Quick Transaction Protocol - QTP

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026 [1].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

1. Introduction

This protocol is intended to provide a short duration transaction service for dial-in and network connected POS, Visa, and other short duration transaction applications to transfer data between the access routers and transaction based servers over an IP network.

As the world has moved towards IP based backbone networks, a need has arisen for the ability to conduct short duration transactions over these networks. Such short duration transactions are integral to Visa, EFTPOS, security monitoring, alarm monitoring, meter reading, toll monitoring, location information (i.e. GPS), EDI, and a wide variety of other applications in similar environments.

The Quick Transaction Protocol (QTP) specifies a protocol to be used over IP networks for short duration transactions. It is based on the concept of peer to peer delivery of transaction data in a fashion analogous to the Connectionless Network Protocol (CLNP). As such it extends the use of CLNP over IP networks.

While QTP may be used as a transport protocol over either UDP or TCP, UDP is recommended as it minimizes the variable delay of the
network. This allows implementation of transaction protocols such as VISA II to be done in the transaction server as well as allowing for transactions where delivery of data must be fair to all customers such as in stock feeds.

Running of QTP over other protocols such as X.25 and TCP is a viable option where reliable transport is required and latency is not an issue. Should QTP be used over X.25 or over RFC 1490 frame relay, a network layer protocol identifier (NLPID) of 0xXX must be used. Should QTP be used over TCP, port 3350 should be used.

Key features of QTP are:

- Peer to peer architecture allowing for outgoing as well as incoming transaction initiation;
- Flow control via the ability of the transaction server or access router to gracefully degrade its service and if required, put itself out of service;
- Network efficiency by allowing multiple QTP messages to be encapsulated into a single UDP PDU.
- The ability to determine network performance and control flooding via status and statistics messages.

QTP does not provide reliable data flow (that is up to protocols such as TCP) and, in the case of running over UDP, does not allow for transmission of single messages of length greater than that of a single UDP PDU.

QTP does provide acknowledged control messages for the logical transaction connection establishment, connection release, and status requests.

1.1 Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [2].

1.2 Terminology

This document uses the following terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>POS</td>
<td>Point of Sale</td>
</tr>
<tr>
<td>QTP</td>
<td>Quick Transaction Protocol. The protocol described in this document.</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
</tbody>
</table>

2. Message Headers
2.1 IP/UDP Message Header

QTP messages are encapsulated within UDP PDUs which are in turn encapsulated within IP.

IP Address Fields
Source Address: The IP address of the interface from which the request is being issued.
Destination Address: The IP address of the interface to which the request is being issued.

UDP Port Fields
Source Port: 3350 (or other port number configured and agreed upon between the Source and Sink).
Destination Port: 3350 (or other port number configured and agreed upon between the Source and Sink).

2.2 Message Header

The UDP header is followed by the following QTP header. Multiple QTP messages may be encapsulated into a single UDP PDU. However, a
QTP message must not be split over a UDP packet boundary.

All numbers are unsigned and are in network byte order (i.e. big endian).

