MS81: MQSeries® internet pass-thru
Version 1.0

Document:	MS81 SCRIPT
Issued:	June 2000
Revision Date:	June 2000
Previous Revision Date:	March 2000
Next Review:	As required

MQSeries Development
IBM United Kingdom Laboratories
Hursley Park
Winchester, SO21 2JN
England

Take Note!

Before using this manual and the product it supports, be sure to read the general information under "Notices" on page 5.

First Edition, June 2000

This edition applies to Version 1.0 of MS81: MQSeries internet pass-thru (program number 5639-L92) and to all subsequent releases and modifications until otherwise indicated in new editions.

A form for reader's comments is provided at the back of this publication.

Contents

Figures

Notices

The following paragraph does not apply in any country where such provisions are inconsistent with local law.

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe any of the intellectual property rights may be used instead of the IBM product. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, USA.

The information contained in this document has not be submitted to any formal IBM test and is distributed AS IS. The use of the information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

FFST	First Failure Support Technology	IBM
IBMLink	MQSeries	SupportPac

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

Other company, product, or service names may be the trademarks or service marks of others.

Preface

What is internet pass-thru?

IBM® MQSeries internet pass-thru:

- Is an MQSeries base product extension that can be used to implement messaging solutions between remote sites across the Internet.
- Makes the passage of MQSeries channel protocols into and out of a firewall simpler and more manageable by tunneling the protocols inside HTTP or acting as a proxy.
- Operates as a standalone service that can receive and forward MQSeries message flows. The system on which it runs does not have to host an MQSeries Queue Manager.
- Helps you to provide business-to-business transactions using MQSeries.
- Enables existing, unchanged MQSeries applications to be used through a firewall
- Provides a single point of control over access to multiple queue managers.

In this book, MQSeries internet pass-thru is often termed “MQIPT” for convenience.

Who this book is for

This book is for systems designers, technical MQSeries administrators, and firewall and network administrators.

What you need to know to understand this document

You need a good understanding of:

- The administration of MQSeries queue managers and message channels, as described in *MQSeries System Administration* and *MQSeries Intercommunication*
- The way that firewalls are implemented
- Internet protocol routing/networking

Prerequisites

This release of internet pass-thru requires either Windows NT® 4.0, with Service pack 3, or Sun Solaris 2.7, with a JDK at level 1.2.2, or later compatible release, to benefit from the improved performance of more recent JDK releases. Earlier releases of JDK, back to 1.1.7, also work with internet pass-thru.

The only supported network protocol is TCP/IP.

Bibliography

You will also find the following publications useful:

- *MQSeries System Administration*, SC33-1872
- *MQSeries Intercommunication*, SC33-1873
- *MQSeries Clients*, GC33-1632

Chapter 1. Introduction to MQSeries internet pass-thru

MQSeries internet pass-thru is an extension to the base MQSeries product. MQIPT runs as a standalone service that can receive and forward MQSeries message flows, either between two MQSeries queue managers or between an MQSeries client and an MQSeries queue manager. MQIPT enables this connection when the client and server are not on the same physical network.

One or more MQIPTs can be placed in the communication path between two MQSeries queue managers, or between an MQSeries client and an MQSeries queue manager. The MQIPTs allow the two MQSeries systems to exchange messages without needing a direct TCP/IP connection between the two systems. This is useful if the firewall configuration prohibits a direct TCP/IP connection between the two systems.

MQIPT listens on one or more TCP/IP ports for incoming connections, which can carry either the normal MQSeries messages or an MQSeries protocol tunneled inside HTTP. It can handle multiple concurrent connections.

The MQSeries channel that makes the initial TCP/IP connection request is referred to as the “caller”, the channel to which it is attempting to connect as the “responder”, and the queue manager that it is ultimately trying to contact as the “destination queue manager”.

The anticipated uses of MQIPT are:

- MQIPT can be used as a channel concentrator, so that channels from or to multiple separate hosts can appear to a firewall as if they are all from or to the MQIPT host. This makes it easier to define and manage firewall filtering rules.

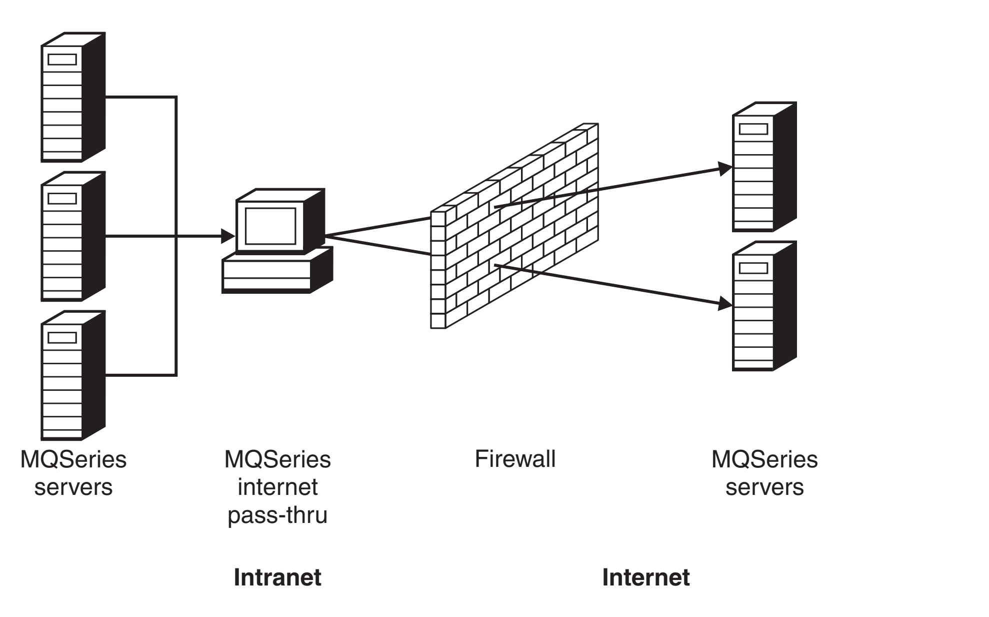


Figure 1. Example of MQIPT as a channel concentrator

- If it is placed in the firewall’s “demilitarized zone” (DMZ), on a machine with a known and trusted internet protocol (IP) address, MQIPT can be used to listen for incoming MQSeries channel connections which it can then forward to the trusted intranet; the inner firewall must allow this trusted machine to make inbound connections. In this configuration, MQIPT prevents external requests for access from seeing the true IP addresses of the machines in the trusted intranet. Thus, MQIPT provides a single point of access.

