO A4 transparent medium
na-letter-white	Specifies the North American letter white medium
na-letter-colored	Specifies the North American letter colored medium
na-letter-transparent	Specifies the North American letter transparent medium
na-legal-white	Specifies the North American legal white medium
na-legal-colored	Specifies the North American legal colored medium
na-9x12-envelope	Specifies the North American 9x12 envelope medium
monarch-envelope	Specifies the Monarch envelope
na-number-10-envelope	Specifies the North American number 10 business envelope medium
na-7x9-envelope	Specifies the North American 7x9 inch envelope
na-9x11-envelope	Specifies the North American 9x11 inch envelope
na-10x14-envelope	Specifies the North American 10x14 inch envelope
na-number-9-envelope	Specifies the North American number 9 business envelope
na-6x9-envelope	Specifies the North American 6x9 inch envelope
na-10x15-envelope	Specifies the North American 10x15 inch envelope
executive-white	Specifies the white executive medium

Name	Description
folio-white	Specifies the folio white medium
invoice-white	Specifies the white invoice medium
ledger-white	Specifies the white ledger medium
quarto-white	Specified the white quarto medium
iso-a0-white	Specifies the ISO A0 white medium
iso-a1-white	Specifies the ISO A1 white medium
a	Specifies the engineering A size medium
b	Specifies the engineering B size medium
c	Specifies the engineering C size medium
d	Specifies the engineering D size medium
e	Specifies the engineering E size medium

Table 5.2 lists standard values defined for input trays.

Table 5.2. Input Tray Names

Name	Description
top	The top input tray in the printer.
middle	The middle input tray in the printer.
bottom	The bottom input tray in the printer.
envelope	The envelope input tray in the printer.
manual	The manual feed input tray in the printer.
large-capacity	The large capacity input tray in the printer.
main	The main input tray
side	The side input tray

/MULTIPLE_DOCUMENT_HANDLING_DEFAULT=*keyword*

This qualifier is relevant only for jobs consisting of two or more documents, and when the IPP server supports jobs with multiple documents. The qualifier controls finishing operations and the placement of one or more pages onto media sheets. When printing multiple copies, it also controls the order in which the copies that result are produced. Standard keyword values are

single-document

If a Job has multiple documents, say, the documents are called A and B, then the result printing the data (A and then B) will be treated as a single sequence of media sheets for finishing operations; that is, finishing would be performed on the concatenation of the two documents. The Printer will not force the data in each document to start on a new page.

If more than one copy is requested, the ordering of the pages resulting from printing will be A, B, A, B, ..., and the Printer will force each copy (A, B) to start on a new media sheet.

separate-documents-uncollated-copies

If a Job has multiple documents, say, the documents are called A and B, then the result of printing each document will be treated as a single sequence of media sheets for finishing operations; that is,

the documents A and B would each be finished separately. The Printer will force each copy of the data in a single document to start on a new sheet.

If more than one copy is made, the ordering of the pages will be A, A, ..., B, B ... .

separate-documents-collated-copies

If a Job has multiple documents, say, A and B, then the result will be that each document will be treated as a single sequence of media sheets for finishing operations; that is, A and B would each be finished separately. The Printer will force each copy of the result of processing the data in a single document to start on a new sheet.

If more than one copy is made, the ordering of the documents will be A, B, A, B,... .

single-document-new-sheet

Same as 'single-document', except that the Printer will ensure that the first page of each document in the job is placed on a new media sheet.

The 'single-document' value is the same as 'separate-documents-collated-copies' with respect to ordering of print-stream pages, but not media sheet generation, since 'single-document' will put the first page of the next document on the back side of a sheet if an odd number of pages have been produced so far for the job, while 'separate-documents-collated-copies' always forces the next document or document copy on to a new sheet. In addition, if the "finishings" attribute specifies 'staple', then with 'single-document', documents A and B are stapled together as a single document with no regard to new sheets, with 'single-document-new-sheet', documents A and B are stapled together as a single document, but document B starts on a new sheet, but with 'separate-documents-uncollated-copies' and 'separate-documents-collated-copies', documents A and B are stapled separately.

Note

None of these values provides means to produce uncollated sheets within a document, i.e., where multiple copies of sheet n are produced before sheet n+1 of the same document.

/NUMBER_UP_DEFAULT=*number*

Specifies the number of page images to be placed on each side of each sheet of paper. The number must be an integer that is acceptable to the IPP server. If the number specified is not a value supported by the server, the job aborts.

/OPCOM_LEVEL=*logging_level*

Specifies one of the Logging Levels value listed in Table 5.3, and is used to determine the severity of a message before it is sent to OPCOM. Only those messages marked as this level, or at a more serious level, are sent. The default is INFO unless overridden by a global `IP$IPP_OPCOM_LEVEL` logical.

/OPCOM_TERMINAL=*opcom_term*

Specifies which OPCOM operator "terminal" to send OPCOM messages to. Available values are listed in OPCOM Terminal Names. The default is the "PRINT" operator. See the OpenVMS documentation for the **REPLY/ENABLE** command for more information on OPCOM terminals.

/ORIENTATION_DEFAULT=*keyword*

Specifies the default page orientation. Case is ignored. Supported values are:

- PORTRAIT
- REVERSE_PORTRAIT
- LANDSCAPE
- REVERSE_LANDSCAPE

/PAGE_RANGE_DEFAULT=*range[,range]...*

Specifies the page numbers to print. *range* is either a single integer page number, or a pair of page numbers, separated by a hyphen. Multiple range specifications are separated by commas. For example:

```
$ PRINT/QUEUE=IPP_QUEUE/PARAM=(PAGE_RANGES=1,3-6,9,10 ,12-14) TEST.TXT
```

The example specifies the pages: 1, 3, 4, 5, 6, 9, 10, 12, 13, and 14. Note that embedded spaces are allowed, and ignored.

/QUALITY_DEFAULT=*keyword*

Specifies the quality of the printed material. Case is ignored. The keywords are:

- DRAFT
- NORMAL
- HIGH

/SIDES_DEFAULT=*keyword*

Specifies how the printing is to be placed on the paper.

- ONE-SIDED: prints each consecutive page upon one side of consecutive media sheets.
- TWO-SIDED-LONG-EDGE: prints each consecutive pair of pages upon the front and back sides of consecutive media sheets, with the orientation of each pair of pages on the long edge. This positioning is called “duplex” or “head-to-head” also.
- TWO-SIDED-SHORT-EDGE: prints each consecutive pair of pages upon front and back sides of consecutive media sheets, with the orientation of each pair of print-stream pages on the short edge. This positioning is called “tumble” or “head-to-toe” also.

/SPOOL_DIRECTORY=*dirspec*

Specifies the directory to use for storing temporary files used while processing print jobs for the queue. The default is VSI TCP/IP default spool directory.

5.7.2.3. Order of Processing

The various logicals and qualifiers described in the previous section sometimes define the same configuration item. The operation has been defined, but the precedence has not. The order, from lowest precedence to highest, is:

1. Built-in hard coded default values.

2. Global logicals, as described in the first section.

3. Queue-specific qualifiers found in the **/DESCRIPTION** string of the queue.

4. Queue-specific qualifiers found in the queue-specific CONFIG logical.

The queue-specific qualifiers are parsed second, allowing for an override of the global settings on a queue-by-queue basis when that behavior is desired.

5.8. Print Command Options

Print command options are specified using the OpenVMS standard /PARAMETERS qualifier. For example,

```
$ PRINT /QUEUE=IPP_PRINTER_1
/PARAMETER=(COPIES=3, ORIENTATION=LANDSCAPE) FILE.TXT
```

These options are not case sensitive. The underscores in the option names are optional. Each may be abbreviated as long as the result is not ambiguous. The available print command options are:

PRINTER=*printer_uri*

Specifies the target printer when the queue default is not defined, or when there is no queue default. The printer URI specified must match at least one of the defined printer_uri's for the print queue.

Wildcards cannot be used in the printer URI.

COPIES=*number*

Specifies the number of copies of each document to print. The default value is 1.

SIDES=*keyword*

Specifies how the printing is to be placed on the paper. The *keyword* must be one of the following:

- ONE-SIDED or 1sided: prints each consecutive page upon one side of consecutive media sheets.
- TWO-SIDED-LONG-EDGE or two-long-edge or 2long_side: prints each consecutive pair of pages upon the front and back sides of consecutive media sheets, with the orientation of each pair of pages on the long edge. This positioning is called “duplex” or “head-to-head” also.
- TWO-SIDED-SHORT-EDGE or two-short-edge or 2short_side: prints each consecutive pair of pages upon front and back sides of consecutive media sheets, with the orientation of each pair of print-stream pages on the short edge. This positioning is called “tumble” or “head-to-toe” also.

ORIENTATION=*keyword*

Specifies the page orientation. The *keyword* must be one of:

- PORTRAIT
- REVERSE_PORTRAIT
- LANDSCAPE
- REVERSE_LANDSCAPE

These can be abbreviated to any non-ambiguous prefix. Case is ignored.

[NO]FLAG

Requests, or suppresses, the printing of an IPP flag page for the job. The printer may, or may not, respond to this request. The exact format of this flag page is up to the IPP Server (printer) implementation.

NUMBER_UP=*number*

Specifies the number of page images to be placed on each side of each sheet of paper. The number must be an integer that is acceptable to the IPP server. If the number specified is not a value supported by the server, the job aborts.

DOCUMENT_FORMAT=*MIME-media-type*

or

DOCUMENT_FORMAT=*printer_default***

Specifies the document format of the files in the job, or specifies use of the printer's built-in default. The default for this qualifier is the default for the queue. Also, if the queue configuration does not specify a default document format, the hard coded default is "text/plain".

JOB_PRIORITY=*integer*

Specifies the priority of the print job at the IPP server (not to be confused with the OpenVMS queue priority). 1 is the lowest, 100 is the highest.

FINISHINGS="*keyword[,keyword]...*"

Specifies finishing operations to be performed on the printed documents. May or may not be supported by a given IPP server. Any or all of the four available finishings may be specified. Case is ignored.

- BIND
- COVER
- PUNCH
- STAPLE

MULTIPLE_DOCUMENT_HANDLING=*keyword*

Specifies how you want the printer to print your job. The *keyword* is one of the following:

- Single_Document or 1Document
- Separate_Documents_Uncollated_Copies or UncollatedSeparate
- Separate_Documents_Collated_Copies or CollatedSeparate
- Single_Document_New_Sheet or NewSheet

Case is ignored. See /MULTIPLE_DOCUMENT_HANDLING_DEFAULT=keyword for information on single document, separate-documents-uncollated-copies, separate-documents-collated-copies, and single-document-new-sheet handling.

PAGE_RANGES="*range[,range]...*"

Specifies the page numbers to print. *range* is either a single integer page number, or a pair of page numbers, separated by a hyphen. Multiple range specifications are separated by commas and enclosed in double quotes.

For example:

```
$ PRINT/QUEUE=IPP_QUEUE/PARAM=(PAGE_RANGES="1,3-6, 9, 10, 12-14") FILE.TXT
```

Note that embedded spaces are allowed, and ignored. The example specifies the pages: 1, 3, 4, 5, 6, 9, 10, 12, 13, and 14.

MEDIA=*name*

This attribute identifies the medium that the Printer uses for all pages of the Job.

The values for "media" include medium-names, medium-sizes, input-trays and electronic forms. See your printer documentation for details concerning what values are supported for your printer.

Standard keyword values are taken from ISO DPA and the Printer MIB and are listed in section 14 of RFC 2566. Some servers may support definition of locally created names as well.

See Table 5.1 and Table 5.2 for the standard media names.

QUALITY=*keyword*

Specifies the quality of the printed material. Case is ignored. The keyword choices are:

- DRAFT
- HIGH
- NORMAL

5.9. Allowable Values

Several of the configuration and job submission settings require values for OPCOM terminal names or logging severity levels. This section defines the allowable values for these options.

5.9.1. OPCOM Terminal Names

CARDS

CENTRAL

CLUSTER

DEVICES

DISKS

LICENSE

NETWORK

PRINTER (default)

SECURITY

TAPES

OPER1

OPER2

OPER3OPER4

OPER5

OPER6

OPER7

OPER8

OPER9

OPER10OPER11

OPER12

NONE (do NOT send to OPCOM except OVERRIDE events)

5.9.2. Logging Levels

All values may be abbreviated to any non-ambiguous prefix. These values are not case sensitive.

Table 5.3. Logging Levels

DETAILED_TRACE	All events
FILE	Events related to processing of individual files
JOB	Events related to processing of individual jobs
SYMBIONT	Events related to the state of the symbiont
INFO	Events providing information about non-error conditions
WARNING	Events warning of potential problems that do not halt processing
ERROR	Events reporting an error that prevented processing of a job
FATAL	Events reporting an error that stopped a queue
ABORT	Events reporting an error that caused the symbiont to exit

There are a few messages that are marked to be reported regardless of the setting of the various OPCOM and log file severity levels. These are kept to a minimum, but are considered to be important enough to override the logging level settings. These cannot be suppressed.

5.10. Using Logicals to Define Queue Configurations

This section provides examples of using logicals to define queue configuration prior to queue initialization. This method can be used both as an alternative to and in addition to the /

DESCRIPTION string shown in the previous examples. See Section 5.7.2 for a complete description of all available qualifiers.

5.10.1. Setting Up IPP Symbiont Queues

Creating an IPP symbiont queue is done using the OpenVMS **INITIALIZE/QUEUE** command. All standard qualifiers are allowed, but the **/DESCRIPTION** qualifier has special use with the IPP symbiont. See Section 5.7.2.

5.10.1.1. Setting up IPP Symbiont Queues Using Queue-Specific Logicals

Set up an IPP symbiont queue named ENG_PRINTER to obtain its configuration information from a queue specific configuration file and to print a flag page with each job.

```
$ DEFINE/SYSTEM IP$IPP_ENG_PRINTER_CONFIG -
_$ "/INCLUDE=SYS$SYSTEM:ENG_PRINTER.SETUP/FLAG_DEFAULT"
$ INITIALIZE/QUEUE/PROCESSOR=IP$IPP_SYMB ENG_PRINTER
```

The file SYS$SYSTEM:ENG_PRINTER.SETUP contains:

```
/printer="ipp://engprinter.mynet.com:631/ipp"
```

5.10.1.2. Setting Up an IPP Symbiont Queue to Print Only to a Specific Printer

Set up the IPP symbiont queues named IPP_PRINT_QUEUE and IPP_PRINT_2 to print only to the iprinter.mynet.com printer on port 631. Additionally, IPP_PRINT_2 will always print two copies of each submitted file if copies are supported by the printer.

```
$ DEFINE/SYSTEM IP$IPP_CONFIG -
_$ "/PRINTER_URI=""ipp://iprinter.mynet.com:631/ipp"""
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB IPP_PRINT_QUEUE
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB -
_$ /DESCRIPTION="/copies_default=2" IPP_PRINT_2
```

5.10.1.3. Setting Up to Print to Multiple Printers Using Wildcards

Set up an IPP symbiont queue to print to any IPP printer in the *mynet.com* domain, with the default printer being *iprinter.mynet.com*:

```
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB /DESCRIPTION="/printer=
""http://iprinter.mynet.com:631/ipp|*.mynet.com""" IPP_PRINT_QUEUE
```

5.10.1.4. Setting Up Two Queues Using a Disk File for Queue Settings

Set up two IPP symbiont queues to print to any printer in the *mynet.com* domain, with the default printer being *iprinter.mynet.com* for one queue, and *oprinter.mynet.com* for the other. Log all possible messages to the log file, but send only messages more severe than FILE_TRACE to OPCOM. Use a 5 minute retry delay, and make the document format default the same as the printer's default. Use a disk file for the configuration information common to both queues:

```
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB -
```

```
_$ /DESCRIPTION="/printer=
""http://iprinter.mynet.com:631/ipp|*.mynet.com""
/include=SYS$SYSTEM:IPP_QUEUE.SETUP" IPRINTER_QUEUE
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB -
_$ /DESCRIPTION="/printer=
""http://oprinter.mynet.com:631/ipp|*.mynet.com""
/include=SYS$SYSTEM:IPP_QUEUE.SETUP" OPRINTER_QUEUE
```

The file SYS$SYSTEM:IPP_QUEUE.SETUP contains:

```
/log_level=DETAILED_TRACE
/opcom_level=FILE_TRACE
/job_retry_delay="0 00:05:00.00"
/default_document_format=***printer_default***
```

5.10.1.5. Setting Up Two Queues with no Configuration Values in the INITIALIZE Command

Do the same as the prior example, but put as much of the configurations in disk files as possible to allow changes to queue characteristics without having to re-initialize the queues:

```
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB -
_$ /DESCRIPTION="/INCLUDE=SYS$SYSTEM:IPP_IPRINTER.SETUP" IPRINTER_QUEUE
$ INITIALIZE/QUEUE /PROCESSOR=IP$IPP_SYMB -
_$ /DESCRIPTION="/INCLUDE=SYS$SYSTEM:IPP_OPRINTER.SETUP" OPRINTER_QUEUE
```

The file SYS$SYSTEM:IPP_IPRINTER.SETUP contains:

```
/printer="http://iprinter.mynet.com:631/ipp|*.mynet.com"
/include=SYS$SYSTEM:IPP_QUEUE.SETUP
```

The file SYS$SYSTEM:IPP_OPRINTER.SETUP contains:

```
/printer="http://oprinter.mynet.com:631/ipp|*.mynet.com"
/include=SYS$SYSTEM:IPP_QUEUE.SETUP
```

The file SYS$SYSTEM:IPP_QUEUE.SETUP contains:

```
/log_level=DETAILED_TRACE
/opcom_level=FILE_TRACE
/job_retry_delay="0 00:05:00.00"
/default_document_format=***printer_default***
```

5.10.2. Submitting Jobs to IPP Symbiont Print Queues

This section describes how to submit jobs to the IPP symbiont print queues.

