

4. Unstructured Peer-to-Peer Networks

4.1 An Introduction to Peer-to-Peer Networks

4.2 Napster

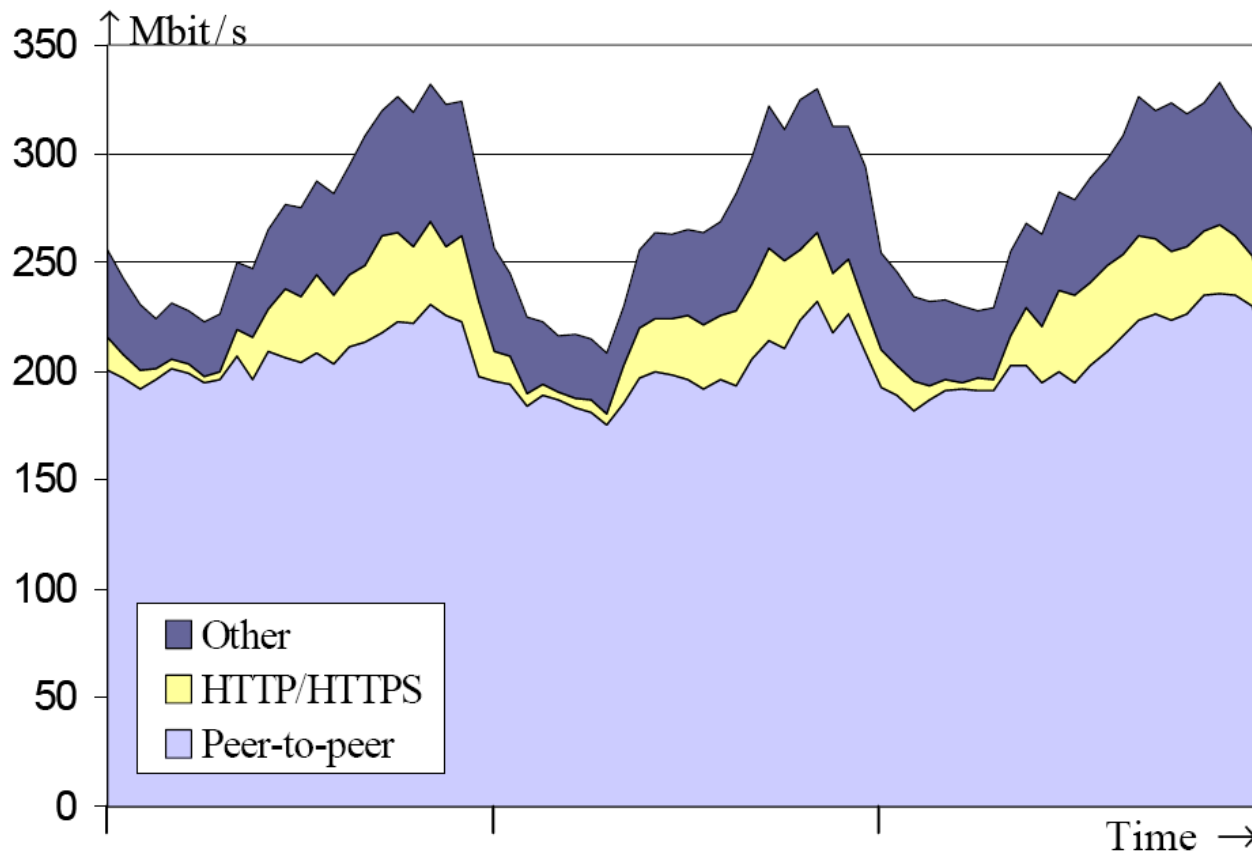
4.3 Gnutella

4.4 BitTorrent

4.1 An Introduction to Peer-to-Peer Networks

The amount of IP Traffic from Peer-to-Peer Networks

Traffic profile of the Deutsche Telekom Access Network (2nd quarter 2003)



Problems of Today's Internet

- Scalability
- Flexibility / extensibility
- Security / reliability
- and more...

Does the Peer-to-Peer principle offer adequate solutions?

