P2PInfo2

Status of this Memo

This is a draft.

The memo is based on the Gnutella Protocol Specification 0.6 (RFC Draft)

Copyright Notice

Rights of Free Implementation

The authors of the various proposals that make up this document
grant the rights to anyone to freely implement those proposals.
Info2 is an Open Protocol, where the specifications are
public and free of any patent.

1 Introduction

1.1 Purpose

P2PInfo2 is a decentralized peer-to-peer system. It allows the
participants to communicate from their system to all other participants of
the network.

Each participant launches a P2PInfo2 program, which will seek out other P2PInfo2 nodes to connect to. This set of connected nodes carries the P2PInfo2 traffic, which is essentially made of queries, replies to those queries, and also other control messages to facilitate the discovery of other nodes.

Users interact with the nodes by sending messages.

This document is intended for readers with a fair knowledge of network programming, but do not require any previous P2PInfo2 experience. Still, other implementations of this protocol will give useful information about implementation techniques that is not included in this document.

1.2 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 [34].

1.2.1 The P2PInfo2 Development Forum

The P2PInfo2 Development Forum is a good place to find more P2PInfo2 documentation, proposals about changes and extensions and to discuss P2PInfo2 development with other developers.

1.3 Terminology

Servent A program participating in the P2PInfo2 network is

called a servent. The words "peer", "node" and "host" have similar meanings, but refers to a network participant rather than a program. When a servent have a clear client or server role the words "client" or "server" may be used. The word "client" is sometimes used as a synonym for servent. This is a contraction of "SERVer" and "cliENT", Some other documents use the word "servant" instead of servent.

Message Messages are the entity in which information is transmitted over the network. Sometimes the word "packet" is used with the same meaning. Some other documents use the word "descriptor"

MID Message IDentifier. This is a 6-byte long value made of random bytes, whose purpose it is to identify messages. This identification is not a signature, just a way to identify network messages in a unique manner.

1.4 Extending the protocol

This document is the definition of the P2PInfo2 0.1 protocol. Servents MAY extend the protocol or even change parts of it (for example by compressing or encrypting the messages), but servents MUST always stay compatible with servents that follow this specification.

If a servent, for example, wants to compress the P2PInfo2 messages, it MUST first make sure the remote host of a connection can

decompress the stream (during handshake), and otherwise leave the messages uncompressed. Servents MAY chose not to accept a connection with a servent that does not support a feature, but MUST always make sure that the P2PInfo2 network is not split into separate networks.

Separate networks for special purposes are, of course, allowed but then it is no longer the P2PInfo2 network, but another network.

This P2PInfo2 also allows for extensions inside many messages. Such extensions can pass through servents that do not know about the extension to reach servents that do.

2 Protocol Definition

The P2PInfo2 protocol defines the way in which servents communicate over the network. It consists of a set of messages used for communicating data between servents and a set of rules governing the inter-servent exchange of messages. Currently, the following messages are defined:

SEND

Used to send a text message on the network

BACKCONNECT

Used to request a connection back from a servent to the requesting client.

FRIEND REQUEST

The primary mechanism for searching further nodes in the distributed network. A servent receiving a the FRIEND REQUEST message will respond with a IP address of a connected client to this P2PInfo2 servent.

LOGOFF

A message used to inform the remote host that you are closing the connection.

2.1 Initiating a Connection

A P2PInfo2 servent connects itself to the network by establishing a connection with another servent currently on the network.
Once the first connection is established, the addresses of more hosts connected to this servent can be requested over the network. The default P2PInfo2 port is 26000, and servents MUST NOT use any other unused port.

Once the address of another servent on the network is obtained, a TCP/IP connection to the servent is created, and a handshaking sequence is initiated. The client is the host initiating the connection and the server is the host receiving it.

1. The client establishes a TCP connection with the server.
2. The client sends "INFO2 CONNECT/0.6\n\n".
3. The server responds with "INFO2 OK\n\n".

Here is a sample interaction between a client and a server. Data sent from client to server is shown on the left; data sent from server to client is shown on the right.

```
Client                      Server
-----------------------------------------------------------
INFO2 CONNECT/0.6\n\n

                            INFO2 OK\n\n

[text messages]             [text messages]
```

A few notes about the responses: first, the client (server) SHOULD
disconnect if receiving any response other than "OK" at step (3). Second,
servents SHOULD NOT ignore higher version numbers in steps (3).
For example, it is perfectly legal for a future client to
reject connecting to a server using an older version of the protocol.