5.10.2.1. Printing a Single Text File to an IPP Queue

Print the file FOO.TXT to the IPRINTER (default destination printer) set up in the prior examples:

```
$ PRINT/QUEUE=IPRINTER_QUEUE foo.txt
```

5.10.2.2. Specifying the Destination Printer on the Print Command

Print a single text file to a non-default printer on a queue with a wild carded printer URL:

```
$ PRINT /QUEUE=iprinter_queue -
_$ /PARAM=(printer="ipp://another.mynet.com/ipp/port1") foo.txt
```

Note

The above will fail unless the queue specifies *another.mynet.com* as a legal URL, either explicitly or by using wildcards.

5.10.2.3. Using Other Print Qualifiers

Print a text file to a default printer on a queue but specify the document format and additional copies:

```
$ PRINT /QUEUE=iprinter_queue -
_$ /PARAM=(document="plain/text",copies=3)foo.txt
```

5.11. VSI TCP/IP IPP SHOW Command

The VSI TCP/IP IPP **SHOW** utility allows a user to learn the capabilities supported by an IPP server. The command syntax is:

```
$ IP IPP SHOW server_URI /qualifiers...
```

Refer to the *VSI TCP/IP Administrator's Reference* for details.

Appendix A. Server Configuration Parameters

A.1. SERVER-CONFIG Service Parameters

The following table describes the service parameters you can set with the SERVER-CONFIG Utility.

Table A.1. SERVER-CONFIG Service Parameters

Parameter	Description
ACCEPT-HOSTS	The list of hosts allowed access to this server.
ACCEPT-NETS	The list of networks or subnetworks that are allowed to access this server.
BACKLOG	The number of server connections to queue up before refusing to accept additional connections when MAX-SERVERS is reached.
CONFFILE	This file is used instead of the default `IP$:NAMED.CONF` file.
CONNECTED[1]	The name of the internal IP$SERVER routine to call when a connection request is received. This varies by protocol and is normally not changed.
DEBUG	Sets the debug level of the Domain Nameserver (the default is no debugging). The larger the number, the more verbose the output. A value of 0 turns off debugging.
DEBUGFILE	This file is used instead of the default `IP$:NAMED.RUN` file.
DISABLED-NODES	The list of OpenVMScluster nodes that cannot run this server.
ENABLED-NODES	The list of OpenVMScluster nodes that can run this server.
FLAGSa	Various flags that control the operation of the service.
INITa	The name of the internal IP$SERVER routine for initializing a service. This varies by protocol and is normally not changed.
LISTENa	The name of the internal IP$SERVER routine for listening for connections to the service. This varies by protocol and is normally not changed.
LOG-ACCEPTS	Specifies whether to log successful connections to the service.
LOG-FILE	Destination of log messages. May be a OpenVMS filename or OPCOM to direct messages to the OpenVMS OPCOM process.
LOG-REJECTS	Specifies whether to log rejected connections to the service. A connection is rejected because of the combination of the REJECT-HOSTS, REJECT-NETS, and REJECT-BY-DEFAULT parameters.
MAX-SERVERS	The maximum number of service processes to allow at any one time. If this limit is reached, additional connections – up to the number specified by the BACKLOG parameter -- are accepted but not processed until one of the previous connections completes.
MAXIMUM-TTL	Changes the maximum time-to-live (TTL) that resource records are cached from the default of 604800 seconds (1 week) to the specified value.

Parameter	Description
MINIMUM-TTL	Changes the minimum time-to-live (TTL) that resource records are cached from the default of zero (0) seconds to the specified value. **Note** It is recommended you use this command only if there is a specific need. This could cause problems in that you may be caching resource records for longer than the authoritative administrator intended.
PARAMETERS	Specifies service-dependent parameters passed to the initialization routine of built-in services. Normally not used for user-written services.
PRESERVE-CASE	Tells NAMED to keep the case of the response from the remote nameserver after doing a query.
PRIORITY	The OpenVMS process priority to assign to created processes.
PROGRAM	The OpenVMS filename of the image to run or merge.
QUERY-LOG	Toggles query logging ON and OFF. Query logging shows an informational message every time a query is received by the server. Query logging can be directed to OPCOM or a file in the `IP $:NAMED.CONF` file using the logging category *queries*.
REJECT-BY-DEFAULT	Whether to reject a connection from a host that does not match any of the ACCEPT-HOSTS, ACCEPT-NETS, REJECT-HOSTS, and REJECT-NETS lists.
REJECT-HOSTS	The list of hosts not allowed access to this server.
REJECT-MESSAGE	A text string to send down the network connection when a service is rejected.
REJECT-NETS	The list of networks or subnetworks not allowed access to this server.
REWRITE-TTL	Sets the time-to-live (TTL) that load balanced resource records are cached from the default of 300 seconds (5 minutes) to the specified value.
SERVICE	The name of the internal IP$SERVER routine to call to perform the service. This is normally **Run_Program** for user-written services.
SERVICE-NAME	The name of the service.
SOCKET-FAMILY	The address family of the service; for example, AF_INET.
SOCKET-OPTIONS	Socket options to be set via **setsockopt()**. See the *VSI TCP/IP Programmer's Reference* for more information on socket options.
SOCKET-PORT	The port number on which to listen for connections.
SOCKET-TYPE	The type of socket; for example, **SOCK_STREAM (TCP)** or **SOCK_DGRAM (UDP)**.
WORKING-SET	The OpenVMS working set to assign to created processes.

[1]For servers supplied with VSI TCP/IP, do not change this parameter. When adding your own servers, you usually do not need to modify the default parameters.

A.2. Services Provided with VSI TCP/IP

Table A.2 shows the VSI TCP/IP servers that can be disabled or enabled using SERVER-CONFIG.

Table A.2. SERVER-CONFIG Services Provided with VSI TCP/IP

Protocol	Server	Service Provided
RPC	NFS	Network File System server
	RPCBOOTPARAM	RPC boot parameters for diskless hosts
	RPCLOCKMGR	RPC lock manager
	RPCMOUNT	RPC procedure for mounting file systems
	RPCPORTMAP	RPC portmapper (RPC naming service)
	RPCQUOTAD	Returns disk quota information
	RPCSTATUS	RPC status daemon
	RUSERS	Returns information about logged in users
	RWALL	Broadcasts messages to users
Special	CLUSTERALIAS	Special server for managing cluster-wide IP addresses
	RARP	Ethernet Reverse Address Resolution Protocol
	UCXQIO	VMS/ULTRIX Connection $QIO emulation special services
TCP	CHARGEN	Character generator
	DAYTIME	Returns time of day in ASCII
	DISCARD	Discard data received
	ECHO	Echo data received
	FINGER	Lists information about users on host
	FTP	File Transfer Protocol
	LPD	Remote printing service
	NETCONTROL	Remote network server control
	NETSTAT	Return network configuration and statistics
	NTALK	Interactive conversation with remote user (newer 4.3 BSD protocol)
	POP2	Post Office Protocol Version 2
	POP3	Post Office Protocol Version 3
	REXEC	Remote command execution (with password)
	RLOGIN	Network virtual terminal service (with BSD "R" services authentication)
	RSHELL	Remote command execution (with BSD "R" services authentication)
	SMTP	Simple Mail Transfer Protocol
	SSH	Secure Shell server and client
	SYSTAT	Remote system status (remote SHOW SYSTEM)

Protocol	Server	Service Provided
	TELNET	Network virtual terminal service
	TIME	Returns time of day in binary
UDP	BOOTP	Remote booting protocol
	DHCP	Dynamic remote booting protocol
	DOMAINNAME	Internet Domain Name System (DNS) name service (BIND)
	GATED	Gateway routing service (EGP, RIP, and HELLO)
	NTP	Network Time Protocol-For time synchronization
	SNMP	Simple Network Management Protocol agent
	SMUX	SNMP Multiplexing Protocol
	TALK	Interactive conversation with remote user (old 4.2 BSD protocol)
	TFTP	Trivial File Transfer Protocol
	UDPCHARGEN	Character generator
	UDPDAYTIME	Returns current time of day in ASCII
	UDPDISCARD	Discard data received
	UDPECHO	Echo data received
	UDPTIME	Returns time of day in binary

A.3. Default Server Values

Table A.3shows the default value for VSI TCP/IP service parameters set using SERVER-CONFIG. Note that not all service parameters have defaults; hence, only those with default values are listed in this table.

Note

Many default values in Table A.3 will be shown ss “Current process JPI$_xxxxx”. These will generally, but not always, correspond to the OpenVMS SYSGEN parameter “PQL_Dxxxx”. For example, JPI$_ASTLM will generally correspond to PQL_DASTLM.

Table A.3. SERVER-CONFIG Default Service Values

Server	Parameter	Default value
- All Services -	BACKLOG	10
	LOG-ACCEPTS	no
	LOG-FILE	none
	LOG-REJECTS	no
	MAX-SERVERS	1000
	PRIORITY	Current process JPI$_PRIB

Server	Parameter	Default value
	REJECT-MESSAGE	none
	USERNAME	no
	WORKING-SET-EXTENT	Current process JPI$_WSEXTENT
	WORKING-SET-QUOTA	Current process JPI$_WSQUOTA
ACCOUNTING	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
BOOTP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
CHARGEN	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM

Server	Parameter	Default value
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
CLUSTERALIAS	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
DAYTIME	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM

Server	Parameter	Default value
	PQL-TQELM	Current process JPI$_TQELM
DHCLIENT	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
DHCP	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
DISCARD	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM

Server	Parameter	Default value
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
DOMAINNAME	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
ECHO	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
FINGER	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM

Server	Parameter	Default value
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
FONTSERVER	PQL-ASTLM	1000
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	500
	PQL-ENQLM	500
	PQL-FILLM	500
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	500
FTP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	30
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA

Server	Parameter	Default value
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
GATED	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
KTELNET	PQL-ASTLM	250
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
LLMR	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	Current process JPI$_DIOLM

Server	Parameter	Default value
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
LPD	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
NAMED	PQL-ASTLM	250
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
NETCONTROL	PQL-ASTLM	Current process JPI$_ASTLM

Server	Parameter	Default value
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
NETSTAT	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
NFS	PQL-ASTLM	1000
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	DIOLM
	PQL-ENQLM	500
	PQL-FILLM	500
	PQL-JTQUOTA	4096

Server	Parameter	Default value
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	500
NTALK	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
NTP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	500
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	100
POP2	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM

Server	Parameter	Default value
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
POP3	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RACOON	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0

Server	Parameter	Default value
	PQL-TQELM	100
RARP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
REXEC	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RLOGIN	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM

Server	Parameter	Default value
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCBOOTPARAM	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCLOCKMGR	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCMOUNT	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM

Server	Parameter	Default value
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCPORTMAP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCQUOTAD	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA

Server	Parameter	Default value
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RPCSTATUS	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RSHELL	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RUSERS	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM

Server	Parameter	Default value
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
RWALL	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
SMTP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
SMUX	PQL-ASTLM	Current process JPI$_ASTLM

Server	Parameter	Default value
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
SNMP	PQL-ASTLM	250
	PQL-BIOLM	150
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
SSH	PQL-ASTLM	250
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	0
	PQL-DIOLM	150
	PQL-ENQLM	500
	PQL-FILLM	100
	PQL-JTQUOTA	4096

Server	Parameter	Default value
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	0
	PQL-TQELM	100
SYSTAT	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
TALK	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
TELNET	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM

Server	Parameter	Default value
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
TFTP	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
TIME	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM

Server	Parameter	Default value
	PQL-TQELM	Current process JPI$_TQELM
UCXQIO	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
UDPCHARGEN	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
UDPDAYTIME	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM

Server	Parameter	Default value
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
UDPDISCARD	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
UDPECHO	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
UDPTIME	PQL-ASTLM	Current process JPI$_ASTLM
	PQL-BIOLM	Current process JPI$_BIOLM

Server	Parameter	Default value
	PQL-BYTLM	Current process JPI$_BYTLM
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	Current process JPI$_DIOLM
	PQL-ENQLM	Current process JPI$_ENQLM
	PQL-FILLM	Current process JPI$_FILLM
	PQL-JTQUOTA	Current process JPI$_JTQUOTA
	PQL-PGFLQUOTA	Current process JPI$_PGFLQUOTA
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	Current process JPI$_TQELM
XDM	PQL-ASTLM	1000
	PQL-BIOLM	500
	PQL-BYTLM	100000
	PQL-CPULM	Current process JPI$_CPULM
	PQL-DIOLM	500
	PQL-ENQLM	500
	PQL-FILLM	500
	PQL-JTQUOTA	4096
	PQL-PGFLQUOTA	65536
	PQL-PRCLM	Current process JPI$_PRCLM
	PQL-TQELM	500

Appendix B. Statements for Configuring Network Routing

This appendix describes statements that you can use to configure network routing.

B.1. Routing Methods Overview

VSI TCP/IP provides two routing methods:

- **The IP routing table (also known as static routing)**: Specifies the IP addresses of destinations and gateways for your communications. Use static routing when you have a very simple network, or a moderately complex network with a configuration that does not change often. Static routing is simple to configure and use.
- **The GATED dynamic gateway routing daemon**: Handles multiple routing protocols and replaces ROUTED and EGPUP. GATED is used when node or gateway reconfigurations occur frequently, where the network geometry changes as networks come and go, or when dynamic routing protocols are being used for other reasons. GATED is more difficult to configure and use, but is much more flexible than static routing and provides a much greater selection of features.

B.1.1. Configuring Static IP Routes

The VSI TCP/IP **SET /ROUTE** command specifies static IP routing, including the default route. This command is normally invoked automatically by the network startup command file generated by the VSI TCP/IP Network Configuration Utility (NET-CONFIG).

Note

If the GATED gateway routing service is enabled, it takes full control of your routing tables and removes any statically-defined routes that are not also specified in the GATED configuration.

B.1.1.1. Adding Static Routes

You can configure a static default route using NET-CONFIG.

You can add additional static routes by creating the command file `IP$:LOCAL_ROUTES.COM`, which is executed as part of the network startup.

An example of the command for configuring a static route is:

```
$ IP SET/ROUTE/ADD=(DEST=DEFAULT,GATE=192.0.0.1)
```

Note

The contents of `IP$:LOCAL_ROUTES.COM` will not be populated if DHCP client is being used. Please contact VSI support for a workaround.

B.1.1.2. Changing the Default Route

To change the default route, first delete the current default route, then add the new one:

```
$ IP SET/ROUTE/DELETE=(DEST=DEFAULT,GATE=IP address)
$ IP SET/ROUTE/ADD=(DEST=DEFAULT,GATE=IP address)
```

B.2. Using GateD

The Gateway Routing Daemon (GateD) manages multiple routing protocols, including the Routing Information Protocol (RIP), Local Network Protocol (HELLO), Router Discovery Protocol, Open Shortest Path First (OSPF) protocol, Exterior Gateway Protocol (EGP), and Border Gateway Protocol (BGP).

Using GateD, the network administrator can control the flow of routing information through a configuration language. Once you start GateD, it makes routing decisions based on the data gathered by the routing protocols. If routing using GateD, use GateD exclusively.

GateD allows you to control importing and exporting routing information by:

- Individual protocol
- Source and destination Autonomous System (AS)
- Source and destination interface
- Previous hop router
- Specific destination address

You can assign preference levels for different combinations of imported routing information by using a flexible masking capability. In VSI TCP/IP, the name of the GateD process is GateD.

B.2.1. GateD Configuration File

VSI TCP/IP stores GateD configuration information in the `IP$:GATED.CONF` file. You must create this file before you can use GateD. For details on GateD configuration, see Section B.4.

B.2.2. GateD Route Selection

GateD determines the best route using preference values set for each protocol or peer. Each route has a single associated preference value, even though you can set preferences at many places in the `GATED.CONF` file. The last (or most specific) preference value is the one GateD uses. Some protocols have a secondary preference, sometimes called a "tie-breaker."

The factors GateD uses in determining best routes include:

- The route with the numerically smallest **preference** value is preferred.
- For two routes with equal preferences, the route with the numerically smallest **preference2** (the "tie-breaker") is preferred.
- A route learned from an interior gateway protocol is preferred over a route learned from an exterior gateway protocol. Least preferred is a route learned indirectly by an interior protocol from an exterior protocol.
- If Autonomous System (AS) path information is available, it helps determine the most preferred route:
    - A route with an AS path is preferred over one without an AS path.

    - If the AS paths and origins are identical, the route with the lower metric is preferred.
    - A route with an AS path origin of interior protocol is preferred over one with an origin of exterior protocol. Least preferred is an AS path with an unknown origin.
    - A route with a shorter AS path is preferred.
- If both routes are from the same protocol and AS, the one with the lower metric is preferred.
- The route with the lowest numeric next-hop address is used.

Preference values range from 0 to 255. Table B.1 summarizes the default preference values for routes learned in various ways.

Table B.1. Default Routing Preference Values Defined by GateD Statements

Default preference value	Is defined by ... statement
0	interface
10	ospf
20	gendefault (internally generated default)
30	redirect
40	kernel (routes learned using the socket route)
60	static
90	hello
100	rip
110	(point-to-point interfaces)
120	interfaces (routes to interfaces that are down)
130	aggregate/generate
150	ospf (AS external)
170	bgp
200	egp

B.3. Starting and Stopping GateD

After creating the `IP$:GATED.CONF` file, you need to stop and restart GateD. Follow these steps:

1. Log in as the system manager.
2. Stop the GateD process by entering: **IP GATED/STOP**
3. Restart the GateD process by entering: **@IP$:START_SERVER**

It is not necessary to stop and restart GateD to get it to read a new `IP$:GATED.CONF` file, just use **IP GATED /LOAD**.

