

Gnutella

Mattias Jansson

fimblo@sUNET.se

February 1, 2004

Contents

1	Gnutella version 0.4	3
1.1	Connecting to another servent	4
1.2	Gnutella Descriptors	6
1.3	Finding other servents	9
1.4	Searching for resources	10
1.5	Downloading resources	13
1.6	Firewalled servents	15

1 Gnutella version 0.4

(Text/Binary, TCP)

Gnutella is an application which enables its users to share files over the Internet. Current versions of Gnutella and its clones^a.

- Every Gnutella application is both a client and a server. This means that there is no centralized “core”, and that it truly is a distributed application. A gnutella application is called a *Servent*.
- Using simple application level routing the servents can connect to the gnutella network.

^aFollowing is a list of Gnutella protocol implementations. This list is by no means complete. BearShare, LimeWire, ToadNode, Gnotella, Mactella, Gnu-cleus, Gnut, Gtk-Gnutella, NapShare, OpenCola, Hagelslag, and Cultiv8r

1.1 Connecting to another servent

The servent initially starts with a file containing other servents addresses and ports. The local servent then tries to connect to at least one of these remote servents.

When the local servent succeeds in opening a TCP session to a remote servent, it then proceeds to send the following in ASCII:

```
GNUTELLA CONNECT/<protocol version string>\n\n
```

Connecting to another servent (con't)

If the remote servent wishes to accept the connection, it must reply with the string:

```
GNUTELLA OK\n\n
```

Other responses indicate the remote servent's unwillingness to accept the connection.

If positive acknowledgement is given, the local servent can proceed with sending descriptors through the TCP session.

1.2 Gnutella Descriptors

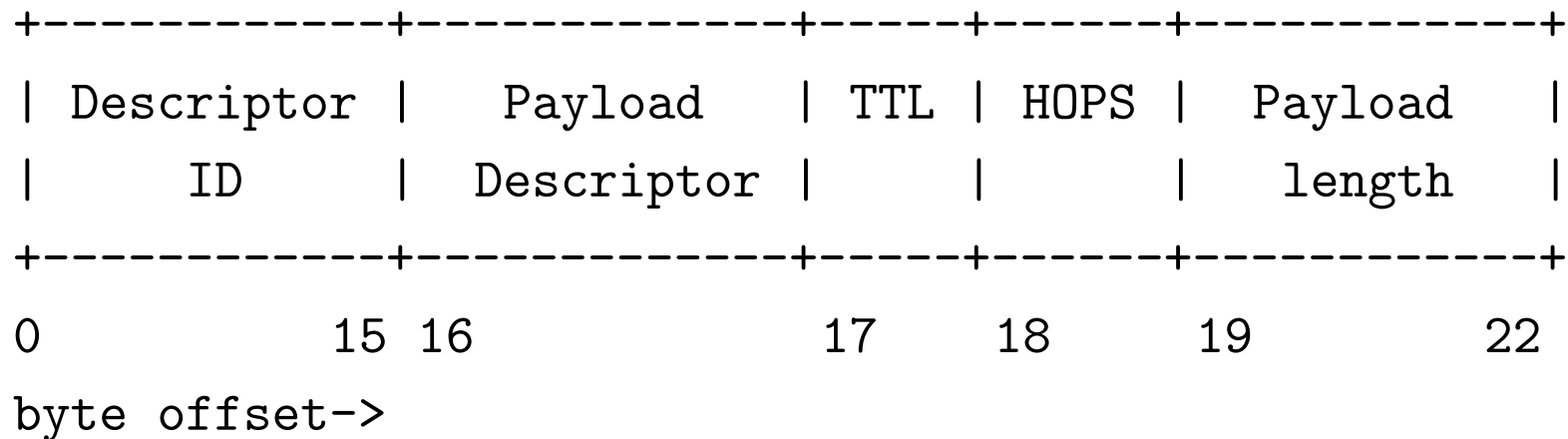
There are five descriptors in Gnutella, each with a specific function.