Problems of Today's Internet: Scalability (1)

Scalability

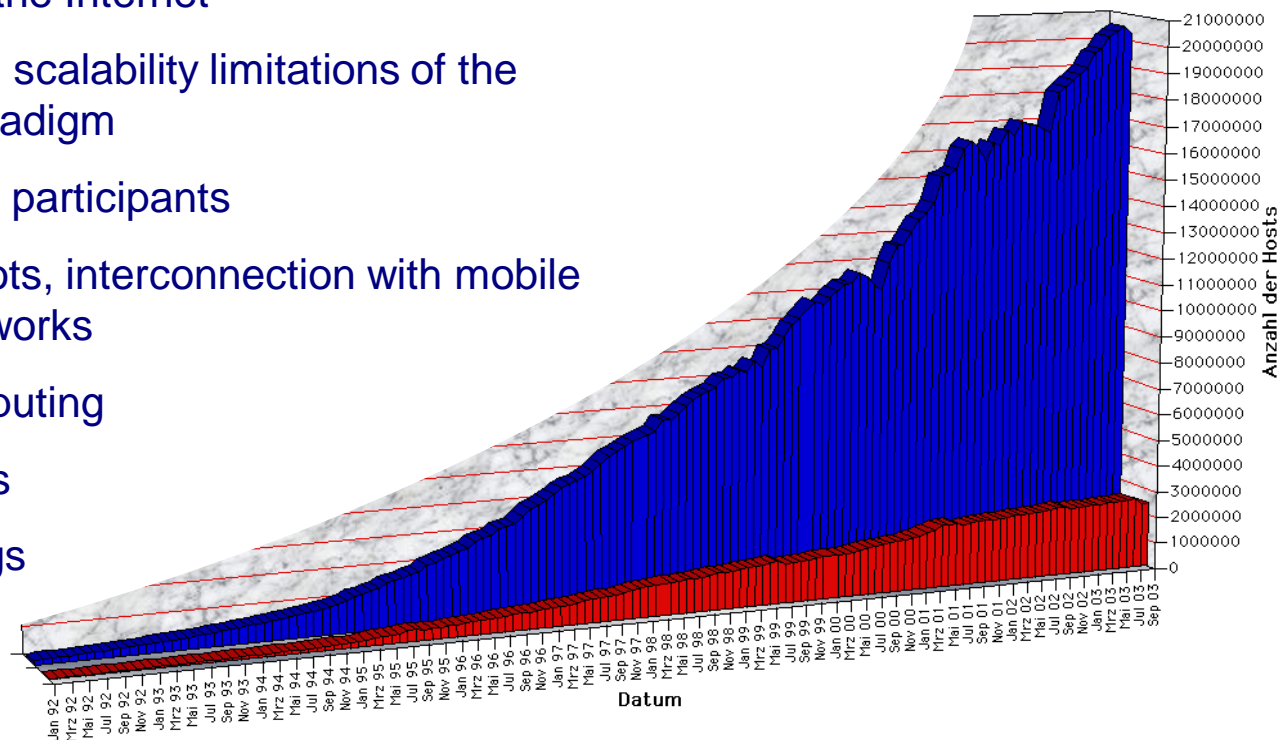
The capability of a system to keep functioning and working efficiently while growing by orders of magnitude

Enormous growth of the Internet

- Seems to reveal scalability limitations of the client-server paradigm

Increasing number of participants

- Wireless hot spots, interconnection with mobile and cellular networks
- Ubiquitous computing
- Sensor networks
- Internet of Things



Problems of Today's Internet: Scalability (2)

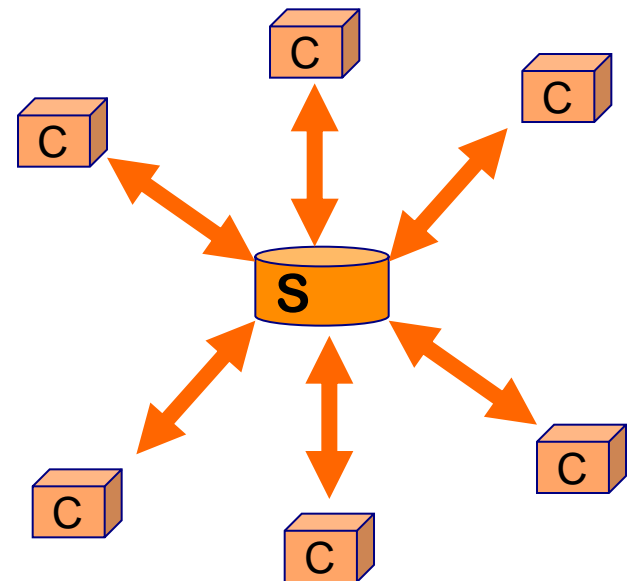
General problems of client-server architectures