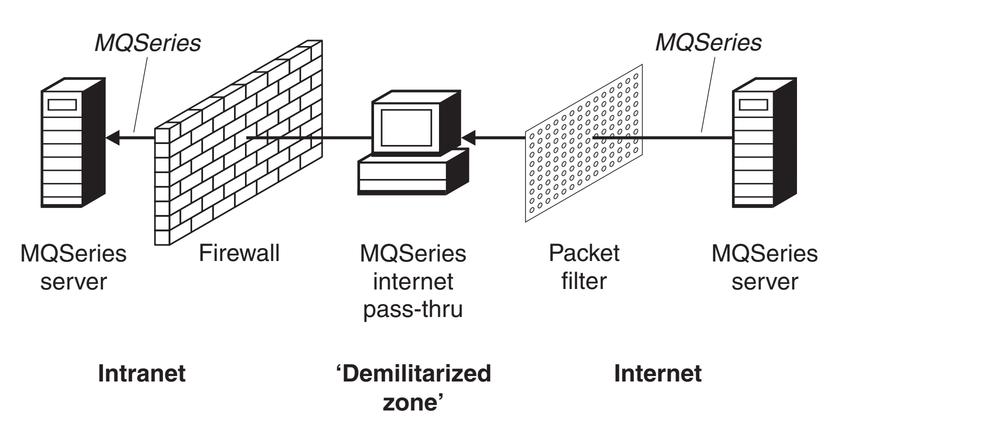


Figure 2. Example of MQIPT with a "demilitarized zone"

- If two MQIPTs are deployed inline, they can communicate using HTTP. Provision of this HTTP tunneling feature enables requests to be transmitted through corporate firewalls, by the use of existing HTTP proxies. The first MQIPT inserts the MQSeries protocol into HTTP and the second extracts the MQSeries protocol from its HTTP wrapper and forwards it to the destination queue manager.

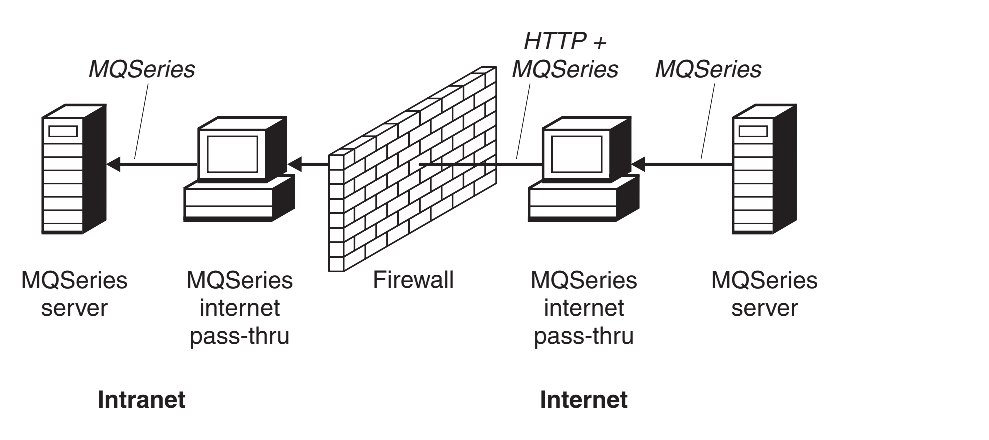


Figure 3. Example of MQIPT and HTTP tunneling

MQIPT provides a number of options for inbound connections through firewalls; it does not provide for outbound connections through firewalls. For outbound connections, it is assumed that the MQSeries client or queue manager is running on a system that is SOCKS-enabled and is configured to communicate with a SOCKS gateway.

MQIPT holds data in memory as it forwards it from its source to its destination. No data is saved on disk (except for memory paged to disk by the operating system). The only time MQIPT accesses the disk explicitly is to read its configuration file and to write log and trace records.

The full range of MQSeries channel types can be made through one or more MQIPTs. The presence of MQIPTs in a communication path has no effect on the functional characteristics of the connected MQSeries components, but there will be some impact on the performance of message transfer.

MQIPT can be used in conjunction with MQSeries publish/subscribe or the MQSeries Integrator message broker.

Chapter 2. How internet pass-thru works

This chapter gives an overview of the way internet pass-thru works, and then describes the following items in more detail:

- "HTTP support" on page 13
- "Supported channel configurations" on page 13
- "Normal termination and failure conditions" on page 14
- "Security considerations" on page 14

Overview of how internet pass-thru works

In its simplest configuration, MQIPT acts as an MQSeries protocol forwarder. It listens on a TCP/IP port (1414 by default) and accepts connection requests from MQSeries channels. If a well-formed request is received, MQIPT establishes a further TCP/IP connection between itself and the destination MQSeries queue manager. It then passes all protocol packets it receives from its incoming connection on to the destination queue manager, and it returns protocol packets from the destination queue manager back on the original incoming connection.

No change to the MQSeries protocol (client/server or queue manager to queue manager) is involved - because neither end is directly aware of the presence of the intermediary - so new versions of the MQSeries client or server code are not required.

To use MQIPT, the caller channel must be configured to use MQIPT's hostname and port, not the hostname and port of the destination queue manager. MQIPT does not examine the channel name; this is simply passed through to the receiving queue manager. Other configuration fields, such as the userid and password in a client/server channel, are similarly passed to the receiving queue manager.

MQIPT can be used to allow access to one or more destination queue managers. For this to work, there must be a mechanism to tell MQIPT which queue manager to connect to, so MQIPT uses the incoming TCP/IP port number to determine which queue manager to connect to, as described in the next paragraph.

To allow access to more than one destination queue manager, MQIPT can be configured to listen on multiple TCP/IP ports. Each incoming port is mapped to a destination queue manager through an MQIPT "route". The MQIPT administrator may define up to 100 such routes, which associate an incoming TCP/IP port with the hostname and port of the destination queue manager. This means that the hostname (IP address) of the destination queue manager is never visible to the originating channel. Each route can handle multiple connections between its listening port and destination, and acts as a different MQIPT "instance".

HTTP support

As an option, MQIPT can be configured so that the data packets it forwards are encoded as HTTP requests. MQIPT supports HTTP tunneling with or without chunking.

Because today's MQSeries channels do not accept HTTP requests, a second MQIPT is required to receive the HTTP requests and convert them back into normal MQSeries protocol packets. The second MQIPT strips off the HTTP header to convert the incoming packet back into a standard MQSeries protocol packet, before passing it on to the destination queue manager.

When using HTTP tunneling without chunking, an HTTP reply is sent back to the first MQIPT. This reply can be the response from the destination queue manager or a dummy acknowledgement.

If either MQSeries system has to send a chain of successive MQSeries protocol packets (as happens when transferring a large message), several HTTP request/reply pairs are used to transfer the data. To achieve this, MQIPTs insert additional request or reply flows.

When using HTTP tunneling with chunking, only the first packet is wrapped in an HTTP header. Middle and last packets have chunking headers. This arrangement removes the wait for a dummy acknowledgement from the second MQIPT, and thus offers slightly better performance than that provided by HTTP tunneling without chunking.

When HTTP is being used between two MQIPTs, the TCP/IP connection on which the HTTP requests and replies are flowed is kept open for the lifetime of the message channel. The MQIPTs do not close the TCP/IP connection between request/reply pairs.

If two MQIPTs are communicating through HTTP, it is possible that an HTTP request might stay outstanding for an extended period. An example is in a requester/server channel, when the server side is waiting for new messages to arrive on its transmission queue. The MQSeries channel protocol provides a “heartbeat” mechanism, which requires the waiting end periodically to send heartbeat messages to its partner (the default channel heartbeat period is 5 minutes) and MQIPT uses this heartbeat as the HTTP reply. Do not disable this channel heartbeat, or set it to an excessively high value, to avoid causing problems with timeouts in some firewalls.