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Rsvd  |M|A|0|0| Type  |        Message Length         |
|        +--------------------------------------------------|
| Source LCN | Destination LCN                                  |
|        +--------------------------------------------------|
| Message Identifier (opt) |  Message Identifier Ack (opt) |                         |
|        +--------------------------------------------------|
| Data/Control Attributes                                         |
```

**Version (bits 1 through 4)**

**Rsvd (bits 5 through 8)**
Reserved for future use. Must be zero.

**Message Identifier Flag (bit 9)**
Indicates that the Message Identifier field exists.

**Message Identifier Ack Flag (bit 10)**
Indicates that the Message Identifier Ack field exists.

**Reserved (bits 11 and 12)**
Reserved for future use. Must be set to zero (0).

**Type (bits 13 through 16)**
The type of QTP message as in *section 2.3* defined below.

**Message Length (two octets)**
The length in bytes of the entire QTP message including header in network byte order.

**Source LCN (two octets)**
A logical channel number used to uniquely identify the entity initiating the QTP message. This number need not have any direct relationship with respect to physical ports, but must uniquely define a transaction source point so that messages sent to that LCN will be directed to the correct point.

**Destination LCN (two octets)**
A logical channel number used to identify the access point of the entity to which the QTP message is being routed. This number need not have any direct relationship with respect to physical ports, but must uniquely define a transaction destination point so that messages sent to that LCN will be directed to the correct point. "Call" messages will be assigned a destination LCN of "0", but the returned "Call Ack" will fill in the appropriate LCN in the "Source LCN" field, which is to be used for the remainder of the
transaction.

Message Identifier (two octets)
Present if the "Message Identifier Flag" is set in the QTP header. This is a number which uniquely identifies the QTP message for the indicated Source/Destination LCN pair. If this is present, an acknowledgment must be returned containing this number in the Message Identifier Ack field. If two messages are received on the same Source/Destination LCN pair with the same Message Identifier, then the second message will be discarded, and a Message Identifier Ack will be transmitted.

Message Identifier Ack (two octets)
Present if the "Message Identifier Ack Flag" is set in the QTP header. This number explicitly acknowledges reception of a previously received QTP message on the same Source/Destination LCN pair that contains the same number as its Message Identifier.

Data/Control
Data or control information formatted as Attribute, Attribute Length, and Attribute Value combinations as defined in this document.

2.3 Message Types

The following QTP message types have been defined.

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Type Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call Request</td>
<td>0x1</td>
</tr>
<tr>
<td>Call Ack</td>
<td>0x2</td>
</tr>
<tr>
<td>Call Reject</td>
<td>0x3</td>
</tr>
<tr>
<td>Clear Request</td>
<td>0x5</td>
</tr>
<tr>
<td>Clear Ack</td>
<td>0x6</td>
</tr>
<tr>
<td>Status Request</td>
<td>0x9</td>
</tr>
<tr>
<td>Status Report</td>
<td>0xA</td>
</tr>
<tr>
<td>Data</td>
<td>0xD</td>
</tr>
</tbody>
</table>
3.2 Call Ack
The QTP header for a Call Acknowledgment message is as follows.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Rsvd  |M|A|0|0| Type |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Source LCN | Destination LCN             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Message Identifier (opt) | Message Identifier Ack (opt) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Attribute n | Attribute n Length |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Attribute n Value . . . . . |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
3.3 Call Reject

The QTP header for a Call Reject message is as follows.

```
<table>
<thead>
<tr>
<th>Version</th>
<th>Rsvd</th>
<th>M</th>
<th>A</th>
<th>0</th>
<th>0</th>
<th>Type</th>
<th>Message Length</th>
</tr>
</thead>
</table>
```

```
+---------------------------------+-----------------+
| Source LCN (0)                  | Destination LCN|
+---------------------------------+-----------------+
| Message Identifier (opt)        | Message Identifier Ack (opt) |
```

The Call Ack message may optionally contain transaction data.
3.4 Clear Request

The QTP header for a Clear Request message is as follows.

0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Rsvd |M|A|0|0| Type |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source LCN           |       Destination LCN         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Message Identifier (opt)    |  Message Identifier Ack (opt) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Attribute n         |      Attribute n Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
A Clear Request message must contain the reason for the Call Clearing as defined in this document. It may optionally contain transaction data.

3.5 Clear Ack

The QTP header for a Clear Ack message is as follows.

0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  Rsvd |M|A|0|0| Type  |        Message Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source LCN           |       Destination LCN         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Message Identifier (opt)    |  Message Identifier Ack (opt) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Attribute n         |      Attribute n Length       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Attribute n Value . . . . .
3.6 Status Request

The QTP header for a Status Request message is as follows.