Limited resources of and around the server

- Increasing traffic load
 - Highly concentrated on server – under-load at other parts of the network
 - Asymmetrical traffic flows
- Problems of resource scalability
 - memory, CPU
- Server poses problem of „single point of failure“

Unused resources at many clients

- CPU
- Memory
- Information



Problems of Today's Internet: Scalability (3)

Client-server systems do not scale infinitely!

- Limited resources

New applications impose increasing resource requirements

- Bandwidth, memory (e.g., file exchange)
- CPU power (e.g., Seti@Home)

The requirements cannot be fulfilled by centralized architectures anymore.

- Examples (year 2002): Kazaa (10^7 Gbyte), Seti@Home (20 GigaFlops)

Driving Forces Behind Peer-to-Peer Networks (1)

Development of end system capabilities

1992:

- Average hard disk size: ~0.3 Gbyte
- Average processing power (clock frequency) of personal computers: ~ 100 MHz

2002:

- Average hard disk size: 100 Gbyte

2004:

- Average processing power (clock frequency) of personal computers: ~ 3 GHz
→ Personal computers now have the capabilities comparable to servers in the 1990s

Driving Forces Behind Peer-to-Peer Networks (2)

Development of communication networks

Early 1990s: private users start to connect to the Internet via 56 kbps modems

1997/1998:

- first broadband connections for residential users become available
- cable modem with up to 10 Mbps

1999:

- Introduction of DSL and ADSL connections
- Data rates of up to 8.5 Mbps via common telephone connections become available.
- The deregulation of the telephone market shows first effects with significantly reduced tariffs, due to increased competition on the last mile.
→ **bandwidth is plentiful and cheap!**

Problems of Today's Internet: Flexibility

Missing flexibility and extensibility of the network

- Limitations for the introduction of new services
- No specialized services available

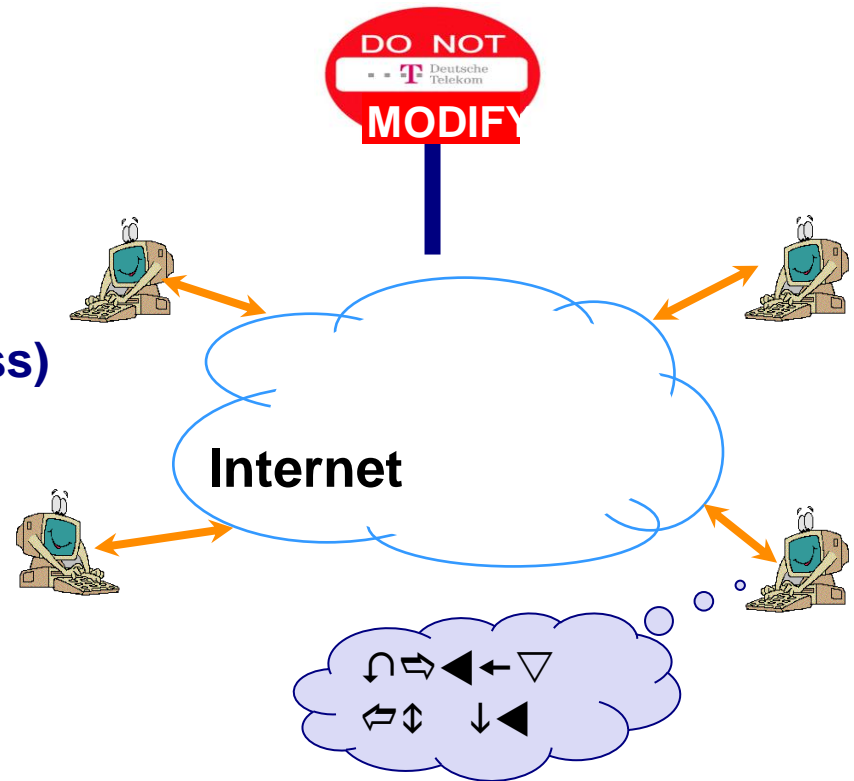
ISPs are shy at changes

- High cost, low earnings
- Limited control, possible instability

Many developments fail (more or less)