B.4. Configuring GATED

Use the commands in Table B.2 to manage the GateD process. To use these commands, you need OPER or SYSPRV privilege.

Table B.2. GateD Commands

Command	Description
`IP GATED/CHECK (file)`	Checks a GateD configuration file for syntax errors
`IP GATED /DUMP`	Dumps the state of the GateD process to a file
`IP GATED /LOAD`	Loads a new GateD configuration file
`IP GATED /SET /TRACE`	Controls tracing in GateD
`IP GATED /SHOW /OSPF /ADVERTISE`	Shows OSPF link state advertisements
`IP GATED /SHOW /OSPF /AS`	Shows the AS external database entries
`IP GATED /SHOW /OSPF /DESTINATIONS`	Shows the list of destinations and their indices
`IP GATED /SHOW /OSPF /ERRORS`	Shows the OSPF error log
`IP GATED /SHOW /OSPF /HOPS`	Shows the set of next hops for the OSPF router queried
`IP GATED /SHOW /OSPF /INTERFACES`	Shows all configured interfaces for OSPF
`IP GATED /SHOW /OSPF /LOG`	Shows the cumulative OSPF log of input/output statistics
`IP GATED /SHOW /OSPF /NEIGHBORS`	Shows all OSPF routing neighbors
`IP GATED /SHOW /OSPF /ROUTING`	Shows the OSPF routing table
`IP GATED /SHOW /OSPF /STATE`	Shows the link state database (except AS Externals)
`IP GATED /SHOW /RIP`	Queries Routing Information Protocol (RIP) gateways
`IP GATED /SHOW /TRACE`	Shows tracing in GateD
`IP GATED /STOP`	Stops the GateD process
`IP GATED /TOGGLE_TRACING`	Toggles tracing in GateD
`IP GATED /UPDATE_INTERFACES`	Rescans the GateD network interfaces

B.5. GateD Configuration Statements

The GateD configuration file is `IP$:GATED.CONF`. This file must be present for the GateD process to run. The structure of the GateD configuration language is similar to C. The configuration file consists of statements terminated by a semicolon (;). Statements consist of tokens separated by a space. This structure simplifies identification of the associated parts of the configuration.

You can include comment lines either by beginning them with a pound sign (#) or delimiting them with slash asterisk (/*) and asterisk slash (*/). The configuration file consists of the following sections, which reflect the order in which the statements, if used, must appear:

Directives	`(%directory, %include)`

Statements	`traceoptions`
	`options`
	`interfaces`

Definitions	`autonomous-system`
	`routeid`
	`martians`

Protocols	`rip`
	`help redirect router-discovery server/client`
	`bgp`
	`ospf`

Static routes	`static`

Control	`import`
	`export`
	`aggregategenerate`

Directives

Directives — Directive statements include: **%directory** and **%include**. Directive statements provide special instructions to the parser. They do not relate to the protocol configuration and can occur anywhere in `GATED.CONF`. They also end in a new line instead of a semicolon (;) like the other statements.

Format

`%directory` *`directory`*

Defines the directory where the include files go if you do not fully specify directory as part of the filename in the %include statement. Does not actually change the current directory, but simply applies the directory prefix.

`%include` *`filename`*

Identifies an include file. GateD includes the contents of the file in GATED.CONF at the point where the %include appears. If you do not fully specify the filename, it is relative to the directory defined in %directory. The %include directive causes GateD to parse the specified file completely before resuming. You can nest up to ten levels of include files.

traceoptions

traceoptions — The **traceoptions** statement controls tracing options. You can configure GateD tracing options at many levels. These include file specifications, control options, and global and protocol-specific tracing options. Lower levels of statements inherit tracing options from the next higher level, unless overridden.

Format

`traceoptions` ["tracefile" [replace] [size *`size`* [k | m] files *`files`*]] [nostamp] traceoptions [except *`traceoptions`*] | none ;

Options and Parameters

"tracefile"

File to receive tracing information. If this filename is not fully specified, GateD creates it in the directory where you started GateD.

[replace]

Replaces an existing file. The default is to append to an existing file.

[size *size* [k | m] files *files*]

Limits the maximum size, in k or m or the files indicated, of the trace file (the minimum is 10k). When the file reaches size, GateD creates a new version.

[nostamp]

Control option which means not to prepend a timestamp to all trace lines. The default is to prepend a timestamp.

traceoptions

Specific to each protocol statement. Note that these global options may not apply to all protocols.

[except *traceoptions*]

Disables more specific trace options after enabling broader ones.

none

Turns off all tracing for the protocol or peer.

Table B.3. Global Trace Options

Option	Description
`adv`	For debugging: traces the allocation and freeing of policy blocks.
`all`	Turns on the general, normal, policy, route, state, task, and timer options.
`general`	Shorthand for specifying both the normal and route options.
`iflist`	Traces reading of the kernel interface. Useful to specify this with the -t option on the command line since the first interface scan occurs before reading the configuration file.
`normal`	Traces normal protocol occurrences (abnormal protocol occurrences are always traced).
`parse`	For debugging: traces the lexical analyzer and parser.
`policy`	Traces how protocol and user-specified policy apply to routes imported and exported.
`route`	Traces routing table changes for routes installed by the protocol or peer.
`state`	Traces state machine transitions in the protocols.
`symbols`	Traces symbols read from the kernel at startup. The only useful way to specify this level of tracing is to use the -t option on the command line, since

Option	Description
	GateD reads the symbols from the kernel before parsing the configuration file.
`task`	Traces system interface and processing associated with the protocol or peer.
`timer`	Traces timer usage by the protocol or peer.

options

options — The options statements let you specify some global options. If used, options must appear before any other type of configuration statement in `GATED.CONF`.

Format

`options` [nosend] [noresolv] [gendefault [preference *`value`*] [*`gateway host`*]] [syslog [upto]*`loglevel`*] [mark *`time`*]

Options and Parameters

[nosend]

Does not send packets. Makes it possible to run GateD on a live network to test protocol interactions, without actually participating in the routing protocols. You can examine the packet traces in the GateD log to verify that GateD functions properly. Most useful for RIP and HELLO. Does not yet apply to BGP, and not useful with EGP and OSPF.

[noresolv]

Does not resolve symbolic names into IP addresses. By default, GateD uses the **gethostbyname()** and **getnetbyname()** library calls that usually use the Domain Name System (DNS) instead of the host's local host and network tables. If there is insufficient routing information to send DNS queries, GateD deadlocks during startup. Use this option to prevent these calls.

Note

When you use this option, symbolic names cause configuration file errors.

[gendefault [preference *`value`*] [*`gateway host`*] nogendefault]

Creates a default route with the special protocol default when a BGP or EGP neighbor is up. You can disable this for each BGP/EGP group with the **nogendefault** option. By default, this route has a **preference** value of **20**. This route is normally not installed in the kernel forwarding table; it is only present for announcement to other protocols. The **gateway** option installs the default route in the kernel forwarding table with a next hop of the gateway defined.

Note

Using more general options is preferred to using **gendefault**. (See aggregate for details on the **generate** statement.)

[syslog [upto]*`loglevel`*]

Amount of data GateD logs to OPCOM. OpenVMS systems map UNIX syslog logging levels to OPCOM severity levels. The default is **syslog upto info**. The mapping of syslog to OPCOM logging levels appears in the Table B.4.

[mark *time*]

GateD sends a message to the trace log at the specified *time* interval. Can be one method of determining if GateD is still running.

Table B.4. Mapping of UNIX syslog Levels to OpenVMS OPCOM Severity Levels

syslog log level	Is equivalent to OPCOM level...
emerg	FATAL
alert	FATAL
crit	FATAL
err	ERROR
warning	WARNING
notice	INFORMATIONAL
info (default)	INFORMATIONAL
debug	INFORMATIONAL

Example

```
#  generate a default route when peering with an EGP or BGP neighbor:
#
options gendefault ;
```

interfaces

interfaces — An interface is the connection between a router and one of its attached networks. Specify a physical interface by interface name, IP address, or domain name. Multiple reference levels in the configuration language let you identify interfaces using wildcards (only the device driver part of the name, to match any unit number), interface type names, or addresses.

Format

```
interfaces {
 options
  [strictinterfaces]
  [scaninterval time] ;
 interface list
  [preference value]
  [down preference value]
  [passive]
  [simplex]
  [reject]
  [blackhole] ;
define address
  [broadcast address] | [pointtopoint address]
  [netmask mask]
  [multicast] ;
```

```
} ;
```

Options Clause

```
options
 [strictinterfaces]
 [scaninterval time] ;
strictinterfaces
```

Makes it a fatal error to use reference interfaces not present when you start GateD or that are not part of the **define** parameter. Normally, GateD issues a warning message and continues.

`scaninterval` *`time`*

Sets how often GateD scans the kernel interface list for changes. The default is every 15 seconds on most systems, and 60 seconds on systems that pass interface status changes through the routing socket (such as BSD 4.4).

Interface Clause

Sets interface options on the specified interfaces. A *`list`* can consist of interface names, domain names, numeric addresses, or the value **all**. Include one or more interface names, including wildcard names (without a number) and those that can specify more than one interface or address.

There are three ways to reference an interface:

- **By wildcard**: Only the device driver part of the name, to match any unit number.
- **By name**: Combined device driver and unit number of an interface.
- **By address**: IP address or domain name (if resolving to one address only).

There are four types of interfaces allowed:

- **Loopback:** Must have the address 127.0.0.1. Packets from a loopback interface go back to the originator. Also used for reject and blackhole routes (not supported in VSI TCP/IP). The interface ignores any net mask. It is useful to assign an additional address to the loopback interface that is the same as the OSPF or BGP router ID; this allows routing to a system based on router ID that works if some interfaces are down.
- **Broadcast:** Multiaccess interface capable of physical level broadcast, such as Ethernet, Token-Ring, and FDDI. A broadcast interface has an associated subnet mask and broadcast address. The interface route to a broadcast network is a route to the complete subnet.
- **Point-to-point:** Tunnel to another host, usually on some sort of serial link. A point-to-point interface has a local address and a remote address. The remote address must be unique among the interface addresses on a given router. Many point-to-point interfaces and up to one non point-to-point interface must share the local address. This conserves subnets as you do not need any when using this technique. If you use a subnet mask on a point-to-point interface, only RIP version 1 and HELLO use it to determine which subnets propagate to the router on the other side of the point-to-point interface.
- **Nonbroadcast multiaccess (NBMA):** Multiaccess but not capable of broadcast, such as frame relay and X.25. This type of interface has a local address and a subnet mask.

`preference` *`value`*

Sets the preference for routes to this interface when it is up and GateD determines it to function properly. The default preference *value* is **0**. While the preference statement is optional, it is strongly recommended that you set an explicit preference value if you do use it.

`down preference` *value*

Sets the preference for routes to this interface when GateD determines that it does not function properly, but the kernel does not indicate that it is down. The default down preference *value* is **120**.

`passive`

Does not change the preference of the route to the interface if determined not to function properly from lack of routing information. GateD checks this only if the interface actively participates in a routing protocol.

`simplex`

The interface does not recognize its own broadcast packets. Some systems define an interface as simplex with the IFF_SIMPLEX flag. On others, the configuration defines it. On simplex interfaces, packets from the local host are assumed to have been looped back in software and are not used to indicate that the interface functions properly.

`reject, blackhole`

Not supported in VSI TCP/IP. Normally, this uses the address of the interface that matches these criteria as the local address when installing reject routes in the kernel. A blackhole route is like a reject route except that it does not support **unreachable** messages.

Define Clause

```
interfaces {
 define address
  [broadcast address] | [pointtopoint address]
  [netmask mask]
  [multicast] ;
} ;
```

Defines interfaces not present when starting GateD so that the configuration file can reference them when using options strictinterfaces.

`broadcast` *address*

Makes the interface broadcast-capable (for Ethernet or Token-Ring) and specifies the broadcast address.

`pointtopoint` *address*

Makes the interface point-to-point (such as SLIP or PPP) and specifies the address on the local side of the interface. The first address in the **define** statement references the host on the remote end of the interface.

An interface not defined as **broadcast** or **pointtopoint** must be nonbroadcast multiaccess (NBMA), such as for an X.25 network.

`netmask` *mask*

Subnet mask to use on the interface. Ignored on point-to-point interfaces.

`multicast`

Makes the interface multicast-capable.

Examples

1. This example sets the interface as passive.

```
#  do not mark interface 192.168.95.41 as down,
#  even if there is no traffic:
#
interfaces{
  interface 192.168.95.41 passive ;
} ;
```

- This example shows the interface statements used with the **rip** statement (see the **rip** description). Users would receive RIP packets only from interfaces sva-0 and sva-1, but not from fza-0, and sva-1 would be the only one that could send them.

```
rip yes {
  interface all noripin noripout ;
  interface sva ripin
;
  interface sva-1 ripout ;
  } ;
```

Definition Statements

Definition Statements — You can use the following definition statements: **autonomoussystem**, **routerid**, **martians**. Definition statements are general configuration statements that relate to all of GateD or at least to more than one protocol. You must use these statements for any protocol statements in the configuration file.

Format

`autonomoussystem` *`ASnumber`* [loops *`number`*];

An autonomous system (AS) is a set of routers under a single technical administration, using an internal protocol and common metrics to route packets within the AS, and an external protocol to route packets to other ASs. The Network Information Center (NIC) assigns AS numbers.

The **autonomoussystem** statement sets the AS number of the router. You require this option if using BGP or EGP. The **loops** option is only for protocols supporting AS paths, such as BGP. It controls the number of times this AS can appear in an AS path, and defaults to **1**.

`routerid`*`host`* ;

A router ID is an IP address used as a unique identifier assigned to represent a specific router, usually the address of an attached interface. The **routerid** statement sets the router ID for the BGP and OSPF protocols. The default is the address of the first interface GateD encounters. The address of a non-point-to-point interface is preferred over the local address of a point-to-point interface, and an address on a loopback interface that is not the loopback address (127.0.0.1) is most preferred.

```
martians {
 host host [allow] ;
 network [allow] ;
 network mask mask [allow] ;
 network masklen number [allow] ;
```

```
default [allow] ;
} ;
```

The martians statement defines a list of invalid addresses, called *martians*, that the routing software ignores. Sometimes a misconfigured system sends out obviously invalid destination addresses. The statement allows additions to the list of martian addresses. (For details on specifying ranges, see Route Filtering section).

You can also use the **allow** parameter to explicitly allow a subset of an otherwise disallowed range.

Example

This example shows the use of all three definition statements, **autonomoussystem**, **routerid**, and **martians**.

```
#  use AS number 249:
#
autonomoussystem 249 ;
#
#  set the router
ID number:
#
routerid 192.168.95.41 ;
#
#  prevent routes to
0.0.0.26 from ever being accepted:
#
martians {
host 0.0.0.26 ;
};
```

Route Filtering

Route Filtering — You can filter routes by matching a certain set of routes by destination, or by destination and mask. Use route filters on **martians**, **import**, and **export** statements. The action taken when no match is found depends on the context. For example, import and export route filters assume an **all reject ;** at the end of a list. A route matches the most specific filter that applies. Specifying more than one filter with the same destination, mask, and modifiers generates an error.

Format

```
network [exact | refines | allow]
network mask mask [exact | refines]
network masklen number [exact | refines]
all
default
host host
```

Options and Parameters

network

Destination network IP address. You can use one of the following options:

- **exact**: Destination mask must match the supplied mask exactly. Used to match a network, but no subnets or hosts of that network.

- **refines**: Destination mask must be more specified (longer) than the filter mask. Used to match subnets or hosts of a network, but not the network.
- **allow**: See the martians definition statement.

mask*mask*

Destination network mask.

masklen *number*

Length of the destination network mask.

all

Entry matches anything. Equivalent to **0.0.0.0 mask 0.0.0.0.**

default

Matches the default route. To match, the address must be the default address and the mask must be all zeros. Equivalent to **0.0.0.0 mask 0.0.0.0 exact.** (Not valid for **martians** statements.)

host *host*

Matches the specific host. To match, the address must match exactly the specified host, and the network mask must be a host mask (all 1s). Equivalent to **host mask 255.255.255 exact.** (Not valid for **martians** statements.)

rip

rip — GateD supports the Routing Information Protocol (RIP). RIP is a distance-vector protocol for distributing routing information at the local network level of the Internet. In distance-vector routing, each router transmits destination addresses and costs to its neighbors (computers communicating over RIP).

Description

RIP versions 1 and 2 are the most commonly used interior protocol. RIP selects the route with the lowest metric as the best route. The metric is a hop count representing the number of gateways through which data must pass to reach its destination. The longest path that RIP accepts is 15 hops. If the metric is greater than 15, a destination is considered unreachable and GateD discards the route. RIP assumes the best route uses the fewest gateways, that is, the shortest path, not taking into account congestion or delay along the way.

RIP uses two types of packets: requests and responses.

Requests. A request asks for information about specific destinations or for all destinations. RIP can send requests when a router:

- Comes up
- Receives timed-out information about a destination

If a request fails to specify a destination, RIP assumes the router requests information about all destinations.

Responses. Responses contain destination and cost pairs. RIP sends responses under the following three conditions:

- In response to a request
- When information changes; for example, cost information
- At set intervals; for example, reporting the destination to each neighbor every 30 seconds

RIP discards the destination and cost information if a neighbor fails to report the distance to a destination after a certain time interval.