```plaintext
+---+---+---+---+---+---+---+---+
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
+---+---+---+---+---+---+---+---+
| 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 |
+---+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 |
+---+---+---+---+---+---+---+---+
| 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
+---+---+---+---+---+---+---+---+
| Version | Rsvd | M | A | 0 | 0 | Type | Message Length |
+-----------------------------+-----------------------------+
| Source LCN | Destination LCN |
+-----------------------------+-----------------------------+
| Message Identifier (opt) | Message Identifier Ack (opt) |
+-----------------------------+-----------------------------+
| Attribute n | Attribute n Length |
+-----------------------------+-----------------------------+
| Attribute n Value . . . . |
+-----------------------------+
```
3.7 Status Report

Status Reports may be solicited via a Status Request, or may be
unsolicited. A Status Report is how a device would advertise its
level of availability.

The QTP header for a Status Report message is as follows.

```
 0                   1                   2                   3
+-----------------------------------------------+
| Version | Rsvd  | M|A|0|0| Type  |        Message Length         |
+-----------------------------------------------+
|          Source LCN           |       Destination LCN         |
+-----------------------------------------------+
|   Message Identifier (opt)    |  Message Identifier Ack (opt) |
+-----------------------------------------------+
|   Attribute n         |      Attribute n Length       |
+-----------------------------------------------+
|   Attribute n Value . . . . . |
```

Source LCN
The number indicating the access point of the entity initiating the
QTP message.
The default Source LCN must be zero (0), unless referencing a
specific transaction (defined non-zero LCN pair).

Destination LCN
The number indicating the access point of the entity to which the
QTP message is being routed.
The default Destination LCN must be zero (0), unless referencing a
specific transaction (defined non-zero LCN pair).

Attribute n
Valid Status Request attribute(s) are as described in this document.

Attribute n Value
Valid Status Request attribute value(s) are as described in this
document.

Multiple Attributes may be contained within a Status Request
message.
Version
1
Type
0xA for Status Report.

Source LCN
The number indicating the access point of the entity initiating the QTP message.

The default Source LCN must be zero (0), unless referencing a specific transaction (defined non-zero LCN pair).

Destination LCN
The number indicating the access point of the entity to which the QTP message is being routed.

The default Destination LCN must be zero (0), unless referencing a specific transaction (defined non-zero LCN pair).

Attribute n
Valid Status Report attribute(s) are as described in this document.

Attribute n Value
Valid Status Report attribute value(s) are as described in this document.

Multiple Attributes may be contained within a Status Report message.

---

Cornish, et. al.        Expires December 1998                      12
Cornish, et. al.        Expires December 1998                      12
Quick Transaction Protocol (QTP)          June 1998
Quick Transaction Protocol (QTP)          June 1998

3.8 Data

The QTP header for a Data message is as follows.