Examples:

- Group communication, IP multicast
- Quality of Service in the Internet
- Support for mobility
- Active Networking



Problems of Today's Internet: Security/Reliability (1)

Availability will be increasingly important for future Internet-based systems

Very high cost in case of failing systems, servers, services

Increasing number of intentional attacks

- Distributed denial-of-service attacks (DDoS)
- Central servers are easy targets

100% fault tolerance

- ... impossible
- Single-vendor platforms are more susceptible to attacks
 - e.g., MS Windows bugs, MS Windows viruses



"Didn't you get my e-mail?"

Problems of Today's Internet: Security/Reliability (2)

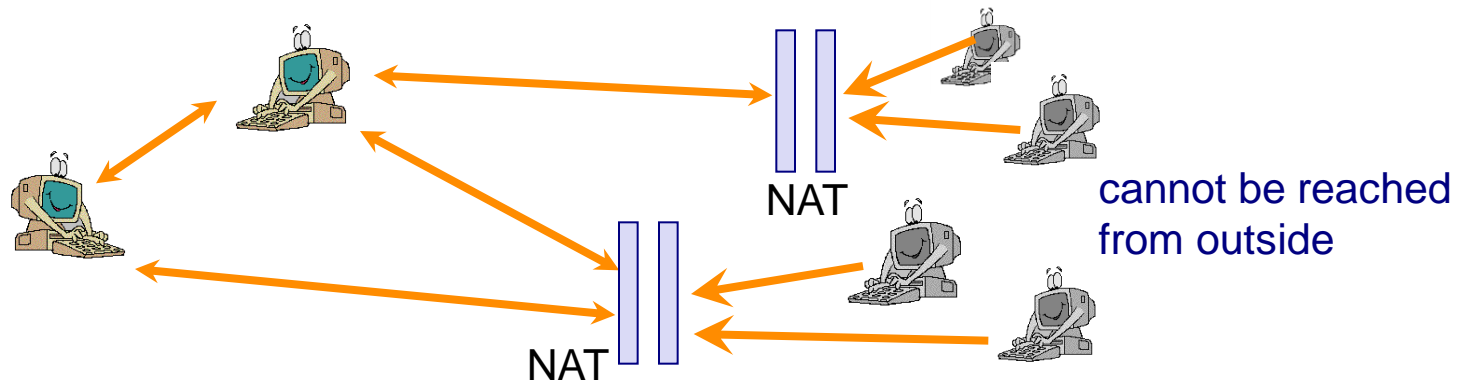
Resistance to censorship

- Great demand from users' point of view
- „Small demand“ from the „agencies“ point of view
- Central server systems can be easily shut down or rendered harmless

Departure from the End-to-End Principle

Further problems due to the evolution of the Internet

- Network Address Translation (NAT) / dynamic addresses
 - Shortage of Internet addresses / IPv6 no big issue anymore
 - Firewall-based protection of intra networks
 - Various forms of „middleboxes“ (proxies, ...)
- No general end-to-end connections anymore
 - Departure from original Peer-to-Peer characteristics
 - Systems „behind“ NAT have no chance to offer public services!



Result of the Developments: De-Centralization

Conclusion

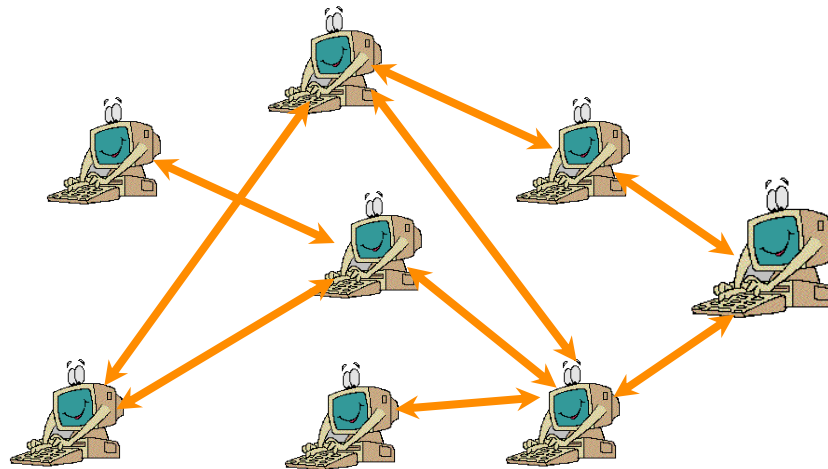
- Centralized systems
 - ... do not scale infinitely
 - ... entail single points of failures (reliability problems)
 - ... are easy to attack
 - ... but are easy to realize
 - and will thus continue to be used (up to the scalability limit)