RIP IP Addresses RIP version 1 contains no provision for passing around a mask. RIP infers the mask based on whether the address is class A, B, or C. Sometimes there are special cases when the inferred mask differs from class A, B, or C. For example:

- When you use RIP with a subnet (in this case the routers must know the subnet mask for a particular network number)
- When the system updates RIP with an address reported as 0.0.0.0, RIP considers this address as a default destination with a mask of 0.0.0.0
- When the system updates RIP with bits set in the host portion of the address, RIP assumes the address refers to a host with a mask of 255.255.255.255

With RIP version 2, you can specify the network mask with each network in a packet.

Configuring RIP. You configure RIP in the `GATED.CONF` file using a GateD protocol statement that enables or disables RIP. The syntax of the rip statement is as follows, with the parameters described next.

Format

```
rip  yes | no | on | off
 [{[no]broadcast ;
 nocheckzero ;
 preference value ;
 defaultmetric metric ;
 query authentication [ none | [ [simple | md5] password ] ] ;
 interface list
  [[no]ripin ] [ [no]ripout ]
  [metricin metric]
  [metricout metric] ;
  [[version 1] | [ version 2 [multicast | broadcast] ]]
  [ [secondary] authentication [ none | [ [simple | md5] password ] ] ] ;
trustedgateways list ;
sourcegateways list ;
traceoptions options ;
}] ;
```

Options and Parameters

yes | on (default)

no | off

When enabled on a host, RIP listens in the background to routing updates. When enabled on a gateway, RIP supplies routing updates. Enabled by default.

[broadcast ;]

Broadcasts RIP packets regardless of the number of interfaces present. Useful when propagating static routes or routes learned from another protocol into RIP. In some cases, using **broadcast** when only one network interface is present can cause data packets to traverse a single network twice. The default for more than one interface.

[nobroadcast ;]

Does not broadcast RIP packets on attached interfaces even if there is more than one. If you use the **sourcegateways** parameter, routes are still unicast directly to that gateway. The default for a single interface.

[nocheckzero ;]

Does not make sure that reserved fields in incoming RIP version 1 packets are zero. Normally RIP rejects packets whose reserved fields are zero.

[preference *value* ;]

Sets the preference for routes learned from RIP. A preference specified in import policy can override this. The default preference *value* is **100**.

[defaultmetric *metric* ;]

Metric used when advertising routes learned from other protocols. Choice of values requires that you explicitly specify a metric in order to export routes from other protocols into RIP. A metric specified in export policy can override this. The default *metric* is **16**.

[query authentication ;]

Authentication required of query packets that do not originate from routers. The default is **none**.

Interface Clause

```
rip  yes | no | on | off
 [{[no]broadcast ;
 nocheckzero ;
 preference value ;
 defaultmetric metric ;
 query authentication [ none | [ [simple | md5] password ] ] ;
 interface list
  [ [no]ripin ] [ [no]ripout ]
  [metricin metric]
  [metricout metric] ;
  [version 1] | [ version 2 [multicast | broadcast] ]
  [ [secondary] authentication [none | [ [simple | md5] password] ] ;
 trustedgateways list ;
 sourcegateways list ;
 traceoptions options ;
}] ;
```

Controls various attributes of sending RIP on specific interfaces. (See the interfaces statement for a description of *list*.) Note that if there are multiple interfaces configured on the same subnet, only

the first one on which RIP output is configured sends the RIP updates. This limitation is required because of the way the UNIX kernel operates. A future GateD release will hopefully remove this limitation. The default *list* value is all.

[ripin(default)] [noripin]

Use **ripin** explicitly when using **noripin** on a wildcard interface descriptor. The **noripin** option ignores RIP packets received over the specified interfaces.

[ripout (default)] [noripout]

Use **ripin** explicitly when using **noripout** on a wildcard interface descriptor. The **noripin** does not send RIP packets over the specified interfaces.

[metricin *metric*]

RIP metric to add to incoming routes before they are installed in the routing table. Makes the router prefer RIP routes learned using the specified interfaces less than those learned from other interfaces. The default is the kernel interface metric plus 1. If using this as the absolute value, the kernel metric is not added.

[[metricout *metric*]]

RIP metric to add to routes sent over the specified interface(s). Makes other routers prefer other sources of RIP routes over this router. The default *metric* value is **0**.

[version 1 (default)]

Sends RIP version 1 packets over the specified interface(s).

[version 2 [multicast | broadcast]]

Sends RIP version 2 packets over the specified interfaces. If IP multicasting support is available on this interface, the default is to send full version 2 packets. If multicasting is not available, version 1 compatible version 2 packets are sent. Options include:

- **multicast:** Multicasts RIP version 2 packets over this interface. This is a default value.
- **broadcast:** Broadcasts RIP version 1 compatible version 2 packets over this interface even if IP multicasting is available

[[secondary] authentication [none | [[simple | md5] password]]]

Authentication type to use. Applies only to RIP version 2 and is ignored for RIP-1 packets. If you specify a *password*, the authentication type defaults to **simple**. The password should be a quoted string with 0 to 16 characters. If you specify **secondary**, this defines the secondary authentication. The default is **authentication none**.

Remaining Options and Parameters

trustedgateways *list* ;

List of gateways from which RIP accepts updates (host names or IP addresses). If used, only updates from the gateways in the list are accepted. The default *list* va**all**lue is .

[[sourcegateways *list* ;]]

List of routers to which RIP sends packets directly, not through multicasting or broadcasting. If used, only updates from the gateways in the list are accepted. The default *list* value is **all**.

traceoptions *options* ;

RIP-specific trace options:

- **packets**: All RIP packets, or packets [detail] send or [detail] recv (detail provides a more verbose format to provide more details; if used, detail must come before send or recv).
- **request**: RIP information request packets, such as REQUEST, POLL and POLLENTRY.
- **response**: RIP RESPONSE packets that actually contain routing information.

hello

hello — GateD supports the HELLO protocol. HELLO is an interior protocol that uses delay as the deciding factor when selecting the best route. Delay is the round trip time between source and destination. HELLO is not as widely used as when it was the interior protocol of the original 56-Kb/sec NSFNET backbone and used between LSI-11 ("fuzzball") routers. Because of this, HELLO is disabled by default.

Description

By default, HELLO, like RIP, uses the kernel interface metric set by the **ifconfig** command to influence metrics added to routes as they are installed in the routing table (**metricin**). Since the kernel interface metric is in hops, it must be translated into HELLO's millisecond metric. For the translation scheme, see Table B.5.

Table B.5. HELLO Hops-to-Metrics Translation

This many Hops	Translate to this HELLO metric	This many Hops	Translate to this HELLO metric	This many Hops	Translate to this HELLO metric
0	0	6	713	12	75522
1	100	7	1057	13	11190
2	148	8	1567	14	16579
3	219	9	2322	15	24564
4	325	10	3440	16	3000
5	481	11	5097		

You configure HELLO in the `GATED.CONF` file using a GateD protocol statement that enables or disables HELLO.

When enabled, HELLO assumes **nobroadcast** when only one interface exists. HELLO assumes broadcast when more than one interface exists.

Format

```
hello  yes | no | on | off
```

```
 [{[no]broadcast ;
  preference value ;
  defaultmetric metric ;
  interface list
   [ [no]helloin ]
   [ [no]helloout ]
   [metricin metric]
   [metricout metric] ;
  trustedgateways list ;
  sourcegateways list ;
  traceoptions options ;
}]
;
```

Options and Parameters

yes | on

no | off(default)

When enabled on a host, HELLO listens in the background for routing updates. When enabled on a gateway, HELLO supplies routing updates. Disabled by default.

broadcast ;

nobroadcast ;

The **broadcast** option broadcasts HELLO packets regardless of the number of interfaces present. Useful when propagating static routes or routes learned from another protocol into HELLO. In some cases, using **broadcast** when only one network interface is present can cause data packets to traverse a single network twice. The default for more than one interface.

The **nobroadcast** option does not broadcast HELLO packets on attached interfaces, even if there is more than one. If you use the **sourcegateways** parameter, routes are still unicast directly to that gateway. The default for a single interface.

preference *value* ;

Preference for routes learned from HELLO. A preference specified in import policy can override this. The default preference *value* is **90**.

defaultmetric *metric* ;

Metric used when advertising routes learned from other protocols. Requires you to explicitly specify a metric in order to export routes from other protocols into HELLO. A metric specified in export policy can override this. The default *metric* is 30000.

Interface Clause

```
interface list
 [ [no]helloin ]
 [ [no]helloout ]
 [metricin metric]
 [metricout metric] ;
```

Controls various attributes of sending HELLO on specific interfaces. (See interfaces statement for a description of *list*.) Note that if there are multiple interfaces configured on the same subnet, only

the first interface that has HELLO output configured sends the HELLO updates. This limitation is required because of the way the UNIX kernel operates. A future GateD release will hopefully remove this limitation. The default interface *list* value is all.

[helloin (default)] [nohelloin]

Use helloin explicitly when using nohelloin on a wildcard interface descriptor. The nohelloin option ignores HELLO packets received over the specified interfaces.

[helloout (default)] [nohelloout]

Use **helloout** explicitly when using **nohelloout** on a wildcard interface descriptor. The **nohelloout** option does not send HELLO packets over the specified interfaces.

[metricin *metric*]

HELLO metric to add to incoming routes before GateD installs them in the routing table. Makes this router prefer HELLO routes learned from other interfaces over those from the specified interface(s). The default is the kernel interface metric plus one. If using this as the absolute value, GateD does not add the kernel metric to the routing table.

[metricout *metric*]

HELLO metric to add to routes that are sent over the specified interface(s). Makes other routers prefer other sources of HELLO routes over this router. The default metric out *metric* value is **0**.

Remaining Options and Parameters

trustedgateways *list* ;

List of gateways from which HELLO accepts updates (host names or IP addresses). If used, HELLO accepts only updates from the gateways in the list. The default *list* value is **all**.

[sourcegateways *list* ;]

List of routers to which HELLO sends packets directly, not through multicasting or broadcasting. If used, HELLO accepts only updates from the gateways in the list. The default *list* value is **all**.

[traceoptions *options* ;]

All HELLO packets, or packets **[detail]**

send or **[detail]**

recv (**detail** provides a more verbose format to provide more details; if used, **detail** must come before **send** or **recv**).

icmp

icmp — On systems without the BSD routing socket, GateD listens to **ICMP** messages received by the system. Processing of **ICMP** redirect messages is handled by the redirect statement. Currently the only reason to specify the icmp statement is to be able to trace the **ICMP** messages that GateD receives.

Format

`icmp` {traceoptions *`options`* ;}

Options and Parameters

{traceoptions *`options`* ;}

ICMP tracing options (which you can modify with **detail** and **recv**) are as follows:

- **packets**: All **ICMP** packets received.
- **redirect**: Only **ICMP** Redirect packets received.
- **routerdiscovery**: Only **ICMP** Router Discovery packets received.
- **info**: Only **ICMP** informational packets, which include mask request/response, info request/response, echo request/response and timestamp request/response.
- **error**: Only **ICMP** error packets, which include time exceeded, parameter problem, unreachable and source quench.

redirect

redirect — GateD controls whether ICMP redirect messages can modify the kernel routing table. If disabled, GateD only prevents a system from listening to ICMP redirects. By default, ICMP redirects are enabled on hosts, and disabled on gateways that run as **RIP** or HELLO suppliers. You configure ICMP redirect handling in the `GATED.CONF` file using a GateD protocol statement.

Format

```
redirect  yes | no | on | off
 [{preference value ;
  interface list [ [no]redirects ] ;
  trustedgateways list ;
}]
;
```

Options and Parameters

yes | no | on | off

Enabled by default on hosts. Disabled by default on gateways running as RIP or HELLO suppliers.

preference *`value`* ;

Preference for routes learned from a redirect. The default preference *`value`* is **30**.

interface *`list`* [[no]redirects] ;

Enables and disables redirects interface by interface. (See **interfaces** for a description of *`list`*.) The default interface *`list`* value is **all**. The possible parameters are:

- **redirects**: May be necessary when you use noredirects on a wildcard interface descriptor. This is a default value.

- **Noredirects**: Ignores redirects received over the specified interface(s). The default is to accept redirects on all interfaces.

trustedgateways *list* ;

List of gateways from which redirects are accepted (host names or addresses). By default, all routers on the shared network(s) are trusted to supply redirects. If used, only redirects from the gateways in the list are accepted. The default *list* value is **all**.

routerdiscovery server

routerdiscovery server — The Router Discovery Protocol is an IETF standard protocol used to inform hosts of the existence of routers without having hosts wiretap routing protocols such as RIP. Use it in place of, or in addition to, statically configured default routes in hosts.

Description

The protocol is in two parts, the server that runs on routers and the client that runs on hosts (see the next statement). GateD treats these much like two separate protocols that you can enable only one at a time.

The Router Discovery Server runs on routers and announces their existence to hosts. It does this by periodically multicasting or broadcasting a Router Advertisement to each interface on which it is enabled. These Router Advertisements contain a list of all router addresses on a given interface and their preference for use as a default router.

Initially these Router Advertisements occur every few seconds, then fall back to occurring every few minutes. In addition, a host may send a Router Solicitation to which the router will respond with a unicast Router Advertisement (unless a multicast or broadcast advertisement is due momentarily).

Each Router Advertisement contains an Advertisement Lifetime field indicating how long the advertised addresses are valid. This lifetime is configured such that another Router Advertisement is sent before the lifetime expires. A lifetime of zero indicates that one or more addresses are no longer valid.

On systems supporting IP multicasting, the Router Advertisements are sent to the all-hosts multicast address 224.0.0.1 by default. However, you can specify **broadcast**. When Router Advertisements are being sent to the all-hosts multicast address, or an interface is configured for the limited-broadcast address 255.255.255.255, all IP addresses configured on the physical interface are included in the Router Advertisement. When the Router advertisements are being sent to a net or subnet broadcast, only the address associated with that net or subnet is included.

Note

Do not mix **routerdiscovery server** and **routerdiscovery client** statements in the `GATED.CONF` file or you may get unintended results. You should also include **preference** statements in the **interfaces** and **routerdiscovery** statements whenever possible.

Format

```
routerdiscovery server  yes | no | on | off
 [{  traceoptions state ;
```

```
  interface list
  [minadvinterval time]
  [maxadvinterval time]
  [lifetime time] ;
 address list
  [advertise] | [ignore]
  [broadcast] | [multicast]
  [ineligible] | [preference value] ;
}] ;
```

Note

Interface *must* be mentioned in the “Interface” directive.

Options and Parameters

yes | on no | off

Enables or disables Router Discovery Protocol Server.

traceoptions state

The **state** is the only trace option, which traces the state transitions. The Router Discovery Server does not directly support packet tracing options; tracing of router discovery packets is enabled through the **icmp** statement described in the **icmp** statement section.

Interface Clause

interface *list*

Parameters that apply to physical interfaces. Note a slight difference in convention from the rest of GateD: **interface** specifies just physical interfaces, while **address** specifies protocol (in this case, IP) addresses.

[minadvinterval *time*]

Maximum time allowed between sending broadcast or multicast Router Advertisements from the interface. Must be no less than **4** and no more than **30:00** (30 minutes). The default is **10:00** (10 minutes).

[maxadvinterval *time*]

Minimum time allowed between sending unsolicited broadcast or multicast Router Advertisements from the interface. Must be no less than 3 seconds and no greater than **maxadvinterval**. The default is **0.75** X **maxadvinterval**.

[lifetime time]

Lifetime of addresses in a Router Advertisement. Must be no less than **maxadvinterval** and no greater than **2:30:00** (two hours, thirty minutes). The default is **3** X **maxadvinterval**.

Address Clause

address *list*

Parameters that apply to the specified set of addresses on this physical interface. Note a slight difference in convention from the rest of GateD: **interface** specifies just physical interfaces while **address** is protocol (in this case, IP) addresses.

[advertise (default)] | [ignore]

The **advertise** keyword includes the specified addresses in Router Advertisements. The **ignore** keyword does not.

[broadcast] | [multicast]

The **broadcast** keyword includes the given addresses in a broadcast Router Advertisement because this system does not support IP multicasting, or some hosts on an attached network do not support IP multicasting. It is possible to mix addresses on a physical interface such that some are included in a broadcast Router Advertisement and some are included in a multicast Router Advertisement. This is the default if the router does not support IP multicasting.

The **multicast** keyword includes the given addresses in a multicast Router Advertisement. If the system does not support IP multicasting, the address(es) is not included. If the system supports IP multicasting, the default is to include the addresses in a multicast Router Advertisement if the given interface supports IP multicasting. If not, the addresses are included in a broadcast Router Advertisement.

[ineligible] | [preference *value*]

The **preference** keyword sets the preferability of the addresses as a default router address, relative to other router addresses on the same subnet. A 32-bit, signed, two's complement integer, with higher values meaning more preferable. Note that hex 80000000 may only be specified as ineligible. The default value is **0**. Use a **preference** statement whenever possible.

The **ineligible** keyword assigns the given addresses a preference of hex 80000000, which means that it is not eligible to be the default route for any hosts. This is useful when the addresses should not be used as a default route, but are given as the next hop in an ICMP Redirect. This allows the hosts to verify that the given addresses are up and available.

routerdiscovery client

routerdiscovery client — A host listens for Router Advertisements through the all-hosts multicast address (224.0.0.2) if IP multicasting is available and enabled, or on the interface's broadcast address. When starting up, or when reconfigured, a host may send a few Router Solicitations to the all-routers multicast address, 224.0.0.2, or the interface's broadcast address.

Description

When a Router Advertisement with a non-zero lifetime is received, the host installs a default route to each of the advertised addresses. If the preference is ineligible, or the address is not on an attached interface, the route is marked unusable but retained. If the preference is usable, the metric is set as a function of the preference such that the route with the best preference is used. If more than one address with the same preference is received, the one with the lowest IP address will be used. These default routes are not exportable to other protocols.