Supported channel configurations

All MQSeries channel types are supported, but configuration is restricted to TCP/IP connections. To an MQSeries client or queue manager, MQIPT appears as if it is the destination queue manager. Where channel configuration requires a destination host and port number, the MQIPT host name and listener port number are specified.

Client/server channels

MQIPT listens for incoming client connection requests, and then forwards them (either using HTTP tunneling or as standard MQSeries protocol packets). If MQIPT is using HTTP tunneling, it forwards them on a connection to a second MQIPT. If it is not using HTTP tunneling, it forwards them on a connection to what it sees as the destination queue manager (although this could in turn be a further MQIPT). Once the queue manager has accepted the client connection, packets are relayed between client and server.

Sender/receiver
If MQIPT receives an incoming request from a sender channel, it forwards it to the next MQIPT or destination queue manager in exactly the same way as for client connection channels. The destination queue manager validates the incoming request and starts the receiver channel if appropriate. All communications between sender and receiver channel (including security flows) are relayed.

Requester/server
This combination is handled in the same manner as the types above. Validation of the connection request is performed by the server channel at the destination queue manager.

Requester/sender
The ‘callback’ configuration could be of use if the two queue managers are not allowed to establish direct connections to each other, but are both allowed to connect to MQIPT and to accept connections from it.

Server/requester and server/receiver
These are handled by MQIPT just like the Sender/Receiver configuration.

Normal termination and failure conditions

When MQIPT detects closure (either normal or abnormal) of an MQSeries channel, it propagates the channel closure. If the administrator closes down a route through the MQIPT, all channels going through that route are closed.

MQIPT provides an optional idle time-out facility. If MQIPT detects that a channel has been idle for a period of time exceeding the timeout, it performs an immediate shutdown on the two connections in question.

The two MQSeries systems at either end of the channel observe these abnormal termination conditions either as network failures, or as termination of the channel by their partner. The channels in question are then able to restart and recover (if the failure happens during a protocol in-doubt period) just as they would do if there were no MQIPTs.

Security considerations

MQIPT does not contain any mechanism to authenticate the originating channel or provide user-based access control to destination queue managers. MQIPTs allow channel security flows, so that MQSeries channel exits can be used to provide security over the entire channel from end to end.

MQIPT has several additional functions that help a designer build a secure solution:

- If there are many clients in an internal network all trying to make outgoing connections, they can all go through an MQIPT located inside the firewall.

The firewall administrator then has to grant external access only to the MQIPT's machine.

- MQIPT can connect only to queue managers for which it has been explicitly configured in its configuration file.
- It provides a connection log. When enabled, this facility logs all connections, successful or otherwise, detailing the host from which the connection was made, and the responding hostname. It also logs connection timeouts and disconnects.
- MQIPT verifies that the messages it receives and transmits are valid and conform to the MQSeries protocol. This helps prevent MQIPTs being used for security attacks outside of the MQSeries protocol.
- It allows channel exits to run their own end-to-end security protocols.
- MQIPT allows you to restrict the total number of incoming connections. This helps protect a vulnerable internal queue manager from denial-of-service attacks.

You must protect the MQIPT's configuration file, `mqipt.conf`, because this file controls access to the internal hosts, and you must prevent unauthorized access to the command port (if it is enabled) because such access allows an external person to shut down MQIPT.

Chapter 3. How to install internet pass-thru

This chapter describes how you install the product:

Downloading the files

MQIPT is downloaded from the MQSeries SupportPac Web page, at:

`www.ibm.com/software/ts/mqseries/downloads`

Follow the instructions for downloading.

Installing on Windows NT

Open a command prompt and unpack `ms81_nt.zip` into a temporary directory. Run the `setup.exe` and follow the online instructions.

MQIPT contains the following files:

File	Purpose
Readme.txt	Latest information not included in the publications
mqiptSample.conf	Sample configuration file
lib\mqipt.jar	Contains runtime, class, and property files
bin\mqipt.bat	Shortcut for running MQIPT from the command line
bin\mqiptadmin.bat	Shortcut for stopping MQIPT and refreshing file information
bin\mqiptservice.exe	For adding/removing MQIPT to/from the Windows NT Service Control Manager
bin\mqiptservice118.exe	For adding/removing MQIPT to/from the Windows NT Service Control Manager, if using JDK at a level earlier than 1.2.0
bin\mqiptVersion.bat	Displays the version number of MQIPT
doc\mqipt.pdf	The *internet pass-thru* manual

The installer updates the system CLASSPATH environment variable with the location of the `mqipt.jar` file.

Installing on Sun Solaris

Login as `root`, uncompress and unpack `ms81_sol.tar.Z` into a temporary directory. Run the `pkgadd` command, as in this example:

```
login root
cd /tmp
uncompress -fv ms81_sol.tar.Z
tar xvf ms81_sol.tar
pkgadd -d . mqipt
```

The example assumes that `ms81_sol.tar.Z` is in the /tmp directory.

MQIPT contains the following files:

File	Purpose
Readme.txt	Latest information not included in the publications
mqiptSample.conf	Sample configuration file
lib/MQipt.jar	Contains runtime, class, and property files
bin/mqipt	Shortcut for running MQIPT from the command line
bin/mqiptAdmin	Shortcut for stopping MQIPT and refreshing file information
bin/mqiptVersion	Display the version number of MQIPT
bin/mqiptService	For installing MQIPT so that it starts automatically at system startup.
bin/mqiptEnv	Defines the location of the `mqipt.jar` file and is used only by the other scripts.

Updating the default configuration file

Before starting MQIPT, copy the sample configuration file, `mqiptSample.conf`, to `mqipt.conf` and edit this new file, using your editor of choice, to change the sample route definitions to meet your requirements. See Chapter 4, “How to use internet pass-thru” on page 22 for a full description of the attributes in this configuration file.