The way out

- De-Centralization
 - Re-emergence of the Peer-to-Peer principle
 - „Back to the roots“ (but on a different scale than before – *challenge!*)

Definition of Peer-to-Peer Systems (1)

What are Peer-to-Peer Systems?



A first definition

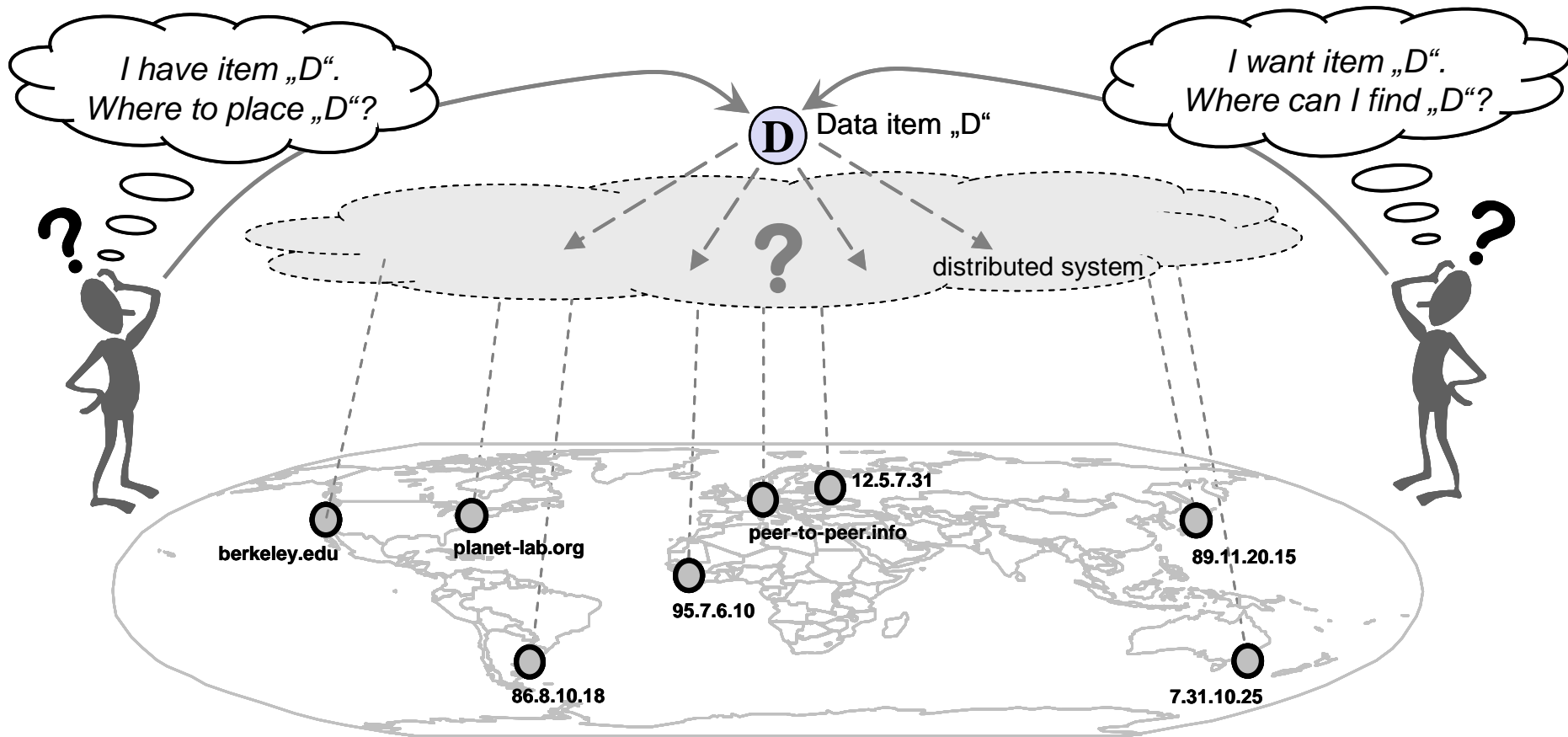
P2P is a class of applications where each node is at the same time a client and a server. Because they operate in an environment of unstable connectivity and un-predictable IP addresses, P2P nodes must operate outside the DNS system and have significant or total autonomy from central servers.