When a Router Advertisement with a zero lifetime is received, the host deletes all routes with next hop addresses learned from that router. In addition, any routers learned from ICMP Redirects pointing

to these addresses will be deleted. The same happens when a Router Advertisement is not received to refresh these routes before the lifetime expires.

Note

Do not mix routerdiscovery server and routerdiscovery client statements in the `GATED.CONF` file or you may get unintended results. You should also include preference statements in the interfaces and routerdiscovery statements whenever possible.

Format

```
routerdiscovery client  yes | no | on | off
 [{    traceoptions state ;
  preference value ;
  interface list
   [enable] | [disable]
   [broadcast] | [multicast]
   [quiet] | [solicit] ;
}] ;
```

Options and Parameters

yes | no | on | off

Enables or disables the Router Discovery Protocol Client.

traceoptions state ;

The **state** is the only trace option, which traces the state transitions. The Router Discovery Server does not directly support packet tracing options; tracing of router discovery packets is enabled through the **icmp** statement described in the **icmp** statement section.

preference *value* ;

Preference of all Router Discovery default routes. Use a preference statement whenever possible. Default is **55**.

Interface Clause

Parameters that apply to physical interfaces. Note a slight difference in convention from the rest of GateD: **interface** specifies just physical interfaces. The Router Discovery Client has no parameters that apply only to interface addresses.

[enable (default)] | [disable]

Either performs or does not perform Router Discovery on the specified interfaces.

[broadcast] | [multicast]

The **broadcast** keyword broadcasts Router Solicitations on the specified interfaces. This is the default if IP multicast support is not available on this host or interface.

The **multicast** keyword multicasts Router Solicitations on the specified interfaces. If IP multicast is not available on this host and interface, no solicitation is performed. The default is to multicast Router Solicitations if the host and interface support it, otherwise Router Solicitations are broadcast.

[quiet] | [solicit (default)]

Either sends or does not send Router Solicitations on this interface, even though Router Discovery is performed.

egp

egp — GateD supports the Exterior Gateway Protocol (EGP). EGP is an exterior routing protocol that moves routing information between Autonomous Systems (ASs). Unlike interior protocols, EGP propagates only reachability indications, not true metrics. EGP updates contain metrics, called distances, which range from 0 to 255. GateD only compares EGP distances learned from the same AS. EGP currently has limited usage. By default, EGP is disabled.

Description

Before EGP sends routing information to a remote router, it must establish an adjacency with that router. This occurs by exchanging Hello and I Heard You (I-H-U) messages with that router. (Hello should not to be confused with the HELLO protocol, or OSPF HELLO messages.) Computers communicating over EGP are called EGP neighbors, and the exchange of Hello and I-H-U messages is known as acquiring a neighbor.

Once you acquire a neighbor, the system polls it for routing information. The neighbor responds by sending an update containing routing information. If the system receives a poll from its neighbor, it responds with its own update packet. When the system receives an update, it includes routes from the update into its routing database. If the neighbor fails to respond to three consecutive polls, GateD assumes that the neighbor is down and removes the neighbor's routes from its database.

You configure EGP in the `GATED.CONF` file using a GateD protocol statement.

Format

```
egp  yes | no | on | off
 [{  preference value ;
  defaultmetric metric ;
  packetsize max ;
  traceoptions options ;
  group
   [peeras ASnumber]
    [localas ASnumber]
   [maxup number]
  {neighbor host
   [metricout metric]
   [preference value]
   [preference2 value]
   [ttl ttl]
   [nogendefault]
   [importdefault]
   [exportdefault]
   [gateway gateway]
   [lcladdr local-address]
   [sourcenet network]
   [minhello | p1 time]
   [minpoll | p2  time]
    [traceoptions options] ;
  } ;
```

```
}] ;
```

Options and Parameters

yes | no | on | off (default)

Enables or disables EGP support. Disabled by default.

preference *value* ;

Preference for routes learned from EGP. A preference specified on the **group** or **neighbor** statements or by import policy can override this. The default preference *value* is **200**.

defaultmetric *metric* ;

Metric used when advertising routes over EGP. This choice of values requires you to explicitly specify a metric when exporting routes to EGP neighbors. A metric specified on the **neighbor** or **group** statements or in export policy can override this. The default *metric* is **255**.

packetsize *max* ;

Maximum size of a packet that EGP expects to receive from this neighbor. If EGP receives a larger packet, it is incomplete and EGP discards it. EGP notes the length of this packet and increases the expected size to be able to receive a packet of this size. Specifying the parameter prevents the first packet from being dropped. All packet sizes are rounded up to a multiple of the system page size. The default packet size *max* value is **8192**.

traceoptions *options* ;

Tracing options for EGP (can be overridden on a group or neighbor basis):

- packets: All EGP packets, or packets **[detail]** send or **[detail]** recv (detail provides a more verbose format to provide more details; if used, detail must come before send or recv).
- hello: EGP HELLO/I-HEARD-U packets used to determine neighbor reachability.
- acquire: EGP ACQUIRE /CEASE packets used to initiate and terminate EGP sessions.
- update: EGP POLL /UPDATE packets used to request and receive reachability updates.

Group Clause

```
group
 [peeras ASnumber]
 [localas ASnumber]
 [maxup number
{neighbor host
  [metricout metric]
  [preference value]
  [preference2 value]
  [ttl ttl]
  [nogendefault]
  [importdefault]
  [exportdefault]
  [gateway gateway]
  [lcladdr local-address]
```

```
  [sourcenet network]
  [minhello | p1 time]
  [minpoll | p2  time]
  [traceoptions options] ; } ;
```

EGP neighbors must be members of a group, which groups all neighbors in one AS. Parameters specified in the group clause apply to all the subsidiary neighbors, unless explicitly overridden on a neighbor clause. Any number of group clauses can specify any number of neighbor clauses. You can specify any parameters from the neighbor subclause on the group clause to provide defaults for the whole group (which you can override for individual neighbors).

The **group** clause is the only place to set the following attributes:

[peeras *ASnumber*]

AS number expected from peers in the group. Learned dynamically.

[localas *ASnumber*]

AS that GateD represents to the group. Usually only used when masquerading as another AS. Use is discouraged. Set globally in **autonomoussystem**.

[maxup *number*]

Number of neighbors GateD should acquire from this group. GateD attempts to acquire the first **maxup** neighbors in the order listed. If one of the first neighbors is not available, it acquires one farther down the list. If after startup, GateD does manage to acquire the more desirable neighbor, it drops the less desirable one. By default, GateD acquires all neighbors in the group.

Group Neighbor Clause

```
egp  yes | no | on | off
 [{  preference value ;
  defaultmetric metric ;
  packetsize max ;
  traceoptions options ;
  group
   [peeras ASnumber]
   [localas ASnumber]
   [maxup number
  {  neighbor host
   [metricout metric]
   [preference value]
   [preference2 value]
    [ttl ttl]
    [nogendefault]
    [importdefault]
    [exportdefault]
    [gateway gateway]
    [lcladdr local-address]
    [sourcenet network]
    [p1  time | minhello]
    [p2  time | minpoll]
    [traceoptions options] ; } ; }] ;
```

Each neighbor subclause defines one EGP neighbor within a group. The only required part of the subclause is the host argument, the symbolic host name or IP address of the neighbor.

[metricout *metric*]

Metric used for all routes sent to this neighbor. Overrides the default metric set in the **egp** statement and any metrics specified by export policy, but only for this specific neighbor or group of neighbors.

[preference *value*]

Preference used for routes learned from these neighbors. Can differ from the default EGP preference set in the **egp** statement, so that GateD can prefer routes from one neighbor, or group of neighbors, over another. Import policy can explicitly override this.

[preference2 *value*]

Tie-breaker, in the case of a preference tie. The default *value* is **0**.

[ttl *ttl*]

IPL time-to-live. Provided when attempting to communicate with improperly functioning routers that ignore packets sent with a TTL 1. The default *ttl* for local neighbors is **1**; the default for nonlocal neighbors is **255**.

[nogendefault]

Does not generate a default route when EGP receives a valid update from its neighbor. The default route is only generated when you enable the **gendefault** option.

[importdefault]

Accepts the default route (0.0.0.0) if included in a received EGP update. For efficiency, some networks have external routers announce a default route to avoid sending large EGP update packets. The default route in the EGP update is ignored.

[exportdefault]

Includes the default route (0.0.0.0) in EGP updates sent to this EGP neighbor. Allows the system to advertise the default route using EGP. Normally a default route is not included in EGP updates.

[gateway *gateway*]

Router on an attached network used as the next hop router for routes received from this neighbor if a network is not shared with a neighbor. Rarely used.

[lcladdr local-address]

Address used on the local end of the connection with the neighbor. The local address must be on an interface shared with the neighbor, or with the neighbor's gateway when using the **gateway** option. A session only opens when an interface with the appropriate local address (through which the neighbor or gateway address is directly reachable) is operating.

[sourcenet *network*]

Network queried in the EGP Poll packets. If there is no network shared with the neighbor, specify one of the networks attached to the neighbor. Also use to specify a network shared with the neighbor, other than the one on which the EGP packets are sent. Normally not needed. The default is the network shared with the neighbor's address.

[minhello | p1 *time*]

Minimum acceptable interval between the transmission of EGP HELLO packets. If the neighbor fails to respond to three hello packets, GateD stops trying to acquire the neighbor. Setting a larger interval gives the neighbor a better chance to respond. The **minhello** is an alias for the **p1** value defined in the EGP specification. The default *time* value is **30**.

[minpoll | p2 *time*]

Time interval between polls to the neighbor. If three polls are sent without a response, the neighbor is declared "down" and all routes learned from that neighbor are removed from the routing database. A longer polling interval supports a more stable routing database but is not as responsive to routing changes. The **minpoll** is an alias for the **p2** value defined in the EGP specification. The default time *value* is **120**.

[traceoptions *options*]

Tracing options for this EGP neighbor, which are:

- **packets**: All EGP packets, or packets [detail] send or **[detail]** recv (detail provides a more verbose format to provide more details; if used, detail must come before send or recv).
- hello: **EGP HELLO/I-HEARD-U** packets used to determine neighbor reachability.
- acquire: **EGP ACQUIRE /CEASE** packets used to initiate and terminate EGP sessions.
- update: **EGP POLL /UPDATE** packets used to request and receive reachability updates.

bgp

bgp — The Border Gateway Protocol (BGP) is an exterior routing protocol used to exchange routing information between multiple transit Autonomous Systems (ASs) as well as between transit and stub ASs. BGP is related to EGP but operates with more capability, greater flexibility, and less bandwidth required. BGP uses path attributes to provide more information about each route. It maintains an AS path, which includes the AS number of each AS the route transits, providing information sufficient to prevent routing loops in an arbitrary topology. You can also use path attributes to distinguish between groups of routes to determine administrative preferences. This allows greater flexibility in determining route preference to achieve a variety of administrative ends.

Description

BGP supports two basic types of sessions between neighbors — internal (sometimes called IBGP) and external. Internal sessions run between routers in the same AS, while external sessions run between routers in different ASs. When sending routes to an external peer, the local AS number is prepended to the AS path. Hence routes received from an external peer are guaranteed to have the AS number of that peer at the start of the path. Routes received from an internal neighbor do not generally have the local AS number prepended to the AS path. Hence, these routes generally have the same AS path the route had when the originating internal neighbor received the route from an external peer. Routes with no AS numbers in the path may be legitimately received from internal neighbors; these indicate that the received route should be considered internal to your own AS.

The BGP implementation supports three versions of the BGP protocol—versions 2, 3 and 4. BGP versions 2 and 3 are similar in capability and function. They only propagate classed network routes, and the AS path is a simple array of AS numbers. BGP version 4 propagates fully general address-

and-mask routes, and the AS path has some structure to represent the results of aggregating dissimilar routes.

External BGP sessions may or may not include a single metric, which BGP calls the Multi-Exit Discriminator (MED), in the path attributes. For BGP versions 2 and 3 this metric is a 16-bit unsigned integer; for BGP version 4 it is a 32-bit unsigned integer. In either case, smaller values of the metric are preferred. Currently this metric only breaks ties between routes with equal preference from the same neighbor AS. Internal BGP sessions carry at least one metric in the path attributes, which BGP calls the LocalPref. The size of the metric is identical to the MED. For BGP versions 2 and 3, this metric is better when its value is smaller; for version 4 it is better when it is larger. BGP version 4 sessions optionally carry a second metric on internal sessions, this being an internal version of the MED. The use of these metrics depends on the type of internal protocol processing specified.

BGP collapses routes with similar path attributes into a single update for advertisement. Routes received in a single update are readvertised in a single update. The churn caused by the loss of a neighbor is minimized, and the initial advertisement sent during peer establishment is maximally compressed. BGP does not read information from the kernel message by message, but fills the input buffer. It processes all complete messages in the buffer before reading again. BGP also does multiple reads to clear all incoming data queued on the socket. This feature may cause other protocols to be blocked for prolonged intervals by a busy peer connection.

All unreachable messages are collected into a single message and sent prior to reachable routes during a flash update. For these unreachable announcements, the next hop is set to the local address on the connection, no metric is sent, and the path origin is set to incomplete. On external connections the AS path in unreachable announcements is set to the local AS; on internal connections the AS path is set to zero length.

BGP implementation expects external peers to be directly attached to a shared subnet, and expects those peers to advertise next hops that are host addresses on that subnet (although this constraint can be relaxed by configuration for testing). For groups of internal peers, however, there are several alternatives that can be selected by specifying the group type. Type internal groups expect all peers to be directly attached to a shared subnet so that, like external peers, the next hops received in BGP advertisements may be used directly for forwarding. Type routing groups instead determine the immediate next hops for routes, by using the next hop received with a route from a peer as a forwarding address, and using this to look up an immediate next hop in an IGP’s routes. Such groups support distant peers, but need to be informed of the IGP whose routes they use to determine immediate next hops. Finally, type IGP groups expect routes from the group peers not to be used for forwarding at all. Instead, they expect that copies of the BGP routes are also received through an IGP, and that the BGP routes are only used to determine the path attributes associated with the IGP routes. Such groups also support distant peers and also need to be informed of the IGP with which they are running.

For internal BGP group types (and for test groups), where possible, a single outgoing message is built for all group peers based on the common policy. A copy of the message is sent to every peer in the group, with possible adjustments to the next hop field as appropriate to each peer. This minimizes the computational load of running large numbers of peers in these types of groups. BGP allows unconfigured peers to connect if an appropriate group was configured with an **allow** clause.

Format

```
bgp  yes | no | on | off
 [{preference value ;
  defaultmetric metric ;
  traceoptions options ;
  group type
```

```
   external peeras ASnumber
   | internal peeras ASnumber
   | igp peeras ASnumber proto proto
   | routing peeras ASnumber proto proto interface list
   | test peeras ASnumber
 { allow
  {   network
   network mask mask
   network masklen number
   all
   host host } ;
   peer host
     [metricout metric]
     [localas ASnumber]
     [nogendefault]
     [gateway gateway]
     [preference value]
     [preference2 value]
     [lcladdr local-address]
     [holdtime time]
     [version number]
     [passive]
     [sendbuffer number]
     [recvbuffer number]
     [indelay time]
     [outdelay time]
     [keep [all | none] ]
     [analretentive]
     [noauthcheck]
     [noaggregatorid]
     [keepalivesalways]
     [v3asloopokay]
     [nov4asloop]
     [logupdown]
     [ttl ttl]
     [traceoptions options] ;
 } ;
}] ;
```

Options and Parameters

yes | no | on | off

Enables or disables BGP support. Disabled by default.

preference *value* ;

Preference for routes learned from BGP. A preference specified on the **group** or **peer** statements, or by import policy, can override this. The default preference *value* is **170**.

defaultmetric *metric* ;

Metric used when advertising routes over BGP. A metric specified on the **group** or **peer** statements, or in export policy, can override this. The default *metric* is **65535**.

traceoptions *options* ;

Tracing options for BGP. May be overridden on a group or peer basis. The trace *options* are:

- **packets**: All BGP packets, or packets [detail] send or [detail] recv (detail provides a more verbose format to provide more details; if used, detail must come before send or recv).
- **open**: BGP OPEN packets used to establish a peer relationship.
- **update**: BGP UPDATE packets used to pass network reachability information.
- **keepalive**: BGP KEEPALIVE packets used to verify peer reachability.

Group Type Clause

peeras

For **group type**, specify one of the following **peeras** options:

Option	Description
external peeras *ASnumber*	In the classic external BGP group, full policy checking is applied to all incoming and outgoing advertisements. The external neighbors must be directly reachable through one of the machine's local interfaces. No metric included in external advertisements and the next hop is computed with respect to the shared interface.
internal peeras *ASnumber*	Internal group operating where there is no IP-level IGP; for example, an SMDS network or MILNET. All neighbors in this group must be directly reachable over a single interface. All next-hop information is computed with respect to this interface. Import and export policy may be applied to group advertisements. Routes received from external BGP or EGP neighbors are readvertised with the received metric.
igp peeras *ASnumber* proto proto	Internal group that runs in association with an interior protocol. The IGP group examines routes the IGP exports, and sends an advertisement only if the path attributes could not be entirely represented in the IGP tag mechanism. Only the AS path, path origin, and transitive optional attributes are sent with routes. No metric is sent, and the next hop is set to the local address the connection uses. Received internal BGP routes are not used or readvertised. Instead, the AS path information is attached to the corresponding IGP route and the latter is used for readvertisement. Since internal IGP peers are sent only a subset of the routes the IGP exports, the export policy used is the IGP's. There is no need to implement the "don't route from peers in the same group" constraint, since the advertised routes are routes that IGP already exports.
routing peeras *ASnumber* proto *proto* interface *list*	Internal group that uses the routes of an interior protocol to resolve forwarding addresses. A type routing group propagates external routes between routers not directly connected, and computes immediate next hops for these routes by using the BGP next hop that arrived with the route as a forwarding address to be resolved using an internal protocol's routing information. In essence, internal BGP is used to carry AS external routes, while the IGP is expected to only carry AS internal routes, and the latter

Option	Description
	is used to find immediate next hops for the former. The next hop in BGP routes advertised to the type routing peers are set to local address on BGP connection to those peers, as it is assumed a route to this address is propagated over IGP. • **proto** ***proto*** — Interior protocol used to resolve BGP route next hops, and can be the name of any IGP in the configuration. • **interface** ***list*** — Optionally provides a list of interfaces whose routes are carried over the IGP for which third party next hops can be used instead.
test peeras *ASnumber*	Extension to external BGP that implements a fixed policy using test peers. Fixed policy and special case code make test peers relatively inexpensive to maintain. Test peers do not need to be on a directly attached network. If GateD and the peer are on the same (directly attached) subnet, the advertised next hop is computed with respect to that network; otherwise the next hop is the local machine's current next hop. All routing information advertised by and received from a test peer is discarded, and all BGP advertisable routes are sent back to the test peer. Metrics from EGP- and BGP-derived routes are forwarded in the advertisement; otherwise no metric is included.