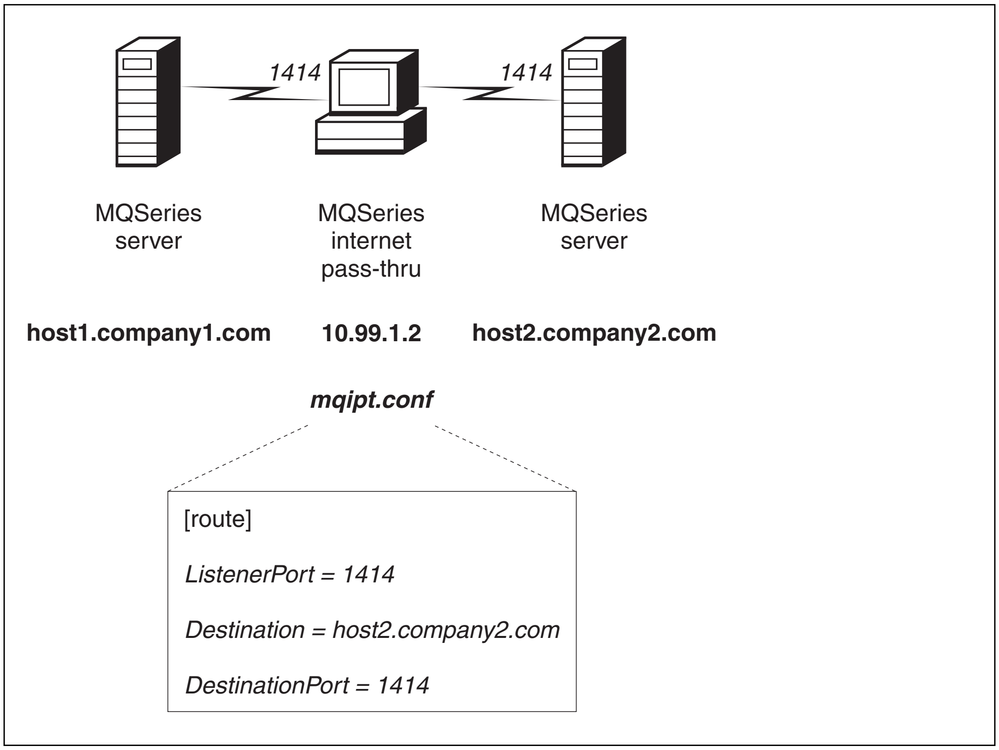
```
#
# Global definitions for all routes
#
# CommandPort, ConnectionLog, RemoteShutDown and MaxLogFileSize can only
# be used in the global definition.
#
[global]
#CommandPort=1881
RemoteShutDown=true
MinConnectionThreads=5
MaxConnectionThreads=100
IdleTimeout=20
ClientAccess=false
QMgrAccess=true
HTTP=false
HTTPChunking=false
Trace=0
ConnectionLog=true
MaxLogFileSize=50

#
# First route definition example
#
# ListenerPort, Destination and DestinationPort can only be used in a
# route definition.
#
[route]
ListenerPort=1414
Destination=host2.company2.com
DestinationPort=1414

#
#
# Second route definition example
#
#[route]
#ListenerPort=1415
#Destination=host1.company1.com
#DestinationPort=1414
```

Figure 4. Part of the default configuration file

The two sample diagrams below show how the route definitions have been defined on each MQIPT.



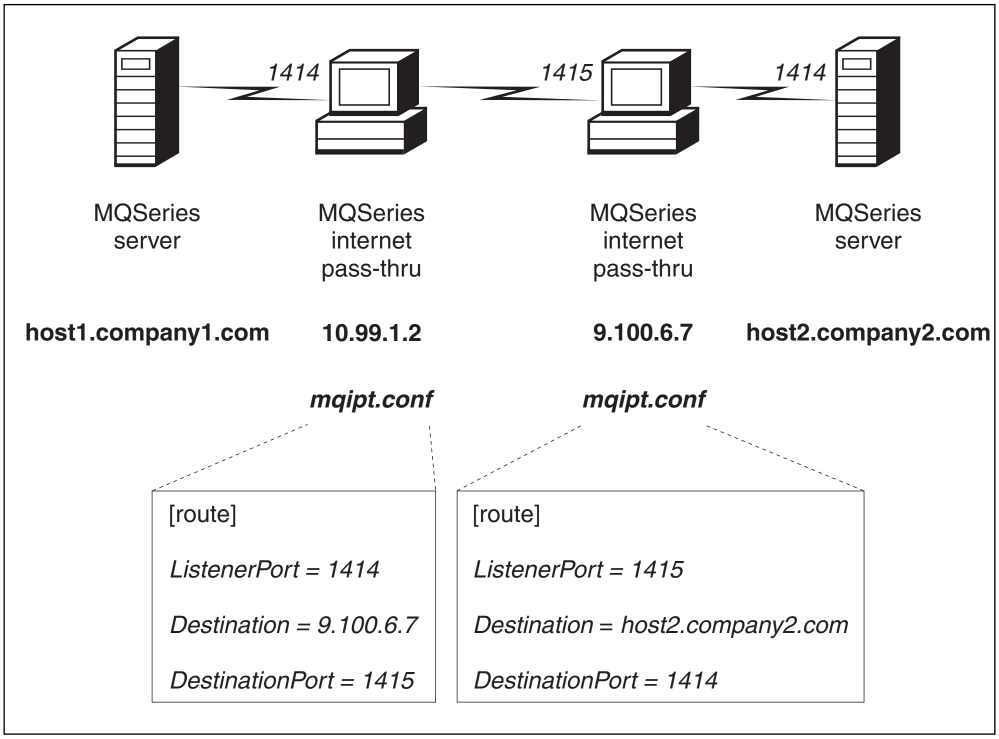
```
[route]

ListenerPort = 1414

Destination = host2.company2.com

DestinationPort = 1414
```

Figure 5. Example mqipt.conf definitions with one MQIPT



```
[route]

ListenerPort = 1414

Destination = 9.100.6.7

DestinationPort = 1415
```

```
[route]

ListenerPort = 1415

Destination = host2.company2.com

DestinationPort = 1414
```

Figure 6. Example mqipt.conf definitions with two MQIPTs

Starting internet pass-thru from the command line

For Windows NT, open a command prompt and change directory to the `bin` directory. For example:

```
c:
cd \mqipt\bin
mqipt ..
```

You can also start MQIPT from the Windows NT Start menu.

For Sun Solaris, login as `root` and change directory to the `bin` directory. For example:

```
cd /opt/mqipt/bin
mqipt ..
```

Running the `mqipt` script without any options uses a default location of "." for the configuration file (`mqipt.conf`). To specify a different location:

```
mqipt <directory name>
```

Messages will appear on the console showing the status of MQIPT. If an error occurs, see "Problem determination" on page 26. The following messages are an example of MQIPT successfully starting:

```
5639-L92 (C) Copyright IBM Corp. 2000 All Rights Reserved
MQCPI001 MQSeries internet pass-thru version 1.0 starting.
MQCPI004 Reading configuration information from E:\MQIPT\mqipt.conf.
MQCPI011 The path E:\MQIPT\logs will be used to store the log files.
MQCPI006 Route 1414 has Started and will forward messages to
         host2.company2.com(1414), using HTTP tunneling.
```

The following subdirectories of the `mqipt` home directory are created automatically when MQIPT is invoked for the first time:

- A "logs" directory in which the connection log is kept
- An "errors" directory in which any First Failure Support Technology™(FFST™) and trace records are written

Starting internet pass-thru automatically in Sun Solaris

To start MQIPT automatically when the system is started, run the `mqiptService` script. For example:

```
login root
cd /opt/mqipt/bin
mqiptService -install
```

To prevent MQIPT from starting automatically:

```
login root
cd /opt/mqipt/bin
mqiptService -remove
```

Using a Windows NT service control program

A separate service control program, `mqiptservice.exe`, is provided to allow MQIPT to be managed and started as a Windows NT service.

