A Course on Foundations of Peer-to-Peer Systems & Applications

CS 6/75995
Foundation of Peer-to-Peer Applications & Systems

Kent State University
Dept. of Computer Science
www.cs.kent.edu/~javed/class-P2P08/LECT-7
## Classification

[*Eberspacher & Schollmeier 2005*]

<table>
<thead>
<tr>
<th><strong>Client-Server</strong></th>
<th><strong>Peer-to-Peer</strong></th>
<th><strong>Unstructured P2P</strong></th>
<th><strong>Structured P2P</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Server is the central entity and only provider of service and content.</td>
<td>1. Resources are shared between the peers.</td>
<td>1. All features of Peer-to-Peer included.</td>
<td>1. All features of Peer-to-Peer included.</td>
</tr>
<tr>
<td>2. Server is the higher performance system.</td>
<td>2. Resources can be accessed directly from other peers.</td>
<td>2. Central entity is necessary to provide the service.</td>
<td>2. Any terminal entity can be removed without loss of functionality.</td>
</tr>
<tr>
<td>3. Clients as the lower performance system.</td>
<td>3. Peer is provider and requestor (Servent concept).</td>
<td>3. Central entity is some kind of index/group database.</td>
<td>3. Dynamic central entities.</td>
</tr>
<tr>
<td>Example: WWW</td>
<td></td>
<td>Example: Gnutella 0.4, JXTA</td>
<td>Example: Gnutella 0.6, JXTA</td>
</tr>
</tbody>
</table>

### Unstructured P2P
- All features of Peer-to-Peer included.
- Any terminal entity can be removed without loss of functionality.
- Example: Gnutella 0.4, JXTA.

### Structured P2P
- All features of Peer-to-Peer included.
- Any terminal entity can be removed without loss of functionality.
- Example: Gnutella 0.6, JXTA.

### DHT-Based
- No central entities.
- Example: Chord, CAN.
Discussion on Centralized Systems

- Disadvantages
  - Single Point of Failure \(\rightarrow\) easily attackable
  - Bottleneck
  - Potential of congestion
  - Central server in control of all peers
- Advantages
  - Fast and complete lookup (one hop lookup)
  - Central managing/trust authority
  - No keep alive necessary, beyond content updates
- Application areas
  - VoIP (SIP, H.323)
  - Auctioning (Ebay)

Overview

1. General Characteristics of Early Peer-to-Peer Systems
2. Centralized Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
3. Pure Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
4. Hybrid Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
Definition of Pure P2P

- Any terminal entity can be removed without loss of functionality
- No central entities employed in the overlay
- Peers establish connections between each other randomly
  - To route request and response messages
  - To insert request messages into the overlay

Model of Pure P2P Networks

Degree distribution:

\[ p(d) = \begin{cases} 
    c \cdot d^{-\gamma}, & 0 < d \leq 7 \\
    0, & \text{in any other case} 
\end{cases} \]

\[ e = \left( \sum_{d} \frac{p(d)}{e} \right)^{-1} \]

Average: \( \bar{d} = 2.2 \)

Variance: \( \text{var}(d) = 1.63 \)

According Sample Graph:

- Separate sub networks
- Major component
Basic Characteristics of Pure P2P

- Bootstrapping:
  - Via bootstrap-server (host list from a web server)
  - Via peer-cache (from previous sessions)
  - Via well-known host
  - No registration

- Routing:
  - Completely decentralized
  - Reactive protocol: routes to content providers are only established on demand, no content announcements
  - Requests: flooding (limited by TTL and GUID)
  - Responses: routed (Backward routing with help of GUID)

- Signaling connections (stable, as long as neighbors do not change):
  - Based on TCP
  - Keep-alive
  - Content search

- Content transfer connections (temporary):
  - Based on HTTP
  - Out of band transmission

Topology of Pure P2P
Example:
Gnutella 0.4

Gnutella 0.4

- Program for sharing files over the Internet
- Focus: decentralized method of searching for files
- A “disruptive” application/technology?

- A decade of existence on March 14, 2010 In late 2007, it was the most popular file sharing network on the Internet with an estimated market share of more than 40%.

- Brief History:
  - **March 2000**: open source release by by Justin Frankel and Tom Pepper of Nullsoft, a division of AOL, and almost immediately withdrawn because of legal concern.
  - This did not stop Gnutella; after a few days, the protocol had been reverse engineered, and compatible free and open-source clones began to appear.
  - Still the third most popular filesharing system (after Bitorrent & FastTrack), 2008.
  - **Spring 2001**: further developments to improve scalability → Gnutella 0.6 (Hybrid P2P)
  - Since then:
    - available in a lot of implementations (Limewire, bearshare, …)
    - Developed further on (privacy, scalability, performance, …)
The Gnutella Network