Group Type Allow Clause

Allows peer connections from any addresses in the specified range of network and mask pairs. Configure all parameters for these peers on the group clause. The internal peer structures are created when an incoming open request is received, and destroyed when the connection is broken. (For details on specifying the network/mask pairs, see Route Filtering section).

Group Type Peer Clause

Configures an individual peer. Each peer inherits all parameters specified on a group as defaults. You can override these defaults using parameters explicitly specified in the **peer** subclause. Allows the following parameters:

[metricout *metric*]

Primary metric on all routes sent to the specified peer(s). Overrides the default metric, a metric specified on the group, and any metric specified by export policy.

[localas *ASnumber*]

AS that GateD represents to this group of peers. *ASnumber* is set globally in **autonomoussystem**.

[nogendefault]

Does not generate a default route when EGP receives a valid update from its neighbor. The default route is generated only when enabling the **gendefault** option.

[gateway *gateway*]

If a network is not shared with a peer, specifies a router on an attached network used as the next hop router for routes received from this neighbor. Not needed in most cases.

[preference *value*]

Preference used for routes learned from these peers. Can differ from the default BGP preference set in the **bgp** statement, so that GateD can prefer routes from one peer, or group of peers, over others. Import policy can explicitly override this.

[preference2 *value*]

In the case of a preference tie, can break the tie.

[lcladdr *local-address*]

Address used on the local end of the TCP connection with the peer. For external peers, the local address must be on an interface shared with the peer or with the peer's gateway when using the **gateway** parameter. A session with an external peer only opens when an interface with the appropriate local address (through which the peer or gateway address is directly reachable) is operating. For other types of peers, a peer session is maintained when any interface with the specified local address is operating. In either case, incoming connections are only recognized as matching a configured peer if they are addressed to the configured local address.

[holdtime *time*]

BGP holdtime value to use when negotiating the connection with this peer, in seconds. According to BGP, if GateD does not receive a keepalive, update, or notification message within the period specified in the Hold Time field of the BGP Open message, the BGP connection is closed. The value must be either 0 (no keepalives are sent) or at least **3**.

[version *number*]

Version of the BGP protocol to use with this peer. If specified, only the specified version is offered during negotiation. Currently supported versions are **2**, **3**, and **4**. By default, the highest supported version is used first, and version negotiation is attempted.

[passive]

Does not attempt active OPENs to this peer. GateD should wait for the peer to issue an open. By default, all explicitly configured peers are active.

[sendbuffer *number*]

[recvbuffer *number*]

Controls the amount of send and receive buffering asked of the kernel. The maximum *number* supported is **65535** bytes, although many kernels have a lower limit. Not needed on normally functioning systems. By default, the maximum supported is configured.

[indelay *time*]

[outdelay *time*]

Dampens route fluctuations. The **indelay** is the amount of time a route learned from a BGP peer must be stable before it is accepted into the GateD routing database. The **outdelay** is the amount of time

a route must be present in the GateD routing database before it is exported to BGP. Default *time* in both cases is **0**.

[keep [all | none]]

Retains routes learned from a peer even if the routes' AS paths contain one of our exported AS numbers.

[analretentive]

Issues warning messages when receiving questionable BGP updates such as duplicate routes and/or deletions of nonexistent routes. Normally these events are silently ignored.

[noauthcheck]

Communicates with an implementation that uses some form of authentication other than the normal authentication field of all ones.

[noaggregatorid]

GateD should specify the routerid in the **aggregator** attribute as zero (instead of its routerid) in order to prevent different routers in an AS from creating aggregate routes with different AS paths.

[keepalivesalways]

GateD should always send keepalives, even when an update could have correctly substituted for one. Allows interoperability with routers that do not completely obey the protocol specifications on this point.

[v3asloopokay]

By default, GateD does not advertise routes whose AS path is looped (that have an AS appearing more than once in the path) to version 3 external peers. Setting this flag removes this constraint. Ignored when set on internal groups or peers.

[nov4asloop]

Does not advertise routes with looped AS paths to version 4 external peers. Can be useful to avoid advertising such routes to peer which would incorrectly forward the routes on to version 3 neighbors.

[logupdown]

Logs a message using syslog whenever a BGP peer enters or leaves ESTABLISHED state.

[ttl *ttl*]

Provided when attempting to communicate with improperly functioning routers that ignore packets sent with a TTL 1. Not all kernels allow the TTL to be specified for TCP connections. The default ttl for local neighbors is 1; the default for nonlocal neighbors is **255**.

[traceoptions *options*]

Tracing options for this BGP neighbor include:

- **packets**: All BGP packets, or packets [detail] send or [detail] recv (detail provides a more verbose format to provide more details; if used, detail must come before send or recv).
- **open**: BGP OPEN packets used to establish a peer relationship.

- **update**: BGP UPDATE packets used to pass network reachability information.
- **keepalive**: BGP KEEPALIVE packets used to verify peer reachability.

ospf

ospf — Open Shortest Path First (OSPF) routing is a shortest-path-first (SPF) or link-state protocol. OSPF is an interior gateway protocol that distributes routing information between routers in a single Autonomous System (AS). OSPF chooses the least cost path as the best path. Suitable for complex networks with many routers, OSPF provides equal cost multipath routing where packets to a single destination can be sent over more than one interface simultaneously. In a link-state protocol, each router maintains a database describing the entire AS topology, which it builds out of the collected link state advertisements of all routers. Each participating router distributes its local state (that is, the router's usable interfaces and reachable neighbors) throughout the AS by flooding.

Description

Each multiaccess network with at least two attached routers has a designated router and a backup designated router. The designated router floods a link state advertisement for the multiaccess network and has other special responsibilities. The designated router concept reduces the number of adjacencies required on a multiaccess network.

OSPF lets you group networks into areas. Routing information passed between areas is abstracted, which can significantly reduce routing traffic. OSPF uses four different types of routes, listed in order of preference—intra-area, inter-area, type 1 external, and type 2 external. Intra-area paths have destinations within the same area, while inter-area paths have destinations in other OSPF areas. AS External (ASE) routes are routes to destinations external to the AS. Routes imported into OSPF as type 1 routes are supposed to be from IGPs whose external metrics are directly comparable to OSPF metrics.

When making a routing decision, OSPF adds the internal cost of the AS Border router to the external metric. Type 2 ASEs are used for EGPs whose metrics are not comparable to OSPF metrics. In this case, GateD uses only the internal OSPF cost of the AS Border router in the routing decision.

From the topology database, each router constructs a tree of the shortest paths with itself as the root. This shortest-path tree gives the route to each destination in the AS. Externally derived routing information appears on the tree as leaves. The link-state advertisement format distinguishes between information acquired from external sources and from internal routers, so that there is no ambiguity about the source or reliability of routes. Externally derived routing information (for example, routes learned from EGP or BGP) passes transparently through the AS and is separate from OSPF's internally derived data. Each external route can also be tagged by the advertising router, enabling a passing of additional information between routers on the borders of the AS.

OSPF optionally includes type of service (TOS) routing and allows administrators to install multiple routes to a given destination for each type of service (such as for low delay or high throughput.) A router running OSPF uses the destination address and the TOS to choose the best route to the destination.

OSPF intra- and inter-area routes are always imported into the GateD routing database with a preference of 10. It would be a violation of the protocol if an OSPF router did not participate fully in the area's OSPF, so it is not possible to override this. Although it is possible to give other routes lower preference values explicitly, it is ill-advised to do so.

Hardware multicast capabilities are also used where possible to deliver link-status messages.

OSPF areas are connected by the backbone area, the area with identifier 0.0.0.0. All areas must be logically contiguous and the backbone is no exception. To permit maximum flexibility, OSPF allows the configuration of virtual links to enable the backbone area to appear contiguous when they are actually not.

All routers in an area must agree on that area's parameters. A separate copy of the link-state algorithm is run for each area. Because of this, most configuration parameters are defined on a per area basis. All routers belonging to an area must agree on that area's configuration. Misconfiguration leads to adjacencies not forming between neighbors, and routing information might not flow, or even loop.

Authentication. You can authenticate OSPF protocol exchanges. Authentication guarantees that routing information is imported only from trusted routers, to protect the Internet and its users. There are two authentication schemes available. The first uses a simple authentication key of up to eight characters and is standardized. The second is still experimental and uses the **MD5** algorithm and an authentication key of up to 16 characters.

The simple password provides very little protection, because in many cases it is possible to easily capture packets from the network and learn the authentication key. The experimental MD5 algorithm provides much more protection, as it does not include the authentication key in the packet.

The OSPF specification currently specifies that you configure the authentication type per area with the ability to configure separate passwords per interface. This was extended to allow configuration of different authentication types and keys per interface. Also, you can specify both a primary and a secondary authentication type and key on each interface. Outgoing packets use the primary authentication type, but incoming packets may match either the primary or secondary authentication type and key.

You configure OSPF in the `IP$:GATED.CONF` file using a GateD protocol statement.

Format

```
ospf  yes | no | on | off
[{defaults
 {preference value ;
  cost cost ;
  tag [as] tag ;
  type 1 | type 2 ;
 } ;
exportlimit routes ;
exportinterval time ;
traceoptions options;
monitorauthkey key ;
monitorauth  none | [simple | md5] authkey ;
 backbone | area area
  { authtype 0 | authtype 1 | none | simple ;
   stub [cost cost] ;
   networks
   {network [restrict] ;
   network mask mask [restrict] ;
   network masklen number [restrict] ;
   host host [restrict] ;
   } ;
   stubhosts
   {host cost cost ; } ;
   interface list [cost cost]
   {interface-parameters } ;
```

```
   interface list nonbroadcast [cost cost]
   {pollinterval time ;
    routers
    {gateway [eligible] ; } ;
    interface-parameters
    } ;
     /* Backbone only: */
    virtuallink neighborid router-id transitarea area
    {interface-parameters } ;
  } ;
}] ;
```

Options and Parameters

yes | no | on | off

Enables or disables OSPF support.

defaults

Defaults used when importing OSPF ASE routes into the GateD routing table, and exporting routes from the GateD routing table into OSPF ASEs, including:

Parameter	Description
preference *value*	How OSPF routes compete with routes from other protocols in the GateD routing table. The default preference *value* is **150**.
cost *cost*	Used when exporting a non-OSPF route from the GateD routing table into OSPF as an ASE. Export policy can explicitly override this. The default *cost* is **1**.
tag *[as] tag*	OSPF ASE routes have a 32-bit tag field that the OSPF protocol does not use, but export policy can use it to filter routes. When OSPF interacts with an EGP, you can use the tag field to propagate AS path information. In this case you would specify the **as** keyword and the tag is limited to 12 bits of information. The default *tag* value is **0**.
type 1 | type 2	Export policy can explicitly change and override the default here. The default is **type 1**.

exportlimit *routes*

How many ASEs are generated and flooded in each batch. The default export limits *routes* value is **100**.

exportinterval *time*

How often a batch of ASE link state advertisements are generated and flooded into OSPF. The default export interval *time* value is **1** (once per second).

traceoptions *options*

In addition to the following OSPF specific trace flags, OSPF supports the state which traces interface and neighbor state machine transitions:

`lsabuild`	Link State Advertisement creation
`spf`	Shortest Path First (SPF) calculations

`lsatransmit`	Link State Advertisement (LSA) transmission
`lsareceive`	LSA reception
`state`	State transitions

Packet tracing options (which you can modify with **detail**, **send**, and **recv**):

`hello`	OSPF HELLO packets used to determine neighbor reachability
`dd`	OSPF Database Description packets used in synchronizing OSPF databases
`request`	OSPF Link State Request packets used in synchronizing OSPF databases
`lsu`	OSPF Link State Update packets used in synchronizing OSPF databases
`ack`	OSPF Link State Ack packets used in synchronizing OSPF databases

monitorauthkey *`key`*

monitorauth none | [simple | md5]*`authkey`*

You can query the OSPF state using the **ospf_monitor** (this should be a hyperlink) utility, which sends nonstandard OSPF packets that generate a text response from OSPF. If you configure an authentication key, the incoming requests must match the specified authentication key. These packets cannot change OSPF state, but the act of querying OSPF can expend system resources. Not authenticated by default.

backbone/area Clause Options and Parameters

backbone | area *`area`*

Configures each OSPF router into at least one OSPF area. If you configure more than one area, at least one must be the backbone. Configure the backbone using the **backbone** keyword only; you cannot specify it as area 0. The backbone interface can be a **virtuallink**.

Further parameters include:

Parameter	**Description**
authtype 0 or **1** or **none** or **simple**	OSPF specifies an authentication scheme per area. Each interface in the area must use this same authentication scheme, although it can use a different authentication key. 0 is the same as **none**; **1** is the same as **simple**.
stub [cost *`cost`*]	A stub area is one in which there are no ASE routes. Use cost to inject a default route into the area with the specified cost.
networks { *`network`* [restrict] ; *`network`* mask *`mask`* [restrict] ; *`network`* masklen *`number`* [restrict] ; host *`host`* [restrict] ; } ;	The **networks** list describes the scope of an area. Intra-area LSAs that fall within the specified ranges are not advertised into other areas as inter-area routes. Instead, the specified ranges are advertised as summary network LSAs. If you specify **restrict**, the summary network LSAs are not advertised. Intra-area LSAs that do not fall into any range are also advertised as summary network LSAs. This option is very useful

Parameter	Description
	on well designed networks in reducing the amount of routing information propagated between areas. The entries in this list are either networks, or a subnetwork/mask pair.
stubhosts {host cost *cost* ;} ; interface *list* [cost *cost*]	The stubhosts list specifies directly attached hosts that should be advertised as reachable from this router, and the costs with which they should be advertised. Specify point-to-point interfaces here on which it is not desirable to run OSPF. It is also useful to assign an additional address to the loopback interface (one not on the 127 network) and advertise it as a stub host. If this address is the same one used as the router ID, it enables routing to OSPF routers by router ID, instead of by interface address. This is more reliable than routing to one of the router's interface addresses, which may not always be reachable.
{*interface-parameters*} ; interface *list* nonbroadcast [cost *cost*]	Use this form of the **interface** clause (with the optional *cost* value, and immediately followed by the **interface-parameters**) to configure a broadcast (which requires IP multicast support) or a point-to-point interface. (See the **interfaces** statement for a description of *list*.) Each interface has a cost. The costs of all the interfaces a packet must cross to reach a destination are summed to get the cost to that destination. The *cost* can be any non-zero value (the default is **1**).

The following are the interface-parameters. You can specify them on any class of interface:

```
enable | disable ;
retransmitinterval time ;
transitdelay time ;
priority value ;
hellointerval time ;
routerdeadinterval time ;
authkey key ;
```

Parameter	Description
retransmitinterval *time*	Number of seconds between link state advertisement retransmissions for adjacencies belonging to this interface.
transitdelay *time*	Estimated number of seconds required to transmit a link state update over this interface. Takes into account transmission and propagation delays and must be greater than 0.
priority *value*	Number between 0 and 255 specifying the priority for becoming the designated router on this interface. When two routers attached to a network both attempt to become designated router, the one with the highest priority prevails. A router whose router priority is 0 is ineligible to become designated router.
hellointerval *time*	Length of time, in seconds, between Hello packets that the router sends on the interface.
routerdeadinterval *time*	Number of seconds not hearing a router's Hello packets before the router's neighbors will declare it down.
authkey *key*	Used by OSPF authentication to generate and verify the authentication field in the OSPF header. You can configure the

Parameter	Description
	authentication key on a per-interface basis. Specify it using one to eight decimal digits separated by periods, a one to eight byte hexadecimal string preceded by 0x, or a one to eight character string in double quotes.

This form of the interface clause (with the nobroadcast option) is for point-to-point interfaces only. By default, OSPF packets to neighbors on point-to-point interfaces are sent using the IP multicast mechanism. GateD detects this condition and falls back to using sending unicast OSPF packets to this point-to-point neighbor.

If you do not want IP multicasting, because the remote neighbor does not support it, specify **nobroadcast** to force the use of unicast OSPF packets. You can also use this option to eliminate warnings when GateD detects the bug mentioned previously. (See the previous page for the **interface-parameters**.)