```
+-----------------------------------------------+
|       |       |       |       |       |       |       |       |
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----------------------------------------------+
| Version | Rsvd | M|A|0|0| Type | Message Length |
+-----------------------------------------------+
| Source LCN | Destination LCN |
+-----------------------------------------------+
| Message Identifier (opt) | Message Identifier Ack (opt) |
+-----------------------------------------------+
| Attribute n | Attribute n Length |
+-----------------------------------------------+
| Attribute n Value . . . . |
+-----------------------------------------------+
```

Version
1

Type
0xD for Data.

Source LCN
The number indicating the access point of the entity initiating the QTP message.

Destination LCN
The number indicating the access point of the entity to which the QTP message is being routed.

Attribute n
Valid Data attribute(s) are as described in this document.

Attribute n Value
Valid Data attribute value(s) are as described in this document.

4. Attributes

Attributes are used within QTP to pass additional information not contained in the standard QTP message header. Attributes must not be nested. That is, attributes must not be put within attributes. Also, there MUST be no duplication of attributes within any individual QTP PDU.

Each attribute consists of a 16 bit attribute number, a 16 bit length field (that includes the attribute number and length fields), and the attribute value.

Numeric attribute values shall be in network byte order.

Attributes values have been grouped into five (5) classes related to the phase of the transaction and other "out-of-band" control activities, namely, session establishment, data transfer, session release, element status and statistical information.

The first 8 bits indicate the Attribute Class, and the remaining 8 bits indicate the member of the class.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Attribute Value</th>
<th>Class</th>
<th>Member</th>
</tr>
</thead>
</table>
4.1 Message Type / Attribute Matrix

The following matrix indicates the valid attributes which may be used for each defined message type.

* Indicates Attribute MAY be used in the corresponding Message Type.
- Indicates Attribute MUST NOT be used in the corresponding Message Type.

<table>
<thead>
<tr>
<th>Message / Attribute</th>
<th>Call Req</th>
<th>Call Ack</th>
<th>Call Rej</th>
<th>Clear Req</th>
<th>Clear Ack</th>
<th>Clear Rej</th>
<th>Status Req</th>
<th>Status Ack</th>
<th>Status Rej</th>
<th>Data Req</th>
<th>Data Ack</th>
<th>Data Rej</th>
<th>Time Since Last Reboot</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0100 CD Num</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0101 CG Num</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0102 Profile</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0103 Speed</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0200 Data</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0201 Mgmt</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0202 Q Data</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0300 Cause</td>
<td>*</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0400 FlowCtrl</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0x0401 Stn Stat</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
4.2 Attribute Definitions

This section defines the format of each attribute declared in section 4.0.

The hexadecimal value of each attribute is shown in brackets after the attribute name.

Called Party Number (0x0100)
This attribute specifies the called party number, indicating the phone number the user dialed to connect to the network for incoming calls and alternatively, the number used to connect to the client for outgoing calls. The attribute value is a character string up to 40 characters long.

Calling Party Number (0x0101)
This attribute specifies the Calling Line Identity, indicating the phone number of the user that wants to connect to the network for incoming calls and alternatively, the number used to identify the Host for outgoing calls. The attribute value is a character string up to 40 characters long.

Profile (0x0102)
This attribute enables access providers to classify user sessions, such as indicating the profile to be used in the connection. Its value is an alphanumeric text string of up to 40 characters.

Speed (0x0103)
This attribute specifies the speed of the connection at the access point (allowing for protocol handling within the IP network). Its value is an ASCII string of up to 10 characters indicating the speed in bits/sec.

Data (0x0200)
This attribute identifies transaction data.

Management Information (0x0201)
This attribute allows the capability to transfer unformatted data between two peer entities out-of-band from the transaction data. (for further study).

Qualified Data (0x0202)