`mqiptservice.exe` takes the following command-line arguments:

mqiptservice -install *path*
Installs and registers the service, so that it appears on the NT services panel as a manual service. Go to the services panel and change the setting to “automatic” to make MQIPT start automatically. You have to reboot Windows NT after installing this service. The path parameter, which must be supplied, is the path to the directory containing the `mqipt.conf` configuration file.

mqiptservice -remove
Removes the service, making it disappear from the services panel.

mqiptservice ?
Displays US English help messages listing the valid arguments.

Specifying both install and remove on the same command causes an error.

Windows NT internally invokes the `mqiptservice` program with no arguments. If you call it from the command line with no arguments, the program times out and returns with an error.

When the MQIPT service is started, all active MQIPT routes start up. When it is stopped, all routes are subjected to immediate shutdown.

Note: If you have a release of JDK earlier than 1.2.0, use `mqiptservice118.exe` instead of `mqiptservice.exe`.

Uninstalling internet pass-thru as a Windows NT Service

You uninstall MQIPT as a service by first stopping it from the Windows NT services panel. Then open a command prompt, go to MQIPT’s `bin` subdirectory, and type:

```
mqiptservice -remove
```

Note: If you have a release of JDK earlier than 1.2.0, use `mqiptservice118 -remove`.

Uninstalling internet pass-thru

For **Windows NT**, before uninstalling MQIPT from your system, remove it as a Windows NT Service, as described above. Then run the uninstall process from the Windows NT Start menu.

For **Sun Solaris**, before uninstalling MQIPT from your system, prevent it from starting automatically, as described in “Starting internet pass-thru automatically in Sun Solaris” on page 20. Login as `root` and run the `pkgrm` command:

```
login root
pkgrm mqipt
```

Chapter 4. How to use internet pass-thru

This chapter describes how you configure the product and how you use it:

Administering internet pass-thru

Basic administration is achieved by editing the `mqipt.conf` file.

If the global section of `mqipt.conf` specifies a value for `CommandPort`, MQIPT listens on this port for the following ASCII administration commands:

STOP
MQIPT closes all connections, stops listening for incoming connections, and then exits. This command is ignored unless the `mqipt.conf` file specifies `RemoteShutDown=true`.

REFRESH
MQIPT re-reads `mqipt.conf`. If it finds:

- That any of the routes currently active are now marked as inactive (or are missing altogether), it closes them down and stops listening for incoming connections on those routes.
- Any routes marked active in the configuration file that it doesn't currently have running, it starts them up.
- That the configuration parameters of a currently running route have changed, it applies the changed values to those routes. Where possible (for example, a change to the setting of trace) it does this without disruption to running connections. For some parameter changes (for example, a change to a destination), MQIPT has to close all connections before effecting the change.

Use the `mqiptadmin` script command to administer these commands:

```
mqiptadmin -refresh {hostname {port} }     sends the refresh command
mqiptadmin -stop    {hostname {port} }     sends the stop command
```

If not provided, hostname defaults to `localhost` and port to `1881`.

MQIPT sends no reply or acknowledgement to commands, so the utility cannot indicate success or failure of any command. A response to commands is a candidate for any future release of MQIPT.

On **Windows NT**, these administrative functions are also available from the Start menu.

Configuring internet pass-thru

Basic configuration information is kept in a file called `mqipt.conf`. See page 17 for an example of the file. The `mqipt.conf` file comprises a set of sections, or stanzas. There is one global section, and an additional section for each route that has been defined through MQIPT. In the simplest configuration, there is only one route, so the file contains two sections, one global and one route section.

Each section contains name/value attribute pairs. Some attributes can appear only in the global sections, some can appear only in the route sections, and some may appear both in route and global sections. If an attribute does appear in both route and global sections, the attribute value in the route section overrides the global value, but only for the route in question. In this way, the global section can be used to establish the default values to be used for those attributes not set in the individual route sections.

The global section starts with a line containing the characters `[global]` and ends when the first route section starts. The global section must precede all route sections in the file. Each route section starts with a line containing the characters `[route]` and ends when the next route section starts, or when the end of the configuration file is reached.

Any unrecognized keyword names (that is to say, any name/value pairs where the name is not one of the names defined in this document) are ignored. If a name/value pair appearing in a route section has a recognized name but has an invalid value (for example `MinConnectionThreads=x` or `HTTP=unsure`), that route is disabled (that is, it does not listen for any incoming connections). If a name/value pair appearing in the global section has a recognized name but has an invalid value, all routes are disabled and MQIPT does not start. Where an attribute is listed as taking the values `true` and `false`, any mixture of upper- and lower-case can be used.

Note that keyword names are always case sensitive; for example, **IdleTimeout** is a valid keyword, but **IdleTimeOut** is not valid and will be ignored, with no error message produced.

Global section

The global section may contain the following attributes and all the attributes in “Route section” on page 24, apart from ListenerPort, Destination and DestinationPort.

CommandPort

The internet protocol port on which MQIPT listens for configuration commands from the `mqiptadmin` utility. The command port value is read when MQIPT is first started. If the value in the file is changed while MQIPT is running, the new value is not actioned until MQIPT is restarted (this is in contrast to other values in the file which take effect when a REFRESH command is sent to the port).

If the `CommandPort` attribute is not present, MQIPT does not listen for configuration commands. If you want to listen on the command port, you are recommended to use 1884, which is the default assumed by the admin utility.

RemoteShutDown
Either `true` or `false`. When true (and when there is a command port) MQIPT shuts down whenever a STOP command is received on the command port. The default value is false.

ConnectionLog
Either `true` or `false`. When true, MQIPT logs all connection attempts (successful or otherwise) in the `logs` subdirectory and disconnection events to the file `mqipt.log`. The default value is true.

MaxLogFileSize
The maximum size (specified in KB) of the `mqipt.log` connection log file. When the `mqipt.log` file size increases above this maximum a backup copy `mqipt.back` is made, and a new file is started. Only one backup file is kept; each time the main log file fills up, any earlier backups are erased. The default value is 50, the minimum allowed value is 5.

Route section

The route section may contain the following attributes:

ListenerPort
The port number on which the route should listen for incoming requests. Each route section **must** contain an explicit ListenerPort value; moreover, the ListenerPort values set in each section must be distinct. Any valid port number can be used, including ports 80 and 443, provided that the ports chosen are not already in use by any other TCP/IP listener running on the same host.

Destination
The hostname (or dotted decimal IP address) of the queue manager (or subsequent MQIPT) to which this route is to connect. Each route section **must** contain an explicit Destination value. You are allowed to have several route sections pointing to the same Destination.

If a change to this property affects a route, the route is stopped and restarted when a REFRESH command is issued. All connections to this route are terminated.

DestinationPort
The port on the Destination host to which this route is to connect. It is valid for more than one route to point at the same combination of Destination and DestinationPort. Each route section **must** contain an explicit DestinationPort value.

If a change to this property affects a route, the route is stopped and restarted when a REFRESH command is issued. All connections to this route are terminated.

HTTP
Set this to `true` for routes responsible for making outbound HTTP tunneling requests (that is, communicating with another MQIPT). Set to `false` for routes directed at MQSeries queue managers.

If this property is changed to `false`, the route is stopped and restarted when a REFRESH command is issued. All connections to this route are terminated.

To use HTTP chunking, set this attribute to `true`.