Descriptor	Description
Ping	Used to discover servents on the network.
Pong	The response to a Ping.
Query	Used to search the network for resources.
QueryHit	The response to a Query.
Push	Used to download resources from firewalled servents.

Gnutella Descriptors (con't)

Ping and Query descriptors are flooded onto the network, while Pong, QueryHit, and Push descriptors are sent back the path which they traveled to the responder.

All descriptors have a common header:



Descriptor Header (con't)

Descriptor ID	Uniquely identifies the descriptor.
Payload Desc.	0x00 = Ping 0x01 = Pong 0x40 = Push 0x80 = Query 0x81 = QueryHit
TTL	Time to live, decremented at each hop
HOPS	Incremented at each hop
Payload length	Length of payload

To make routing possible, the Descriptor ID should be identical between: Ping/Pong and Query/QueryHit.

1.3 Finding other servants

The local servant sends out a Ping descriptor to the connected servant. This servant and all following servants will flood the ping to its neighbours.

The Ping descriptor has no associated payload and are of zero length. If a remote servant elects to acknowledge this Ping, it will reply with a Pong. Below the Pong descriptor:

```
+-----+-----+-----+-----+
| Port  |      IP   | # of files | # of KB  |
|       |          | shared    | shared   |
+-----+-----+-----+-----+
 0      1 2           5 6           9 10       13
byte offset->
```

1.4 Searching for resources

The Query Descriptor is used to search for files on the gnutella network.

```
+-----+-----+
| Minimum Speed | Search Criteria |
+-----+-----+
  0             1 2             ...
byte offset->
```

Min Speed The Minimum bandwidth (kb/s) a server must fulfill to reply to this query.

Search Crit. A null (0x00) terminated search string. The max length of this field is bounded by the header's Payload Length.

Searching for resources (con't)

If a server on the network has data which matches the search criteria in a Query Descriptor, it may choose to send a QueryHit Descriptor in response.

Number of hits	Port	IP	Speed	Result set	Server Identifier
0	1	2 3	6 7	10 11	... n
byte offset->					

Nr of hits Number of hits.

Port TCP port of responding host

IP IP of responding host

Speed speed (kb/s) of responding host

Searching for resources (con't)

Result set A set of responses to the Query. This set contains one or more `Number_of_Hits` elements, each with the structure:

```
+-----+-----+-----+
| File Index | File Size | File Name |
+-----+-----+-----+
0           3 4           7 8           ...
```

File Index A number uniquely describing the file.

File Size The size (in bytes) of the file.

File Name The double-null (0x0000) terminated name of the file.

Servent ID A 16 byte string uniquely identifying the replying host. Instrumental in the operation of the Push Descriptor.

1.5 Downloading resources

Once a server receives a QueryHit descriptor, it may start to download the file from the target server. This download is always done out-of-network, i.e. a direct TCP connection is established between the local server and the target server.

The file download protocol is HTTP.

```
GET /get/<File Index>/<File Name>/ HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0-\r\n
User-Agent: Gnutella\r\n
\r\n
```

Downloading resources (con't)

Example HTTP query:

```
GET /get/2468/Foobar.mp3/ HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0-\r\n
User-Agent: Gnutella
\r\n
```

Example HTTP response:

```
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type: application/binary\r\n
Content-length: 4356789\r\n
\r\n
```

... data ...

1.6 Firewallled servents

If a direct connection cannot be established due to some reason (firewalls being a typical reason), the servent attempting the file download may send a Push Descriptor requesting that the servent sharing the file “push” the file instead.

Servent Identifier	File Index	IP	Port
0	15 16	19 20 23 24	25

byte offset->

Servent ID The ID of the firewallled servent.

File Index The index of the desired file.

IP The IP of the servent requesting the file.

Port The port number of the servent requestion the file.

Firewalled servents (con't)

Push Descriptors are routed using the Servent Identifier.

If a servent receives a push descriptor destined for itself, it may attempt to connect to the IP and port described in the Push descriptor, and send the following:

```
GIV <File Index>:<Servent Identifier>/<File Name>\n\n
```

At which the original requestor can send the usual GET command.