Use this form of the **interface** clause to specify a nonbroadcast interface on a nonbroadcast multiaccess (NBMA) media. Since an OSPF broadcast media must support IP multicasting, you must configure a broadcast-capable media, such as Ethernet, that does not support IP multicasting as a nonbroadcast interface. A nonbroadcast interface supports any of the standard interface clauses listed previously, plus the following two that are specific to nonbroadcast interfaces:

- **pollinterval time**: Before adjacency is established with a neighbor, OSPF packets are sent periodically at the specified poll interval.
- **routers gateway**: By definition, it is not possible to send broadcast packets to discover OSPF neighbors on a nonbroadcast, so you must configure all neighbors. The list includes one or more neighbors and an indication of their eligibility to become a designated router.

```
virtuallink neighborid routerid transitarea area
{ interface-parameters } ;
```

For backbone only:

Virtual links are used to establish or increase connectivity of the backbone area. The **neighborid** is the router-ID of the other end of the virtual link. The transit area specified must also be configured on this system. You can specify all standard interface parameters defined by the interface clause previously described on a virtual link. (See *`interface-parameters`*.)

static

static — The static statements define the static routes GateD uses. A single static statement can specify any number of routes. These statements must occur after protocol statements and before control statements in `GATED.CONF`. Specify any number of static statements, each containing any number of static route definitions. You can override these routes with ones with better preference values.

Format

```
static
 {   host host  gateway list
  | network [mask mask  | masklen number] gateway list
  | default gateway list
```

```
    [interface list]
    [preference value]
    [retain]
    [reject]
    [blackhole]
    [noinstall]
    ;
  network [mask mask  |  masklen number]
    interface interface
    [preference value]
    [retain]
    [reject]
    [blackhole]
    [noinstall]
    ;
} ;
```

Options and Parameters

`host...gateway` *`list`*

`default gateway` *`list`*

Most general form of the static statement. Defines a static route through one or more gateways. Static routes are installed when one or more of the gateways listed are available on directly attached interfaces. If more than one eligible gateway is available, they are limited by the number of multipath destinations supported.

The second form of the ***`network`*** **mask...** clause farther down in the statement is for primitive support of multiple network addresses on one interface.

`interface` *`list`*

Gateways are valid only when they are on one of these interfaces.

`preference` *`value`*

Preference of this static route. Controls how this route competes with routes from other protocols. The default *`value`* is 60.

`retain`

Prevent specific static routes from being removed. Normally GateD removes all routes except interface routes from the kernel forwarding table during a graceful shutdown. Useful for ensuring that some routing is available when GateD is down.

`reject`

`blackhole`

Not supported in VSI TCP/IP. Install this route as a reject or blackhole route. Instead of forwarding a packet like a normal route, reject routes drop packets and send **unreachable** messages to the packet originators. Not all kernel forwarding engines support reject routes. A blackhole route is like a reject route, except that **unreachable** messages are not supported.

`noinstall`

Do not install the route in the kernel forwarding table when active, but make it still exportable to other protocols. Normally the route with the lowest preference is installed there and is the route exported to other protocols.

import

import — The control statements are: import, export, aggregate, and generate.

Format

`import` [restrict | preference *`value`*]

The **import** statements control importing routes from routing protocols, and installing the routes in GateD's routing database. The format of an **import** statement varies depending on the source protocol. In all cases, you can specify one of two keywords to control how routes compete with other protocols:

- **restrict**: Restrict the routes from the routing table. In some cases this means that the routes are not installed in the routing table. In others, it means that they are installed with a negative preference; this prevents them from becoming active so that they will not be installed in the forwarding table or exported to other protocols.

- **preference value**: Preference value used when comparing this route to other routes from other protocols. The route with the lowest preference available at any given route becomes the active route, is installed in the forwarding table, and can be exported to other protocols. The individual protocols configure the default preferences.

Importing Routes from BGP and EGP

You can control EGP importation by AS. Note that EGP and BGP versions 2 and 3 only support propagating natural networks, so the host and default route filters are meaningless. BGP version 4 supports propagating any destination along with a contiguous network mask.

EGP and BGP both store any routes rejected implicitly by their not being mentioned in a route filter, or explicitly if **restrict** appears in the routing table with a negative preference. A negative preference prevents a route from becoming active, which prevents it from being installed in the forwarding table or exported to other protocols. This removes the need to break and reestablish a session on reconfiguring if changing the importation policy.

The syntax of the **import** statement for importing routes from BGP or EGP is any of the following:

```
import proto  bgp | egp  autonomoussystem ASnumber restrict ;
import proto  bgp | egp  autonomoussystem ASnumber
 [preference value]  {
 route-filter [restrict  |  preference value] ; } ;
import proto bgp aspath ASpathregexp
 origin   any | [igp] [egp] [incomplete]   restrict ;
import proto bgp aspath ASpathregexp
 origin   any | [igp] [egp] [incomplete]
   [preference value] {
   routefilter [restrict | preference value] ; } ;
```

The third and fourth variation of the import statements is for BGP only and supports controlling propagation by using AS path regular expressions. An AS path is a list of ASs that routing

information passes through to get to a router, and an indicator of the origin of the AS path. Use this information to set the preference of one path to a destination network over another. You do this by listing patterns applied to AS paths when importing and exporting routes. Each AS that a route passes through prepends its AS number to the beginning of the AS path.

Aspath Clause

The following **aspath** clause in the **import** statement indicates that an AS matching the **ASpathregexp** with the specified origin is matched. The parameters follow:

aspath *ASpathregexp*origin any | [igp] [egp] [incomplete]

Aspath Clause Regular Expression

ASpathregexp

Regular expression, with the alphabet as the set of AS numbers, consisting of one or more AS path expressions, which are terms and operators. An AS path term (***ASpathterm***) consists of the following:

Expression	Description
ASnumber	Any valid AS system number, from 1 through 65534.
.	Matches any AS number.
(*ASpathregexp*)	Parentheses group sub-expressions. An operator such as asterisk(*) or question mark (?) works on a single element or on a regular expression enclosed in parentheses.

Aspath Clause Operators

AS path operators consists of the following:

Parameter	Description
*ASpathterm***{m}**	Exactly m repetitions, where *m* is a positive integer.
*ASpathterm***{m,}**	*m* or more repetitions, where *m* is a positive integer.
*ASpathterm***{m,n}**	At least *m* and at most *n* repetitions, where *m* and *n* are both nonnegative integers and *m* <= *n*.
*ASpathterm**	Zero or more repetitions (shorthand for **{0,}**).
ASpathterm+	One or more repetitions (shorthand for **{1,}**).
*ASpathterm***?**	Zero or one repetition (shorthand for **{0,1}**).
ASpathterm | *ASpathterm*	Matches either term.

Remaining Import Statement Options

origin any | igp egp incomplete

Details the completeness of AS path information. An origin of **igp** indicates that the route was learned from an interior routing protocol and is most likely complete. An origin of **egp** indicates that the route

was learned from an exterior routing protocol that does not support AS paths (EGP for example), and that the path is most likely not complete. When the path information is definitely not complete, use **incomplete**.

Importing Routes from RIP, HELLO, and Redirects

You can control importing RIP, HELLO, and Redirect routes by any protocol, source interface, or source gateway. If using more than one, they are processed from most general (protocol) to most specific (gateway). RIP and HELLO do not support preferences to choose between routes of the same protocol; they use metrics instead. They also do not save rejected routes since they have short update intervals.

The syntax of the **import** statement for importing routes from RIP, HELLO, or redirects is either of the following:

```
import proto  rip | hello | redirect
 [interface list  |  gateway list]
 restrict ;

import proto  rip | hello | redirect
 [interface list  |  gateway list]
 [preference value]
 {   routefilter [restrict  |  preference value] ; } ;
```

Importing Routes from OSPF

You can only control importing AS External (ASE) routes. OSPF intra- and inter-area routes are always imported into the GateD routing table with a **preference** of **10**. If using an **ospftag**, the import clause only applies to routes with the specified tag.

You can only restrict importing OSPF ASE routes if functioning as an AS border router. Do this by specifying an **export ospfase** clause. Specifying an empty export clause can restrict importing ASEs, when no ASEs are exported.

Like the other interior protocols, you cannot use **preference** to choose between OSPF ASE routes; OSPF costs accomplish this. Routes rejected by policy go into the table with a negative preference.

The syntax of the **import** statement for importing routes from OSPF is either of the following:

```
import proto ospfase [tag ospftag] restrict ;

import proto ospfase [tag ospftag]
  [preference value]
  { routefilter [restrict  | preference value] ; } ;

export
```

The control statements are:

- **import**
- **export**
- **aggregate**
- **generate**

Format

`export` [restrict | metric ***`metric`***]

The **export** statement controls which routes GateD advertises to other systems. Like **import**, the **export** syntax varies slightly for each protocol. Both syntaxes are similar and the meanings of many of the parameters are the same. The main difference is that while source information controls importing routes, both destination and source information control exporting routes.

The outer portion of a given **export** statement specifies the destination of the routing information you control. The middle portion restricts the sources. The innermost portion is a route filter used to select individual routes.

One thing that applies in all cases is the specification of a metric. All protocols define a default metric for routes exported. In most cases, this can be overridden at several levels of the export statement. The most specific specification of a metric is the one applied to the route exported. The values you can specify for a metric depend on the destination protocol the **export** statement references:

- **restrict**: Do not export anything. If specified on the destination portion of the export statement, it means not to export anything to this destination. If specified on the source portion, it means not to export anything from this source. If specified as part of a route filter, it means not to export the routes matching that filter.
- **metric *`metric`***: Metric used when exporting to the specified destination.

Exporting to EGP and BGP

The AS controls exporting to EGP and BGP, the same policy applied to all routers in the AS. EGP metrics range from 0 through 255, with 0 the most attractive. BGP metrics are 16-bit unsigned quantities (that range from 0 through 65535, inclusive with 0 the most attractive). While BGP version 4 actually supports 32-bit unsigned quantities, GateD does not yet support this.

If you do not specify an export policy, only routes to attached interfaces are exported. If you specify any policy, the defaults are overridden; you should explicitly specify everything you want exported. (Note that EGP and BGP versions 2 and 3 only support the propagation of natural networks, so the host and default route filters are meaningless. BGP version 4 supports the propagation of any destination along with a contiguous network mask.)

The syntax of the **export** statement for exporting routes to EGP or BGP is either of the following:

```
export proto  bgp | egp  as ASnumber  restrict ;
export proto  bgp | egp  as ASnumber  [metric metric]
 {  exportlist ; } ;
```

Exporting to RIP and HELLO

Any protocol, interface, or gateway can control exporting to RIP and HELLO. If you specify more than one, they are processed from most general (protocol) to most specific (gateway). It is not possible to set metrics for exporting RIP routes into RIP, or exporting HELLO routes into HELLO. Attempts to do this are silently ignored.

If you do not specify an export policy, RIP and interface routes are exported into RIP and HELLO, and interface routes are exported into HELLO. If you specify any policy, the defaults are overridden; it is necessary to explicitly specify everything that should be exported.

RIP version 1 and HELLO assume that all subnets of the shared network have the same subnet mask, so they are only able to propagate subnets of that network. RIP version 2 is capable of propagating all routes, when not sending version 1 compatible updates.

To announce routes that specify a next hop of the loopback interface (static and internally generated default routes) over RIP or HELLO, specify the metric at some level in the export clause. Just setting a default metric is not sufficient. This is a safeguard to verify that the announcement is intended.

The syntax of the **export** statement for exporting routes to RIP or HELLO is either of the following:

```
export proto  rip | hello
 [interface list  |  gateway list] restrict ;

export proto  rip | hello
 [interface list  |  gateway list]  [metric metric]
 { exportlist ; } ;
```

Exporting to OSPF

It is not possible to create OSPF intra- or inter-area routes by exporting routes from the GateD routing table into OSPF. It is only possible to export from the GateD routing table into OSPF ASE routes. It is also not possible to control the propagation of OSPF routes within the OSPF protocol.

There are two types of OSPF ASE routes, type 1 and type 2 (see the OSPF protocol configuration for details on the two types). Specify the default type using the **defaults** subclause of the **ospf** clause. You can override this with the **export** statement.

OSPF ASE routes also have the provision to carry a tag. This is an arbitrary 32-bit number you can use on OSPF routers to filter routing information. (See the OSPF protocol configuration for details on OSPF tags.) You can override the default tag specified by the **ospf defaults** clause with a tag specified on the **export** statement.

The syntax of the **export** statement for exporting routes to OSPF is either of the following:

```
export proto osfpase [type 1 | 2] [tag ospf-tag]  restrict ;

export proto osfpase [type 1 | 2] [tag ospf-tag]
 [metric metric]
 { exportlist ; } ;
```

Exporting BGP and EGP Routes

You can specify BGP and EGP routes by source AS. You can export all routes by AS path. The syntax of the **proto** statement for exporting BGP or EGP routes is either of the following:

```
proto  bgp | egp  autonomoussystem ASnumber  restrict ;
proto  bgp | egp  autonomoussystem ASnumber  [metric metric]
 {  routefilter [restrict  |  metric metric] ; } ;
```

Exporting RIP and HELLO Routes

You can export RIP and HELLO routes by protocol, source interface, or source gateway. The syntax of the **proto** statement for exporting RIP or HELLO routes is either of the following:

```
proto  rip | hello
```

```
 [interface list  |  gateway list]  restrict ;
proto  rip | hello
 [interface list  |  gateway list]  [metric metric]
{  routefilter [restrict  |  metric metric] ; } ;
```

Exporting OSPF Routes

You can export both OSPF and OSPF ASE routes into other protocols. The syntax of the **proto** statement for exporting OSPF routes is either of the following:

```
proto  ospfase | ospfase  restrict ;
proto  ospfase | ospfase [metric metric]
 { routefilter [restrict  |  metric metric] ; } ;
```

Exporting Routes from Nonrouting Protocols with Interface

If you want GateD to export direct or static routes, or routes learned from the kernel, use the protocol statement or interface statement along with the interface of the next hop in the GateD configuration file. The syntax of the **proto** statement for exporting routes from nonrouting protocols with an interface is either of the following:

```
proto  direct | static | kernel
 [interface list] restrict ;

proto  direct | static | kernel
 [interface list]  [metric metric]
 { routefilter [restrict  |  metric metric] ; } ;
```

The proto statement parameters include:

- **direct**: Routes to directly attached interfaces.
- **static**: Static routes specified in a static clause.
- **kernel**: On systems with the routing socket, routes learned from the routing socket are installed in the GateD routing table with a protocol of kernel. You can export these routes by referencing this protocol. This is useful when it is desirable to have a script install routes with the route command and propagate them to other routing protocols.

Exporting Routes from Nonrouting Protocols by Protocol

If you want GateD to export default or aggregate routes, use the protocol statement in the GateD configuration file. The syntax of the **proto** statement for exporting routes from nonrouting protocols by protocol is either of the following:

```
proto default | aggregate  restrict ;
proto default | aggregate
 [metric metric]
 { routefilter [restrict  |  metric metric] ; } ;
```

The proto statement parameters include:

- **default**: Routes created by the gendefault option. Use route generation instead.
- **aggregate**: Routes synthesized from other routes when using the aggregate and generate statements.

Exporting by AS Path

When configuring BGP, all routes get an AS path when added to the routing table. For all interior routes, this AS path specifies IGP as the origin and no ASEs in the AS path (the current AS is added when the route is exported). For EGP routes, this AS path specifies EGP as the origin and the source AS as the AS path. For BGP routes, the AS path is stored as learned from BGP. (The AS path regular expression syntax appears in the .)

The syntax of the **proto** statement for exporting by AS path is either of the following:

```
proto  proto | all  aspath ASpathregexp
 origin  any | [igp] [egp] [incomplete]  restrict ;
proto  proto | all  aspath ASpathregexp
 origin  any | [igp] [egp] [incomplete]  [metric metric]
  { routefilter [restrict  |  metric metric] ; } ;
```

Exporting by Route Tag

Both OSPF and RIP version 2 currently support tags. All other protocols always have a tag of zero. You can select the source of exported routes based on this tag. This is useful when classifying routes by tag when exporting them into a given routing protocol. The syntax of the **proto** statement for exporting by route tag is either of the following:

```
proto  proto | all  all tag tag restrict ;
proto  proto | all  all tag tag
 [metric metric]
 { routefilter [restrict  |  metric metric] ; } ;
```

aggregate

aggregate — Use route aggregation to generate a more general route from a specific one. Use it, for example, at an AS border to generate a route to a network to be advertised through EGP, given the presence of one or more subnets of that network learned through RIP. Regional and national networks also use route aggregation to reduce routing information. By carefully allocating network addresses to clients, regional networks can just announce one route to regional networks instead of hundreds. No aggregation occurs unless explicitly requested in an aggregate statement.

Description

The control statements are:

- **import**
- **export**
- **aggregate**
- **generate**

Aggregate routes are not actually used for packet forwarding by the originator of the aggregate route, only by the receiver (if it wishes). A router, receiving a packet that does not match one of the component routes that led to the generation of an aggregate route, is supposed to respond with an ICMP **network unreachable** message. This prevents packets for unknown component routes from following a default route into another network where they would be continuously forwarded back to

the border router, until their TTL expires. Sending an unreachable message for a missing piece of an aggregate is only possible on systems that support reject routes, which VSI TCP/IP does not.

Format

```
aggregate  default | network [mask mask  |  masklen number]
 [preference value] [brief]
 { proto [all | direct | static | kernel | aggregate | proto]
  [as AS  |  tag tag  |  aspath ASpathregexp] restrict ;
 proto [all | direct | static | kernel | aggregate | proto]
  [as AS  |  tag tag  |  aspath ASpathregexp]  [preference value]
  { routefilter [restrict  |  preference value] ; } ;
 } ;
```

Options and Parameters

`preference` *`value`*

The default preference *`value`* is **130**.

`brief`

Truncate the AS path to the longest common AS path. The default is to build an AS path consisting of SETs and SEQUENCEs of all contributing AS paths.

`proto` *`proto`*

In addition to the special protocols listed, you can select the contributing protocol from among those currently configured in GateD.