HTTPChunking

Set this to `true` for routes responsible for making outbound requests using HTTP tunneling with chunking. The HTTP attribute must also be set to `true`. Set to `false` when you are not using HTTP chunking.

If this property is changed to `false`, the route is stopped and restarted when a REFRESH command is issued. All connections to this route are terminated.

Active

The route accepts incoming connections only if the value of Active is set to `true`. This means that you can temporarily shut off access to the destination, by setting `Active=false`, without having to delete the route section from the configuration file.

If this property is changed to `false`, the route is stopped when a REFRESH command is issued. All connections to this route are terminated.

ClientAccess

The route allows incoming client channel connections only if the value of ClientAccess is set to `true`. Note that potentially you can configure MQIPTs to accept client requests only, queue manager requests only, both types of request, or neither type.

If this property is changed to `false`, the route is stopped when a REFRESH command is issued. All connections to this route are terminated.

QMgrAccess

The route allows incoming queue manager channel connections (for example Sender channels) only if the value of QMgrAccess is set to the value `true`.

If this property is changed to `false`, the route is stopped when a REFRESH command is issued. All connections to this route are terminated.

MinConnectionThreads

The minimum number of connection threads (threads to handle incoming connections on this route). This is the number of threads allocated when the route is started, and the total number of threads allocated does not drop below this value during the time the route is active. The minimum allowed value is 0 and the value must be less than that specified for MaxConnectionThreads.

Changes to this property take effect only when the route is restarted.

MaxConnectionThreads

The maximum number of connection threads, and thus the maximum number of concurrent connections, that can be handled by this route. If this limit is reached, the `MaxConnectionThreads` value also indicates the number of connections that will be queued once all the threads are in use. Beyond that number, subsequent connection requests are refused.

The minimum allowed value is 1.

If a change to this property affects a route, the new value is used when the REFRESH command is issued. All connections pick up the new value immediately. The route is not terminated.

IdleTimeout

The time, in minutes, after which an idle connection is closed. Note that queue manager to queue manager channels also have the DISCINT attribute. If you set the IdleTimeout parameter, take note of DISCINT.

A value of 0 indicates no idle timeout.

Changes to this property take effect only when the route is restarted.

Trace

The level of tracing required can be specified by an integer in the range 0-5. A value of 0 means no tracing; 5 requests full tracing.

If a change to this property affects a route, the new value is used when the REFRESH command is issued. All connections pick up the new value immediately. The route is not terminated.

Default attributes

If an attribute is missing from both the route section and the global section, the following defaults are used:

Attribute	**Default value**
DestinationPort	1414
HTTP	False
HTTPChunking	False
Active	True
ClientAccess	False
QMgrAccess	True
MinConnectionThreads	5
MaxConnectionThreads	100
IdleTimeout	0
Trace	0

Maintenance

The configuration file, `mqipt.conf`, is the only file that must be backed up on a regular basis. Include it in your normal backup procedures.

Problem determination

There are some common pitfalls to check first if you encounter a problem:

- The properties entered are in the wrong case. The properties are case-sensitive.
- The MQIPT system has just been installed and has not been rebooted.
- HTTP has been set to true on a route directly connected to a queue manager.
- The CLASSPATH has not been set up correctly.

The next step is to follow the flowchart shown in Figure 7 on page 27. The numbers refer to the following notes.

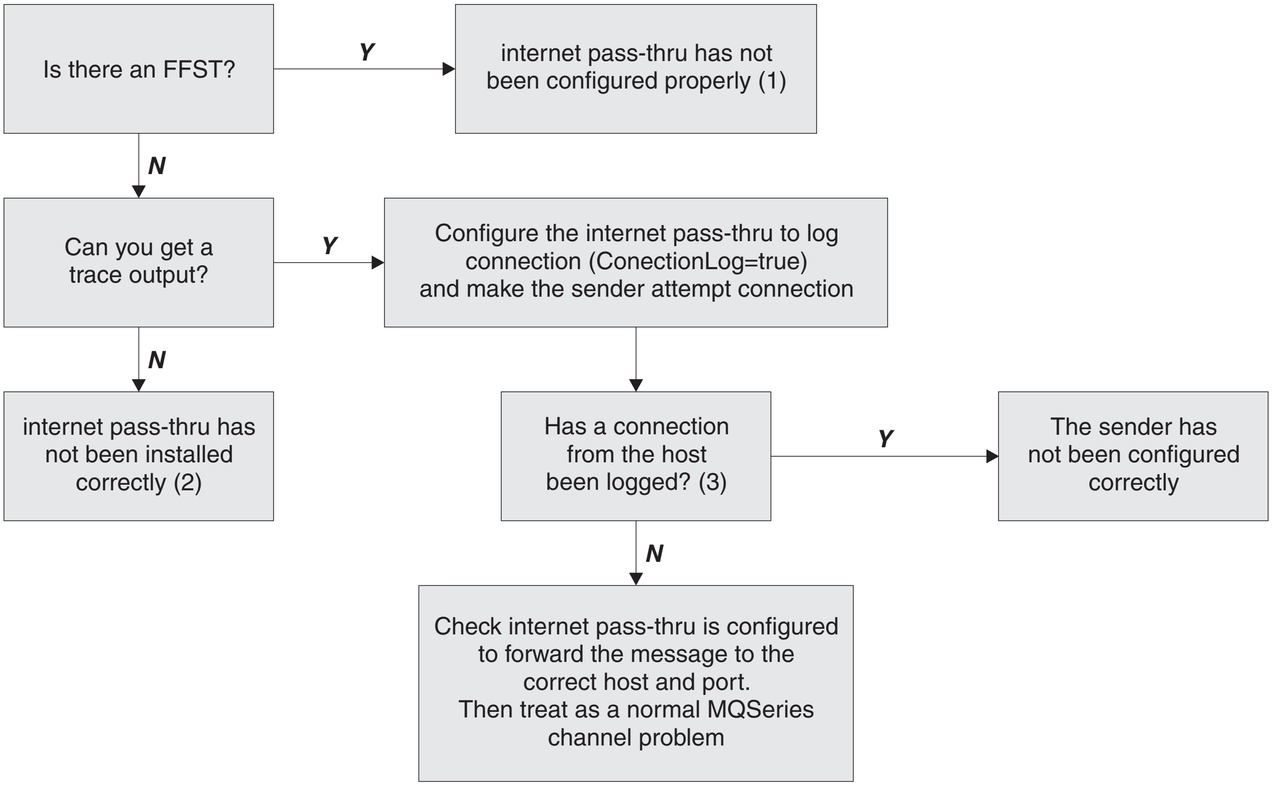

Is there an FFST?
Y
internet pass-thru has not been configured properly (1)
N
Can you get a trace output?
Y
Configure the internet pass-thru to log connection (ConectionLog=true) and make the sender attempt connection
N
internet pass-thru has not been installed correctly (2)
Has a connection from the host been logged? (3)
Y
The sender has not been configured correctly
N
Check internet pass-thru is configured to forward the message to the correct host and port.
Then treat as a normal MQSeries channel problem

Figure 7. Problem determination flowchart

Notes:

1. If you find any FFST reports (in the errors subdirectory), you know that MQIPT was correctly installed. There might have been a problem with the configuration.