This attribute identifies that the data contained within is to be used for session management (e.g. X.29 control messages, QLLC XID’s, etc.).

Cause (0x0300)
This attribute specifies a transaction release cause. The format of the Cause Attribute Value Field is as follows:

Cause Value: 8 bits
Broken down into Class and Member subfields as follows:

3 Bits (0-2) : Class
5 Bits (3-7) : Member

Diagnostic: 8 bits
Included when Cause Attribute Length is greater than 5. (Usage to be determined)

Information:
Included when Cause Attribute Length is greater than 6. A String of up to 40 characters.

The following table contains the list of valid Cause Values per Class.

<table>
<thead>
<tr>
<th>Class</th>
<th>Member</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>%000 Protocol Error</td>
<td>%000001 Unsupported Version</td>
<td>0x01</td>
</tr>
<tr>
<td></td>
<td>%00010 Invalid Use Of Flag</td>
<td>0x02</td>
</tr>
<tr>
<td></td>
<td>%00011 Invalid Message Type</td>
<td>0x03</td>
</tr>
<tr>
<td></td>
<td>%00100 Invalid Message Length</td>
<td>0x04</td>
</tr>
<tr>
<td></td>
<td>%00101 Invalid Source LCN</td>
<td>0x05</td>
</tr>
<tr>
<td></td>
<td>%00110 Invalid Dest LCN</td>
<td>0x06</td>
</tr>
<tr>
<td></td>
<td>%00111 Invalid Attribute</td>
<td>0x07</td>
</tr>
<tr>
<td></td>
<td>%01000 Invalid Attribute Length</td>
<td>0x08</td>
</tr>
<tr>
<td>%001 ProcedureError</td>
<td>%00001 Invalid LCN Pair</td>
<td>0x21</td>
</tr>
<tr>
<td></td>
<td>%00010 Invalid Attribute Usage</td>
<td>0x22</td>
</tr>
<tr>
<td></td>
<td>%00011 Timeout</td>
<td>0x23</td>
</tr>
<tr>
<td>%010 Invalid Message</td>
<td>%00001 Bad Calling Party Number</td>
<td>0x41</td>
</tr>
</tbody>
</table>
%00010  Bad Called Party Number    0x42
%00011  Bad Profile                0x43
%00100  Bad Speed                  0x44
%00101  Missing Attribute          0x45

%011 Network
%00001  Number Busy                0x61
%00010  No Network User Responding  0x62
%00011  Destination Unreachable    0x63
%00100  Synchronisation Error       0x64
%00101  Network Congestion          0x65
%00110  Resource                   0x81
%00010  SW Resources Unavailable    0x82
%00011  Network Resource Unavail    0x83

%101 User
%00001  Normal Clearing             0xA1
%00010  Max Packet Size Exceeded    0xA2
%00011  Flooding                    0xA3
%00100  Out Of Order                0xA4
%00101  Invalid Response            0xA5
%00110  User Not Responding         0xA6

Cornish, et. al.        Expires December 1998               17
Quick Transaction Protocol (QTP)      June 1998

Flow Control (0x0400)
This attribute indicates that the ability of the reporting station
to service further transactions. The attribute value is an 8 bit
number indicating:

Available (1)
Partially congested (2)
Congested (3)
Shutdown (4)

Station Status (0x0401)
This attribute indicates that the relationship of the sending
station to the receiving station. The attribute value is an 8 bit
number with 1 indicating primary, 2 indicating secondary, etc.

Ping (0x0402)
This attribute is used to identify if a remote station exists and to
determine network timing. It must be responded to regardless of the
ability of the receiving station to receive transactions. The
attribute value is an arbitrary binary message which will be echoed
in the resulting Status Report. This attribute must not be
contained in an unsolicited Status Report.

Call State  (0x0403)
This attribute is used to identify the state of a transaction (ie.
LCN pair). The attribute value is an 8 bit number indicating:

Disconnected       (0)
QTP Call received   (1)
QTP Call sent       (2)
QTP Clear received  (3)
QTP Clear sent      (4)
Connected          (5)

Number Of Messages Received on an Interface  (0x0500)
This attribute is used to indicate the number of messages the
station has received through the QTP Transport Service Port (e.g.
3350). The value of this attribute is a 32 bit number.

Number Of Messages Transmitted on an Interface  (0x0501)
This attribute is used to indicate the number of messages the
station has sent through the QTP Transport Service Port (e.g.
3350). The value of this attribute is a 32 bit number.

Number Of Unacknowledged Messages (on the system)  (0x0502)
This attribute is a global counter on a system element used to
indicate the number of unacknowledged QTP (control) messages
identified by the system. The value of this attribute is a 32 bit
number. This attribute provides some information about the health
of the network.

Cornish, et. al.        Expires December 1998                      18
Quick Transaction Protocol (QTP)          June 1998

Time Since Last Boot.   (0x0503)
This attribute specifies the time since last reboot. The value of
the attribute is a 32 bit number indicating the time since last
reboot in seconds.

5. Protocol Operation

QTP is a symmetrical protocol. Messages may be initiated at either
end of a connection, and hence transactions can be originated by
either peer entity.

The UDP protocol does not guarantee packet delivery. This is