Measurements taken at the LKN in May 2002

Gnutella: How Does It Work

- Application-level, peer-to-peer protocol over point-to-point TCP
- Participants:
  - Gnutella peers/servents
  - Router Service
    - Flood incoming requests (regard TTL)
    - Keep alive
    - Content
    - Route responses for other peers (regard GUID of message)
    - Keep alive (PING/PONG)
    - Content (QUERY/QUERYHIT)
  - Data-requests
  - Download-requests
- Lookup Service
  - Initialize Data requests
  - Initialize keep alive requests
  - “Server”-Service
    - Serve Data-requests (HTTP)
- Five steps:
  - Connect to at least one active peer (address received from bootstrap)
  - Explore your neighborhood (PING/PONG)
  - Submit Query with a list of keywords to your neighbors (they forward it)
  - Select “best” of correct answers (which we receive after a while)
  - Connect to providing host/peer
Node Joining

- In order to join the system a new node/servent initially connects to one of several known hosts that are almost always available (e.g., gnutellahosts.com).
- Once attached to the network (e.g., having one or more open connections with nodes already in the network), nodes send messages to interact with each other.

PING & PONG

- *Used to Establish Group Membership.*
- A node joining the network initiates a broadcasted PING message to announce its presence.
- When a node receives a PING message it forwards it to its neighbors and initiates a back-propagated PONG message.
- The PONG message contains information about the node such as its IP address and the number and size of shared files.
QUERY & QUERY HIT

- *Used For Searching.*

- QUERY messages contain a user specified search string that each receiving node matches against locally stored file names. QUERY messages are broadcasted.

- QUERY RESPONSES are back-propagated replies to QUERY messages and include information necessary to download a file.

Gnutella Message Structure

**General Header Structure:**

<table>
<thead>
<tr>
<th>MESSAGEHEADER: 23Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>GnodeID 16 Bytes</td>
</tr>
</tbody>
</table>

- **GnodeID**: unique 128bit Id of any Hosts
- **TTL** (Time-To-Live): number of servents, a message may pass before it is killed
- **Hops**: number of servents a message already passed

Describes the message type (e.g. login, search,...)

Describes parameters of the message (e.g. IDs, keywords,...)
Gnutella Messages

**PING (Function: 0x00)**

No Payload

**PONG (Function: 0x01)**

<table>
<thead>
<tr>
<th>Bytes</th>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>Port Number</td>
<td>The port number on which the responding host can accept incoming connections.</td>
</tr>
<tr>
<td>2-5</td>
<td>IP Address</td>
<td>The IP address of the responding host. Note: This field is in big-endian format.</td>
</tr>
<tr>
<td>6-9</td>
<td>Number of shared files</td>
<td>The number of files that the servent with the given IP address and port is sharing on the network.</td>
</tr>
<tr>
<td>10-13</td>
<td>Number of kilobytes shared</td>
<td>The number of kilobytes of data that the servent with the given IP address and port is sharing on the network.</td>
</tr>
<tr>
<td>14-</td>
<td>GGEP block</td>
<td>OPTIONAL extension (see GGEP).</td>
</tr>
</tbody>
</table>

**QUERY (Function: 0x80)**

<table>
<thead>
<tr>
<th>Bytes</th>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Speed</td>
<td>2 Bytes</td>
<td></td>
</tr>
<tr>
<td>Search Criteria</td>
<td>n Bytes</td>
<td></td>
</tr>
</tbody>
</table>

**QUERY HIT (Function: 0x81)**

<table>
<thead>
<tr>
<th>Bytes</th>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nb. of Hits</td>
<td>1 Byte</td>
<td></td>
</tr>
<tr>
<td>Port</td>
<td>2 Bytes</td>
<td></td>
</tr>
<tr>
<td>IP Address</td>
<td>4 Bytes</td>
<td></td>
</tr>
<tr>
<td>Speed</td>
<td>1 Byte</td>
<td></td>
</tr>
<tr>
<td>Result Set</td>
<td>n Bytes</td>
<td></td>
</tr>
<tr>
<td>GnodeID</td>
<td>16 Bytes</td>
<td></td>
</tr>
<tr>
<td>File Index</td>
<td>4 Bytes</td>
<td></td>
</tr>
<tr>
<td>File Name</td>
<td>n Bytes</td>
<td></td>
</tr>
</tbody>
</table>
## QUERY MESSAGE

<table>
<thead>
<tr>
<th>Bytes</th>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>Minimum Speed (Flags)</td>
<td>The minimum speed (in kb/second) of servents that should respond to this message. A servent receiving a Query message with a Minimum Speed field of n kb/s SHOULD only respond with a Query Hit if it is able to communicate at a speed &gt;= n kb/s.</td>
</tr>
<tr>
<td>2-</td>
<td>Search Criteria</td>
<td>This field is terminated by a NUL (0x00). See section 2.2.7.3 for rules and information on how to Interpret the Search Criteria.</td>
</tr>
<tr>
<td>Rest</td>
<td>Extensions Block</td>
<td>OPTIONAL. The rest of the query message is used for extensions to the original query format. The allowed extension types are GGEP, HUGE and XML (see Section 2.3 and Appendices 1 and 2). If two or more of these extension types exist together, they are separated by a 0x1C (file separator) byte. Since GGEP blocks can contain 0x1C bytes, the GGEP block, if present, MUST be located after any HUGE and XML blocks. The type of each block can be determined by looking for the prefixes &quot;urn:&quot; for a HUGE block, &quot;&lt;&quot; or &quot;{&quot; for XML and 0xC3 for GGEP. The extension block SHOULD NOT be followed by a null (0x00) byte, but some servents wrongly do that.</td>
</tr>
</tbody>
</table>