   Each FFST reports a problem that causes MQIPT, or a route, to terminate its startup process. Fix the problem that caused each FFST. Then delete the old FFSTs and restart or refresh MQIPT.

2. If MQIPT has not been installed correctly, check that all the files have been put in the correct place and the CLASSPATH has been updated. To check this is correct, try to start MQIPT manually.

3. Manually starting MQIPT.

   Open a command prompt. Go to the `bin` subdirectory and type:

   ```
   mqipt xxx
   ```

   where *xxx* is the MQIPT home directory; in this case, it is "`..`".

   This will start MQIPT and look for the configuration in the home directory. Look for any error messages and FFSTs in the `errors` subdirectory.

   Look at the text output from MQIPT for any error messages and correct the error(s). Check for FFSTs and correct any errors. MQIPT will not start if there is a problem in the global section of the configuration file. A route will not start if there is a problem in the route section of the configuration file.

Automatically starting internet pass-thru

If you install MQIPT as a Windows NT Service, and have changed its startup to be automatic, it starts when the system is brought up. Always start MQIPT manually once before trying to install MQIPT as a Windows NT Service to confirm correct installation. See “Using a Windows NT service control program” on page 21 for more details.

If you receive the error message “Unable to locate DLL...”, either you are using the wrong `mqiptservice` program or you have not configured the system CLASSPATH correctly. Use `mqiptservice.exe` for JDK 1.2.0 or above, and `mqiptservice118.exe` for earlier releases. To use `mqiptservice.exe`, include the classic subdirectory of JDK in the system path. For example, add `c:\jdk122\jre\bin\classic;` to the system PATH on the Windows NT System Properties panel. Then reboot the system.

Checking for end-to-end connectivity

If MQIPT is correctly installed, the next step is to check that the routes are set up correctly.

In the configuration file, `mqipt.conf`, set the `ConnectionLog` property to true. Start or refresh MQIPT and attempt a connection. The connect log is created in the logs directory below the home directory. If it is not created, you know that MQIPT has not been installed correctly. If no connection attempts are recorded, the sender has not been set up correctly. If attempts are recorded, check that MQIPT is forwarding the messages to the correct address.

Tracing errors

MQIPT provides a detailed execution trace facility, which is controlled by the trace attribute. Each route can be traced independently. Trace files are written to the *xxx*`\errors` directory (where *xxx* is the directory containing `mqipt.conf`. Each trace file produced has a name with the following format:

`mqiptroute`*nnnnn*`.log`

where *nnnnn* is the number of the port on which the route is listening. Trace output from threads not directly associated with any particular route (for example, the thread handling command input) is written to a separate file called `mqiptmain.log`.

Unexpected fatal errors are written as FFST records to an error log file, held in the *xxx*`\errors` directory (where *xxx* is the directory containing `mqipt.conf`). The FFST files have the following format:

`mqipt`*xxx*`.FFST`

where *xxx* is the sequence that the FFST was generated (1 is the oldest). In a long-running system, you might reach the maximum number the system can generate. In this case, any FFSTs that are generated are written to the file `mqipt0.FFST`. If the file `mqipt0.FFST` is created, you should stop and restart MQIPT at the first opportunity and delete the old files.

Performance

Here are some pointers for tuning your system.

Thread pool management

The relative performance of each route can be tuned using a combination of a thread pool and an idle timeout specification.

Connection threads

Each MQIPT route is assigned a working pool of concurrently running threads that handle incoming communication requests. At initialization, a pool of threads is created (of the size specified in the route's `MinConnectionThreads` attribute), and a thread is nominated to handle the first incoming request. When this request comes in, the thread is set to work on this request immediately, and the next thread assigned as ready for the next incoming request. When all threads are assigned to work, a new thread is created, added to the working pool, and assigned for work. In this way, the pool grows until `MaxConnectionThreads` is reached. When the number of working threads is at `MaxConnectionThreads`, the next incoming request waits until a thread is released back to the working pool. This is the maximum working capacity of the route, after which no additional requests can be accepted. Threads are released back to the pool when a conversation ends, or the specified idle timeout period has elapsed.

IdleTimeout

By default, working threads are not terminated because of inactivity. When a thread has been assigned to a conversation, it remains assigned to that conversation until it is closed normally, the route is deactivated, or MQIPT is shut down. Optionally, an idle timeout interval may be specified, so that any thread that has been inactive for the specified period of time (in minutes) is terminated. A monitor thread keeps a regular check on thread idle times, and terminates those that have exceeded the threshold. Threads are recycled for use by placing them back into the working pool.

Chapter 5. Messages

When run from the command line, MQIPT displays a small number of information and error messages on the console, in US English only. When an error occurs, the error message is also reported in an FFST.

MQCAE001, Unknown host: <text>
The internet pass-thru host cannot be found. Check you have correctly specified the hostname where the internet pass-thru is located.

MQCAI001, usage options: (-stop | -refresh) {hostname {port}}
This message is issued as a standard error to provide guidance in use of the administration function.

MQCAE002, The following error was reported by the system. <text>
An error has occurred. While following a system command, an error was reported.

MQCPE001, Directory does not exist or is not a directory
At initialization, a required directory could not be found. This message refers to a directory specified either in the internet pass-thru configuration file mqipt.conf or in the internet pass-thru command line startup options on the default directory.

MQCPI001, MQSeries internet pass-thru version 1 starting
This internet pass-thru instance is beginning execution. Further initialization messages will follow.

MQCPE002, An error occurred while trying to access the file system. The following exception was thrown: <text>
This error occurs because a file is locked out or missing. Read the error text and take appropriate corrective action.

MQCPI002, MQSeries internet pass-thru shutting down
internet pass-thru is going to shut down. This can result from a STOP command, or automatically if a configuration error prevents a successful startup or REFRESH action.

MQCPE003, Unable to listen on port <number>
internet pass-thru is unable to listen for requests on the specified port. See further messages and the error log directory for full details.

MQCPI003, MQSeries internet pass-thru shutdown complete
The shutdown process has completed. All internet pass-thru processes are now ended.

MQCPE004, Route startup failed on port <number>
It was not possible to start the route with the listed ListenerPort number. Further error messages and log records will provide further explanation of the problem.

MQCPI004, Reading configuration information from <directory name>
The internet pass-thru configuration file mqipt.conf is being read from the directory described in this message.

MQCPE005, The configuration file <number> could not be found
The internet pass-thru configuration file mqipt.conf could not be found in the specified directory

MQCPI005, Listener port specified as not active - <route number> -> <host name>(<port number>)
The route referred to in the message has been marked as inactive. No communication requests will be accepted on this route.

MQCPE006, The number of routes has exceeded <number>. The MQSeries internet pass-thru will start but this configuration is unsupported.
Your configuration has exceeded the maximum supported number of routes for one instance of internet pass-thru. Operation will not be halted but the system may become unstable or overloaded as a result. Configurations that exceed the stated maximum number of routes will not be supported. Consider starting additional instances of MQIPT with fewer routes per instance.