2.2 P2PInfo2 Messages

Once a servent has connected successfully to the network, it
communicates with other servents by sending and receiving P2PInfo2
protocol messages.
Each message is starts with a keyword following optional further information.
Each message consists of a string send from a client to the connected servents.

2.2.2 SEND

The SEND messages SHOULD start with the keyword SEND and SHOULD contain a 6-byte long Message IDentifier (MID) which MAY be randomly assigned by the client to uniqly mark a text message send throughout the network. The text message follows the keyword and the MID.
An example message is
SEND 123456 This is an example message

2.2.2.1 Use of SEND

A servent SHOULD print the SEND message text to the screen.
A servent SHOULD forward incoming SEND messages to all of its directly connected servents, except the one that delivered the incoming SEND. The MID is used to determine wether a SEND message was alread received and forwarded.

2.2.3 BACKCONNECT

BACKCONNECT messages SHOULD start with the keyword BACKCONNECT and SHOULD contain the client‘s IP address.

An example message is

BACKCONNECT 127.0.0.1

2.2.3.1 Use of BACKCONNECT

A servent SHOULD try to establish a back connection to the requesting servent.

2.2.4 FRIEND REQUEST

FRIEND REQUEST messages start with the keyword FRIEND REQUEST, contains no further Information but is ended with \n\n.

An example message is

FRIEND REQUEST\n\n

2.2.3.1 Use of FRIEND REQUEST

A servent SHOULD send the IP address of a connected servent to the requesting servent.

An example reply message is

127.0.0.1

2.2.5 LOGOFF

The LOGOFF messages SHOULD start with the keyword LOGOFF, SHOULD contain a 6-byte long Message IDentifier (MID) which MAY be randomly assigned by the client to uniqly mark a LOGOFF message send throughout the network and is followed by the servent‘s IP address.

An example message is

LOGOFF 123456 127.0.0.1

2.2.5.1 Use of LOGOFF

A servent SHOULD forward incoming LOGOFF messages to all of its directly connected servents, except the one that delivered the incoming LOGOFF. The MID is used to determine wether a SEND message was alread received and forwarded.

A servent SHOULD close the connection to the requesting servent if such a connection is

present.

3 Protocol Usage

Apart from the protocol definition in section 2, there are also some guidelines on how to use the protocol. These are not absolutely necessary to participate in the network, but very important for an effective network.

3.1 Flow Control

It is very important that all servents have a system for regulating the data that passes through a connection.
P2PInfo2 does not have flow control guidlines.

3.2 Network Structure

P2PInfo2 nodes are connected to each other randomly.
It worked fine for users with broadband connections, but not for users with slow connection. That problem can be solved by organizing the network in a more structured form.
P2PInfo2 does not have more advanced network structure guidlines.

3.3 Firewalled servents

It is not always possible to establish a direct connection to a P2PInfo2 servent in an attempt to initiate contact. The servent may, for example, be behind a firewall that does not permit incoming connections to its P2PInfo2 port. If a direct connection cannot be established, the servent cannot participate in the P2PInfo2 network.

4 Security Considerations

[TODO: This section is very incomplete. Any suggestions are welcome.]

5.1 Threats against individual P2PInfo2 participants

Inexperienced users might share sensitive information, such as cookie or password files, on the P2PInfo2 network. Servents SHOULD warn users who try share such information.

Attackers might try to reveal the netowkr structure by using FRIEND REQUEST to map the entire P2PInfo2 network.

Attackers might try to reveal the identity of a message origin by using FRIEND REQUEST to map the P2PInfo2 network and evaluate the time differences of receiving message times.

[TODO: Write about threats against individual P2PInfo2 participants Such as flooding, fake files, DoS, etc. Flooding host cache with faked friend request]
[TODO: How one can protect oneself and other P2PInfo2 users]

5.2 Threats against the P2PInfo2 network

[TODO: Write about threats against the P2PInfo2 network
Such as query flooding, fake files, DoS, fake random friend request etc.]
[TODO: What a servent should do to protect the network]
[TODO: A solution is fair sharing of bandwidth between connections and message types]

5.3 Threats against third parties

[TODO: Write about threats against third parties. Such (D)DoS, etc.]

[TODO: How to avoid]

Would it be possible to use the power of the P2PInfo2 network to attack internet hosts? That issue will be discussed in this section.

The ways of doing so an attacker might try is:

* Responding to FRIEND REQUEST messages with messages containing the IP address and port of a target host. This would cause other P2PInfo2 servents to attempt to connect to the target host, thinking it is a P2PInfo2 host.
  [TODO: Can this really be an effective attack? Would the target receive that many connection attempts?]