`as` *`AS`*

Restrict selection of routes to those learned from the specified AS.

`tag` *`tag`*

Restrict selection of routes to those with the specified tag.

`aspath` *`ASpathregexp`*

Restrict selection of routes to those that match the specified AS path.

`restrict`

Restrict certain routes from contributing to the specified aggregate.

A route can only contribute to an aggregate route that is more general than itself; it must match the aggregate under its mask. Any given route can only contribute to one aggregate route, which will be the most specific configured, but an aggregate route can contribute to a more general aggregate.

generate

generate — A slight variation on aggregation is generating a route based on certain conditions. This is sometimes known as the "route of last resort." This route inherits the next hops and AS path from the contributor specified with the lowest (most favorable) preference. The most common usage is

to generate a default based on the presence of a route from a peer on a neighboring backbone. The control statements are: **import**, **export**, **aggregate**, and **generate**

Format

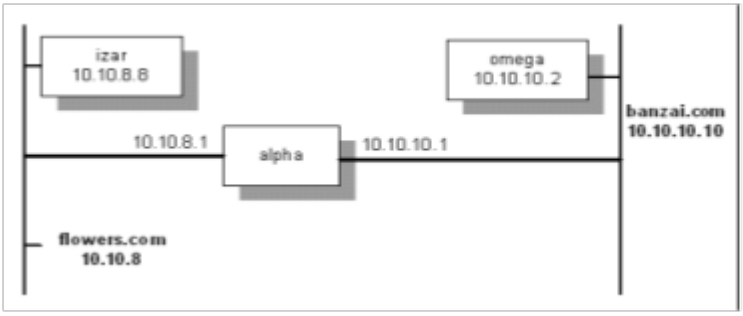
```
generate  default | network [mask mask  |  masklen number]
 [preference value] [brief]
   { [as AS  |  tag tag  |  aspath ASpathregexp]
   restrict ;
 proto [all | direct | static | kernel | aggregate | proto]
   [as AS  |  tag tag  |  aspath ASpathregexp]
   [preference value]
   { routefilter [restrict  |  preference value] ; } ;
 } ;
```

B.6. Sample GateD Configurations

Figure B.1 shows two networks connected within an AS using RIP.

Figure B.1, Example B.1, and Example B.3 show the RIP statements on each end host and gateway alpha, which has IP forwarding enabled. All systems are running GateD.

Figure B.1. Sample RIP Configuration



Example B.1. GateD Configuration File for izar

```
#  turn on RIP and listen for updates.
#
rip on;
```

Example B.2. GateD Configuration File for alpha

```
#  turn on RIP.
#
rip yes;
#
#  use RIP to pass routing information to the banzai network.
#
export proto rip interface 10.10.10.1
 {
 #  we know about the flowers network, so announce it.
 #
 proto direct {
   10.10.8.0 mask 255.255.255.0;
   };
```

```
 #  use RIP to announce all routes learned from flowers.
 #
 proto rip interface 10.10.8.0 {
    all;
    };
  };
```

Example B.3. GateD Configuration File for omega

```
#  turn on RIP and listen for updates.
#
rip on;
```

Example B.4 shows a sample RIP statement where the gateway announces a default route to the backbone, and announces all of the individual subnet routes to the outside world.

Example B.4. Default RIP Announcements

```
#  enable RIP:
#
rip yes;

#  using RIP, announce all local subnets via interface 192.168.12.3:
#
export proto rip interface 192.168.12.3 metric 3
 {
    proto rip interface 192.168.1.5
    {
     all;
    };
  };
#
#  Using RIP, announce default via interface 192.168.1.5:
#
export proto rip interface 192.168.3.1
  {
  proto rip interface 192.168.1.5
   {
   default;
   };
   };
```

Example B.5 shows a configuration for AS 283 that enables RIP and OSPF, which you can use to test both.

Example B.5. Using RIP and OSPF

```
# this interface is passive:
#
interfaces {
  interface SVA-0 passive;
  };
#
#  this Autonomous System number is 283:
#
autonomoussystem 283;
#
#  turn on RIP:
```

```
# packets are to be broadcast.
# metric for routes learned via other protocols is 5.
# multicast RIP V2 packets on SVA-0.
#
rip yes {
  broadcast;

     defaultmetric 5;
     interface SVA-0 version 2 multicast;
     };
#
# turn on OSPF:
# Trace Link State Advertisement creation and
# Shortest Path First calculations
# use authentication key "ZZZZZZZZ" when handling OSPF queries.
# this system is on the backbone.
#  use simple password authentication for this area.
#  make this system very unlikely to be a designated router.
#  set the OSPF header authentication key to "YYYYYYYY" for
#  packets going out on SVA-0.
#
ospf yes {
  traceoptions lsabuild spf;
  monauthkey "ZZZZZZZZ";
  backbone {
   authtype simple;
   interface all {
    priority 2;
    };
   interface SVA-0 {
    authkey "YYYYYYYY";
    };
   };
  };
```

Example B.6 shows a configuration for a static route.

Example B.6. Static Routes

```
#
# in this example our host's address is 192.168.1.42
#
static {
   192.168.2.0 masklen 24 interface 192.168.1.42 retain;
  default gateway 192.168.1.1 ;
     ;
```

Appendix C. Trademark and Copyright Notifications

This appendix contains a complete listing of trademarks and copyright notification contained in this manual.

The material in this document is for informational purposes only and is subject to change without notice. It should not be construed as a commitment by VMS Software, inc. VMS Software, inc. assumes no responsibility for any errors that may appear in this document.

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

The following third-party software may be included with your product and will be subject to the software license agreement.

Network Time Protocol (NTP). Copyright © 1992-2004 by David L. Mills. The University of Delaware makes no representations about the suitability of this software for any purpose.

Point-to-Point Protocol. Copyright © 1989 by Carnegie-Mellon University. All rights reserved. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by Carnegie Mellon University. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED ``AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

RES_RANDOM.C. Copyright © 1997 by Niels Provos <provos@physnet.uni-hamburg.de> All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by Niels Provos.

4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

Copyright © 1990 by John Robert LoVerso. All rights reserved. Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated

in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by John Robert LoVerso.

Kerberos. Copyright © 1989, DES.C and PCBC_ENCRYPT.C Copyright © 1985, 1986, 1987, 1988 by Massachusetts Institute of Technology. Export of this software from the United States of America is assumed to require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting. WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

DNSSIGNER (from BIND distribution) Portions Copyright (c) 1995-1998 by Trusted Information Systems, Inc.

Portions Copyright (c) 1998-1999 Network Associates, Inc.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS" AND TRUSTED INFORMATION SYSTEMS DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL TRUSTED INFORMATION SYSTEMS BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

ERRWARN.C. Copyright © 1995 by RadioMail Corporation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of RadioMail Corporation, the Internet Software Consortium nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY RADIOMAIL CORPORATION, THE INTERNET SOFTWARE CONSORTIUM AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL RADIOMAIL CORPORATION OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,

OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software was written for RadioMail Corporation by Ted Lemon under a contract with Vixie Enterprises. Further modifications have been made for the Internet Software Consortium under a contract with Vixie Laboratories.

IMAP4R1.C, MISC.C, RFC822.C, SMTP.C Original version Copyright © 1988 by The Leland Stanford Junior University

ACCPORNAM technology Copyright (c) 1999 by Brian Schenkenberger - TMESIS SOFTWARE

NS_PARSER.C Copyright © 1984, 1989, 1990 by Bob Corbett and Richard Stallman

This program is free software. You can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 1, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139 USA

IF_ACP.C Copyright © 1985 and IF_DDA.C Copyright © 1986 by Advanced Computer Communications

IF_PPP.C Copyright © 1993 by Drew D. Perkins

ASCII_ADDR.C Copyright © 1994 Bell Communications Research, Inc. (Bellcore)

DEBUG.C Copyright © 1998 by Lou Bergandi. All Rights Reserved.

NTP_FILEGEN.C Copyright © 1992 by Rainer Pruy Friedrich-Alexander Universitaet Erlangen-Nuernberg

RANNY.C Copyright © 1988 by Rayan S. Zachariassen. All Rights Reserved.

MD5.C Copyright © 1990 by RSA Data Security, Inc. All Rights Reserved.

Portions Copyright © 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989 by SRI International

Portions Copyright © 1984, 1989 by Free Software Foundation

Portions Copyright © 1993, 1994, 1995, 1996, 1997, 1998 by the University of Washington. Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notices appear in all copies and that both the above copyright notices and this permission notice appear in supporting documentation, and that the name of the University of Washington or The Leland Stanford Junior University not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. This software is made available "as is", and THE UNIVERSITY OF WASHINGTON AND THE LELAND STANFORD JUNIOR UNIVERSITY DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, WITH REGARD TO THIS SOFTWARE, INCLUDING WITHOUT LIMITATION ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND IN NO EVENT SHALL THE UNIVERSITY OF WASHINGTON OR THE LELAND STANFORD JUNIOR UNIVERSITY BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER

RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, TORT (INCLUDING NEGLIGENCE) OR STRICT LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

   THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies, and that the name of Hewlett-Packard Corporation not be used in advertising or publicity pertaining to distribution of the document or software without specific, written prior permission. THE SOFTWARE IS PROVIDED "AS IS" AND HEWLETT-PACKARD CORP. DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL HEWLETT-PACKARD CORPORATION BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

International Business Machines, Inc. (hereinafter called IBM) grants permission under its copyrights to use, copy, modify, and distribute this Software with or without fee, provided that the above copyright notice and all paragraphs of this notice appear in all copies, and that the name of IBM not be used in connection with the marketing of any product incorporating the Software or modifications thereof, without specific, written prior permission. To the extent it has a right to do so, IBM grants an immunity from suit under its patents, if any, for the use, sale or manufacture of products to the extent that such products are used for performing Domain Name System dynamic updates in TCP/IP networks by means of the Software. No immunity is granted for any product per se or for any other function of any product. THE SOFTWARE IS PROVIDED "AS IS", AND IBM DISCLAIMS ALL WARRANTIES, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL IBM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE, EVEN IF IBM IS APPRISED OF THE POSSIBILITY OF SUCH DAMAGES.

Portions Copyright © 1995, 1996, 1997, 1998, 1999, 2000 by Internet Software Consortium. All Rights Reserved. Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright (c) 1996-2000 Internet Software Consortium.

Use is subject to license terms which appear in the file named ISC-LICENSE that should have accompanied this file when you received it. If a file named ISC-LICENSE did not accompany this file, or you are not sure the one you have is correct, you may obtain an applicable copy of the license at: http://www.isc.org/isc-license-1.0.html.

This file is part of the ISC DHCP distribution. The documentation associated with this file is listed in the file DOCUMENTATION, included in the top-level directory of this release. Support and other services are available for ISC products - see http://www.isc.org for more information.

ISC LICENSE, Version 1.0

1. This license covers any file containing a statement following its copyright message indicating that it is covered by this license. It also covers any text or binary file, executable, electronic or printed image that is derived from a file that is covered by this license, or is a modified version of a file covered by this license, whether such works exist now or in the future. Hereafter, such works will be referred to as "works covered by this license," or "covered works."

2. Each source file covered by this license contains a sequence of text starting with the copyright message and ending with "Support and other services are available for ISC products - see http://www.isc.org for more information." This will hereafter be referred to as the file's Bootstrap License.

3. If you take significant portions of any source file covered by this license and include those portions in some other file, then you must also copy the Bootstrap License into that other file, and

that file becomes a covered file. You may make a good-faith judgement as to where in this file the bootstrap license should appear.

4. The acronym "ISC", when used in this license or generally in the context of works covered by this license, is an abbreviation for the words "Internet Software Consortium."

5. A distribution, as referred to hereafter, is any file, collection of printed text, CD ROM, boxed set, or other collection, physical or electronic, which can be distributed as a single object and which contains one or more works covered by this license.

6. You may make distributions containing covered files and provide copies of such distributions to whomever you choose, with or without charge, as long as you obey the other terms of this license. Except as stated in (9), you may include as many or as few covered files as you choose in such distributions.

7. When making copies of covered works to distribute to others, you must not remove or alter the Bootstrap License. You may not place your own copyright message, license, or similar statements in the file prior to the original copyright message or anywhere within the Bootstrap License. Object files and executable files are exempt from the restrictions specified in this clause.

8. If the version of a covered source file as you received it, when compiled, would normally produce executable code that would print a copyright message followed by a message referring to an ISC web page or other ISC documentation, you may not modify the file in such a way that, when compiled, it no longer produces executable code to print such a message.

9. Any source file covered by this license will specify within the Bootstrap License the name of the ISC distribution from which it came, as well as a list of associated documentation files. The associated documentation for a binary file is the same as the associated documentation for the source file or files from which it was derived. Associated documentation files contain human-readable documentation which the ISC intends to accompany any distribution.

   If you produce a distribution, then for every covered file in that distribution, you must include all of the associated documentation files for that file. You need only include one copy of each such documentation file in such distributions.

   Absence of required documentation files from a distribution you receive or absence of the list of documentation files from a source file covered by this license does not excuse you from this requirement. If the distribution you receive does not contain these files, you must obtain them from the ISC and include them in any redistribution of any work covered by this license. For information on how to obtain required documentation not included with your distribution, see: http://www.isc.org/getting-documentation.html.

   If the list of documentation files was removed from your copy of a covered work, you must obtain such a list from the ISC. The web page at http://www.isc.org/getting-documentation.html contains pointers to lists of files for each ISC distribution covered by this license.

   It is permissible in a source or binary distribution containing covered works to include reformatted versions of the documentation files. It is also permissible to add to or modify the documentation files, as long as the formatting is similar in legibility, readability, font, and font size to other documentation in the derived product, as long as any sections labeled CONTRIBUTIONS in these files are unchanged except with respect to formatting, as long as the order in which the CONTRIBUTIONS section appears in these files is not changed, and as long as the manual page which describes how to contribute to the Internet Software Consortium (hereafter referred to as the Contributions Manual Page) is unchanged except with respect to formatting.

Documentation that has been translated into another natural language may be included in place of or in addition to the required documentation, so long as the CONTRIBUTIONS section and the Contributions Manual Page are either left in their original language or translated into the new language with such care and diligence as is required to preserve the original meaning.

10. You must include this license with any distribution that you make, in such a way that it is clearly associated with such covered works as are present in that distribution. In any electronic distribution, the license must be in a file called "ISC-LICENSE".

    If you make a distribution that contains works from more than one ISC distribution, you may either include a copy of the ISC-LICENSE file that accompanied each such ISC distribution in such a way that works covered by each license are all clearly grouped with that license, or you may include the single copy of the ISC-LICENSE that has the highest version number of all the ISC-LICENSE files included with such distributions, in which case all covered works will be covered by that single license file. The version number of a license appears at the top of the file containing the text of that license, or if in printed form, at the top of the first page of that license.

11. If the list of associated documentation is in a separated file, you must include that file with any distribution you make, in such a way that the relationship between that file and the files that refer to it is clear. It is not permissible to merge such files in the event that you make a distribution including files from more than one ISC distribution, unless all the Bootstrap Licenses refer to files for their lists of associated documentation, and those references all list the same filename.

12. If a distribution that includes covered works includes a mechanism for automatically installing covered works, following that installation process must not cause the person following that process to violate this license, knowingly or unknowingly. In the event that the producer of a distribution containing covered files accidentally or wilfully violates this clause, persons other than the producer of such a distribution shall not be held liable for such violations, but are not otherwise excused from any requirement of this license.

13. COVERED WORKS ARE PROVIDED "AS IS". ISC DISCLAIMS ALL WARRANTIES WITH REGARD TO COVERED WORKS INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

14. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OF COVERED WORKS.

Use of covered works under different terms is prohibited unless you have first obtained a license from ISC granting use pursuant to different terms. Such terms may be negotiated by contacting ISC as follows:

Internet Software Consortium

950 Charter Street

Redwood City, CA 94063

Tel: 1-888-868-1001 (toll free in U.S.)

Tel: 1-650-779-7091

Fax: 1-650-779-7055

Email: info@isc.org

Email: licensing@isc.org

DNSSAFE LICENSE TERMS

This BIND software includes the DNSsafe software from RSA Data Security, Inc., which is copyrighted software that can only be distributed under the terms of this license agreement.

The DNSsafe software cannot be used or distributed separately from the BIND software. You only have the right to use it or distribute it as a bundled, integrated product.

The DNSsafe software can ONLY be used to provide authentication for resource records in the Domain Name System, as specified in RFC 2065 and successors. You cannot modify the BIND software to use the DNSsafe software for other purposes, or to make its cryptographic functions available to end-users for other uses.

If you modify the DNSsafe software itself, you cannot modify its documented API, and you must grant RSA Data Security the right to use, modify, and distribute your modifications, including the right to use any patents or other intellectual property that your modifications depend upon.

You must not remove, alter, or destroy any of RSA's copyright notices or license information. When distributing the software to the Federal Government, it must be licensed to them as "commercial computer software" protected under 48 CFR 12.212 of the FAR, or 48 CFR 227.7202.1 of the DFARS.

You must not violate United States export control laws by distributing the DNSsafe software or information about it, when such distribution is prohibited by law.

THE DNSSAFE SOFTWARE IS PROVIDED "AS IS" WITHOUT ANY WARRANTY WHATSOEVER. RSA HAS NO OBLIGATION TO SUPPORT, CORRECT, UPDATE OR MAINTAIN THE RSA SOFTWARE. RSA DISCLAIMS ALL WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO ANY MATTER WHATSOEVER, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS.

If you desire to use DNSsafe in ways that these terms do not permit, please contact:

RSA Data Security, Inc.

100 Marine Parkway

Redwood City, California 94065, USA