MQCPI006, Route <route number> has started and will forward messages to <route number> -> <host name>(<port number>), using <protocol>
An internet pass-thru route has been started on the listener port shown in this message. The message also shows details of the intended destination host/port. The protocol may be MQSeries protocol, HTTP tunneling, or HTTP chunking.

MQCPE007, Route not restarted on Listener Port <number>
On a REFRESH operation, the route that was operating on the specified ListenerPort was not restarted on the new configuration.

MQCPI007, Route stopped on port <port number>
The route that was operating on the specified ListenerPort is being shut down. This action normally occurs when a REFRESH command is issued to internet pass-thru and the route configuration has been changed

MQCPE008, Duplicate route defined for listener port <number>
More than one route has been defined with the same ListenerPort value.

MQCPI008, Listening for control commands on port <port number>
This internet pass-thru instance is listening for control commands on the specified port.

MQCPE009, LogPath parameter <text> is not valid.
The log path shown in the text either does not exist or is not accessible at the time.

MQCPI009, Control command received: <text>
This message indicates that a control command has been received at the command port. Where applicable, details are included in the message.

MQCPE010, Listener or command port number <text> is not valid
The port number supplied for the command port or listener port parameter is invalid. Use a valid port number. For guidance on use of port numbers in your network, consult your network administrator.

MQCPI010, Stopping command port on <number>
On a REFRESH operation, the command port is no longer in use in the new configuration. Commands will no longer be accepted at the specified port.

MQCPE011, The trace level <number> is outside the valid range 0 - 5
The specified trace option was requested, but it is not in the valid range 0-5.

MQCPI011, The path <directory name> will be used to store the log files
Logging output will be directed to the location described in this message, under the current configuration. This may change if the configuration is amended and a REFRESH operation is requested.

MQCPE012, The value <value> is not valid for the attribute <name>
An invalid parameter value has been specified. Refer to the chapters in this user guide for full details of the valid values for each control parameter.

MQCPI012, Changing the value of MinConnectionThreads has no effect once the route is started
The minimum number of connection threads is assigned at route startup and cannot be changed until internet pass-thru is restarted.

MQCPE013, Listener port property was not found in the [route] <text>
The listener port property is the primary and unique identifier for each route, and is therefore mandatory. internet pass-thru has detected a [route] stanza in the configuration file that does not contain a ListenerPort property.

MQCPI013, Connection from <host name> to host <host name> closed
This message is issued in the connection log to record connection activity

MQCPE014, Listener port property value <value> is not valid
An invalid port number has been specified for the ListenerPort property of a [route] stanza. The ListenerPort property represents the TCP/IP port number on this host, on which internet pass-thru will accept incoming connections from an MQSeries Client, MQSeries Queue Manager or an MQSeries internet pass-thru.

MQCPI014, Connection from <host name> to host <hostname> closed. The protocol was not recognized
This message is issued in the connection log to record connection activity

MQCPE015, No text was found for message number <number>
An internal error has been encountered for which no description is available. This condition should be reported to IBM Technical Support.

MQCPI015, Connection from a client on <host name> to host <host name> was rejected because client access has been disabled on this route
This message is issued in the connection log to record connection activity

MQCPE016, The maximum number of connection threads is <number> but this is less than the minimum number of connection threads, which is <number>
Your configuration has specified the minimum number of connection threads with a value exceeding the maximum number of connection threads. This could be an error in a single [route] stanza, a conflict between [global] property specification and [route] overriding specification, or [route] specification overriding system default values. Refer to the earlier chapters of this user guide for full details of the valid values and applicable defaults.

MQCPI016, Connection from a queue manager on <host name> to host <hostname> was rejected because queue manager access has been disabled on this route
This message is issued in the Connection Log to record connection activity

MQCPE017, The exception <text> was thrown which caused the internet pass-thru to shut down
internet pass-thru has abnormally terminated and has been shut down. This may have occurred owing to system environmental conditions or constraints, such as memory overflow. If the condition persists, contact IBM Technical Support.

MQCPI017, A queue manager on <host name> was connected to host <host name>
This message is issued in the Connection Log to record connection activity

MQCPE018, The ListenerPort property is blank. A route will not start
The ListenerPort number has been omitted in a [route] stanza. A value is required.

MQCPI018, A client on <host name> was connected to host <host name>
This message is issued in the Connection Log to record connection activity

MQCPE019, The stanza <name> was not found before the following; <name>
A sequence error has occurred in the configuration file. See “Configuring internet pass-thru” on page 23 for details of the required configuration file content and format.

MQCPI019 <number> routes have been created. This exceeds the maximum number of supported routes, which is <number>.
The maximum number of supported routes has been exceeded. MQIPT will continue to operate, but it is recommended that a second MQIPT instance is created and the routes split between the two.

MQCPE020, The new value for MaxConnectionThreads is <number>. This must be greater than the current value <number>.
Once the route has started the MaxConnectionThread property can only be increased.

MQCPE021, The property Destination was not supplied for route <number>
The property Destination is mandatory within a route, but was omitted in the route specified.

MQCPE022, The CommandPort value <number> is outside the valid range 1 - 65535.
The CommandPort property was outside the range 1-65535

MQCPI023, The HTTP property must also be set to true to use HTTP chunking.
HTTPChunking has been set to true, but HTTP is still set to false. Set HTTP=true and HTTPChunking=true to use HTTP chunking.

Sending your comments to IBM

MS81: MQSeries® internet pass-thru
Version 1.0

MS81 SCRIPT

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, use the Readers' Comment Form.
- By fax:
  - From outside the U.K., after your international access code use 44 1962 870229
  - From within the U.K., use 01962 870229
- Electronically, use the appropriate network ID:
  - IBMLink™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Readers' Comments

MS81: MQSeries® internet pass-thru
Version 1.0

MS81 SCRIPT

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Be sure to print your name and address below if you would like a reply.

Name

Company or Organization

Telephone

Address

Email

You can send your comments POST FREE on this form from any one of these countries:

Australia
Belgium
Bermuda
Cyprus
Denmark
Finland
France
Germany
Greece
Hong Kong
Iceland
Israel
Italy
Luxembourg
Monaco
Netherlands
New Zealand
Norway
Portugal
Republic of Ireland
Singapore
Spain
Sweden
Switzerland
United Arab Emirates
United States of America

If your country is not listed here, your local IBM representative will be pleased to forward your comments to us. Or you can pay the postage and send the form direct to IBM (this includes mailing in the U.K.).

1 Cut along this line

2 Fold along this line

By air mail
Par avion

IBRS/CCRI NUMBER: PHQ - D/1348/SO

NE PAS AFFRANCHIR

NO STAMP REQUIRED

REPONSE PAYEE
GRANDE-BRETAGNE

IBM United Kingdom Laboratories Limited
Information Development Department (MP 095)
Hursley Park
WINCHESTER, Hants
SO21 2ZZ United Kingdom

3 Fold along this line

From: Name ____________________
Company or Organization ____________________
Address ____________________
____
EMAIL ____________________
Telephone ____________________

1 Cut along this line

4 Fasten here with adhesive tape