### QUERY Flag/ Speed Flag (Actually G.06)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Flag Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>MinSpeed/Flags Indicator</td>
<td>MUST be set to 1 to indicate that the flags below are used instead of encoding the Minimum Speed.</td>
</tr>
<tr>
<td>14</td>
<td>Firewalled Indicator</td>
<td>The host who sent the query is unable to accept incoming connections. This flag can be used by the remote servent to avoid returning Query Hits if it is itself firewalled, as the requesting servent will not be able to download any files.</td>
</tr>
<tr>
<td>13</td>
<td>XML Metadata</td>
<td>Set this bit to 1 if you want the sharing servent to send XML Metadata in the Query Hit. This flag has been assigned to spare bandwidth, returning metadata in queryHits only if the requester asks for it. If this bit is not set, the sharing host MUST NOT send XML metadata in return Query Hit messages.</td>
</tr>
<tr>
<td>12</td>
<td>Leaf Guided Dynamic Query</td>
<td>When the bit is set to 1, this means that the query is sent by a leaf which wants to control the dynamic query mechanism. This is part of the Leaf guidance of dynamic queries proposal. This information is only used by the ultrapeers shielding this leave if they implement leaf guidance of dynamic queries. If this bit is set in a Query from a Leaf it indicates that the Leaf will respond to Vendor Messages from its Ultrapeer about the status of the search results for the Query.</td>
</tr>
<tr>
<td>11</td>
<td>GGEP &quot;H&quot; Allowed</td>
<td>If this bit is set to 1, then the sender is able to parse the GGEP &quot;H&quot; extension which is a replacement for the legacy HUGE GEM extension. This is meant to start replacing the GEM mechanism with GGEP extensions, as GEM extensions are now deprecated.</td>
</tr>
<tr>
<td>10</td>
<td>OOB Query</td>
<td>This flag is used to recognize a Query which was sent using the Out Of Band Query extension.</td>
</tr>
<tr>
<td>9</td>
<td>?</td>
<td>Reserved for a future use. Must be set to 0.</td>
</tr>
<tr>
<td>0-8</td>
<td>Maximum Query Hits</td>
<td>Set when a maximum number of Query Hits is expected. 0 if no maximum. This does not mean that no more Query Hits may be returned, but that the query should be propagated in a way that will cause the specified number of hits.</td>
</tr>
</tbody>
</table>
### Gnutella Routing

- **Basic Routing Principle:** "Enhanced" Flooding
  - Save Origin of received PINGs and QUERIES
  - Decrease TTL by 1
  - If TTL equals 0, kill the message
- **Flooding:** Received PINGS and QUERIES must be forwarded to all connected Gnodes
- PINGS or QUERIES with the same FUNCTION ID and GNODE ID as previous messages are destroyed (avoid loops)
- **PONG and QUERY HIT are forwarded to the origin of the corresponding PING or QUERY**

### Gnutella Connection Setup

Gnode 2000 establishes a connection to 4000

<table>
<thead>
<tr>
<th>Gnode ID</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>001</td>
</tr>
<tr>
<td>2000</td>
<td>002</td>
</tr>
<tr>
<td>3000</td>
<td>003</td>
</tr>
<tr>
<td>4000</td>
<td>004</td>
</tr>
</tbody>
</table>

- **17** Gnutella Connect
- **18** Gnutella OK
- **19** PING
- **20** PONG/IP: 004
- **21** PING
- **23** PONG/IP: 001
- **27** PONG/IP: 001
- **22** PING
- **24** PONG/IP: 003
- **28** PONG/IP: 003
- **25** PING
- **26** PING
Gnutella Wrap-Up

- 1. Query-Hit
- Broadcast
Query[XYZ,TTL=2,…]

Messaging Overhead

- PING 23 bytes
- PONG 37 bytes
- GnuCon+OK 34 bytes
- Huge overhead.
- Non Optimum Physical/Overlay mapping
Discussion

- Disadvantages
  - High signaling traffic, because of decentralization
  - Modern nodes may become bottlenecks
  - Overlay topology not optimal, as
    - no complete view available,
    - no coordinator
  - If not adapted to physical structure delay and total network load increases
    - Zigzag routes
    - loops
- Advantages
  - No single point of failure
  - Can be adapted to physical network
  - Can provide anonymity
  - Can be adapted to special interest groups
- Application areas
  - File-sharing
  - Context based routing (see chapter about mobility)
Overview

1. General Characteristics of Early Peer-to-Peer Systems
2. Centralized Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
3. Pure Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion
4. Hybrid Peer-to-Peer Networks
   1. Basic Characteristics
   2. Signaling Characteristics
   3. Discussion