satisfactory for transaction data, but is not sufficient for passing
call establishment and other control information over the UDP link.
For this reason, all message types in QTP except for Data are
acknowledged explicitly. If the initiator does not receive a
response to an acknowledged message within a configurable time frame
(generally 2 seconds), it will re-initiate the request. To
prevent the possibility of indefinite message looping, the
originator of a Clear Request Message may enter a "ClearAck Pending"
state awaiting a Clear Ack message. In the event that the Clear or
Clear Ack is lost, the entity originating the Clear Request message
shall resend the Clear Request message and release all transaction
resources.

The following sequences show protocol operation in some of the more
common environments. This does not preclude the use of the protocol
in other environments.
5.1 Call Request (incoming)

On receiving an incoming connection from a dial-up line or from a remote network, the Access Device initiates a Call Request to the appropriate Server. The Server has the option of accepting the call with a Call Ack or rejecting it with a Call Reject.

Call Acceptance:

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL SETUP</td>
<td>Call Request</td>
<td>Call Ack</td>
</tr>
</tbody>
</table>

Call Rejection:

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL SETUP</td>
<td>Call Request</td>
<td>Call Reject</td>
</tr>
</tbody>
</table>

5.2 Call Request (outgoing)

QTP is a bi-directional protocol. As such, the network may also set up an outgoing call. As in the incoming case, this Call Request may be accepted or rejected. The Call Rejection will contain the reason why the call was not accepted.

Call Acceptance:

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL SETUP</td>
<td>Call Request</td>
<td>Call Ack</td>
</tr>
</tbody>
</table>

Call Rejection

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
</table>
5.3 Call Clearing (incoming)

When the remote network or Access Device finishes a transaction or wishes to abort it, a Clear Request will be generated from the Access Device indicating the call is cleared and the reason for the clear. The Clear Request message may contain transaction data.

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL CLEARING</td>
<td>Clear Request</td>
<td>Clear Ack</td>
</tr>
<tr>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
</tr>
</tbody>
</table>

5.4 Call Clearing (outgoing)

A connection can be cleared at any point although normally this is only done if a transaction is complete.

In the case of a Server clearing the call the reason for clearing the call will be contained in the Clear Request message. The Clear Request message may contain transaction data.

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL CLEARING</td>
<td>Clear Request</td>
<td>Clear Ack</td>
</tr>
<tr>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
</tr>
</tbody>
</table>

5.5 Status Request

Status messages may be requested or reported by either end of the QTP connection.

<table>
<thead>
<tr>
<th>Terminal</th>
<th>Access Device</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status Request</td>
<td>Status Report</td>
<td>&lt;------------------</td>
</tr>
<tr>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
<td>&lt;------------------</td>
</tr>
</tbody>
</table>

5.6 Status Report

Status Reports may be initiated by either end of the QTP connection without being initiated by a Status Request. In this case, the Status Report will not be acknowledged.
5.7 Data Transfer

Data is the only message type which does not result in any acknowledgement. The following sequences indicate data to and from the network.

Incoming Data:

Terminal          Access Device          Server

DATA                Data
----------------------->       --------------------------------->

Outgoing Data:

Terminal          Access Device          Server

DATA                Data
<-----------------------       <---------------------------------
8. Authors’ Addresses

Allan Cornish
INETCO Systems Limited
201-3773 Still Creek Ave.
Burnaby, B.C., V5C 4E2
Canada
Phone: +1 604-451-1567
Fax: +1 604-451-1565
Email: allan_cornish@inetco.com

Michael Cox
Alcatel Australia Pty. Ltd.
3/321 Exhibition St.
Melbourne, Victoria, 3000
Australia
Phone: +61 3-9296-5436
Fax: +61 3-9296-5401
Email: Michael.Cox@alcatel.com.au

Ian Palmer
Alcatel Australia Pty. Ltd.
3/321 Exhibition St.
Melbourne, Victoria, 3000
Australia
Phone: +61 3-9296-5434
Fax: +61 3-9296-5401
Email: Ian.Palmer@alcatel.com.au

Angus Telfer
INETCO Systems Limited
201-3773 Still Creek Ave.
Burnaby, B.C., V5C 4E2
Canada
Phone: +1 604-451-1567
Fax: +1 604-451-1565
Email: angus_telfer@inetco.com

Cliff Wignell
Ascend Communications, Inc.
Level 38, ANZ Tower, 55 Collins Street
Melbourne, Victoria 3000
Australia
Phone: +61 3-9656-7081
Fax: +61 3-9656-7005
Email: cwignell@ascend.com.au