Definition of Peer-to-Peer Systems (2)

Characteristics of Peer-to-Peer systems

- Direct interaction of end systems (peers)
- Joint usage of resources in end systems
- No central control or usage of central services
- Equal and autonomous participants
- Self-organization of the system

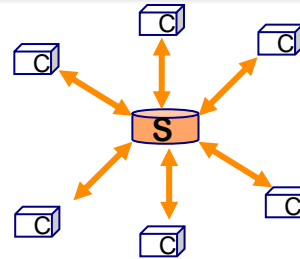
Information Management is a Basic Challenge



Types of Peer-to-Peer Systems

Client-Server systems

- Classical role-based systems
- No direct interaction between clients



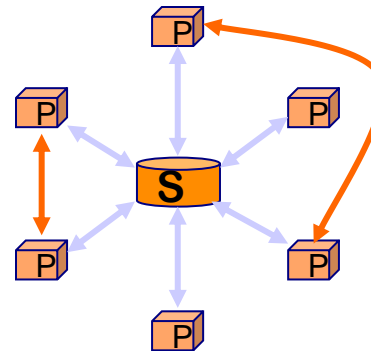
Examples

WWW

DNS

Hybrid P2P systems

- Joint usage of distributed resources
- Usage of servers for coordination
- Direct interaction between peers for data exchange



Napster

ICQ

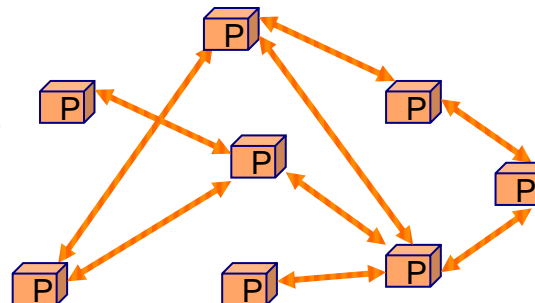
AIM (AOL)

Seti@Home

Skype

Pure P2P systems

- Completely de-centralized management *and* usage of the resources

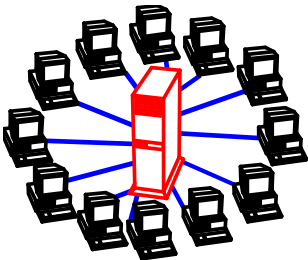
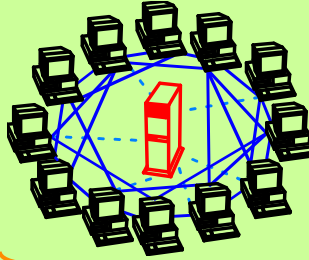
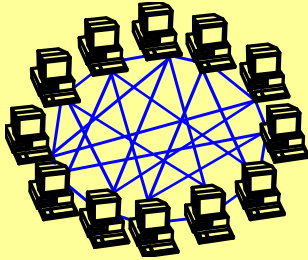
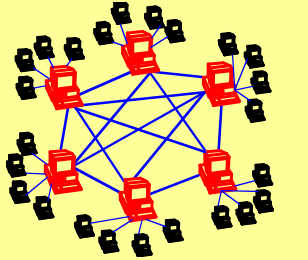
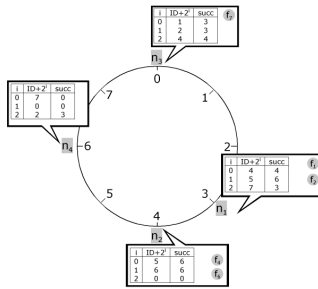


Gnutella

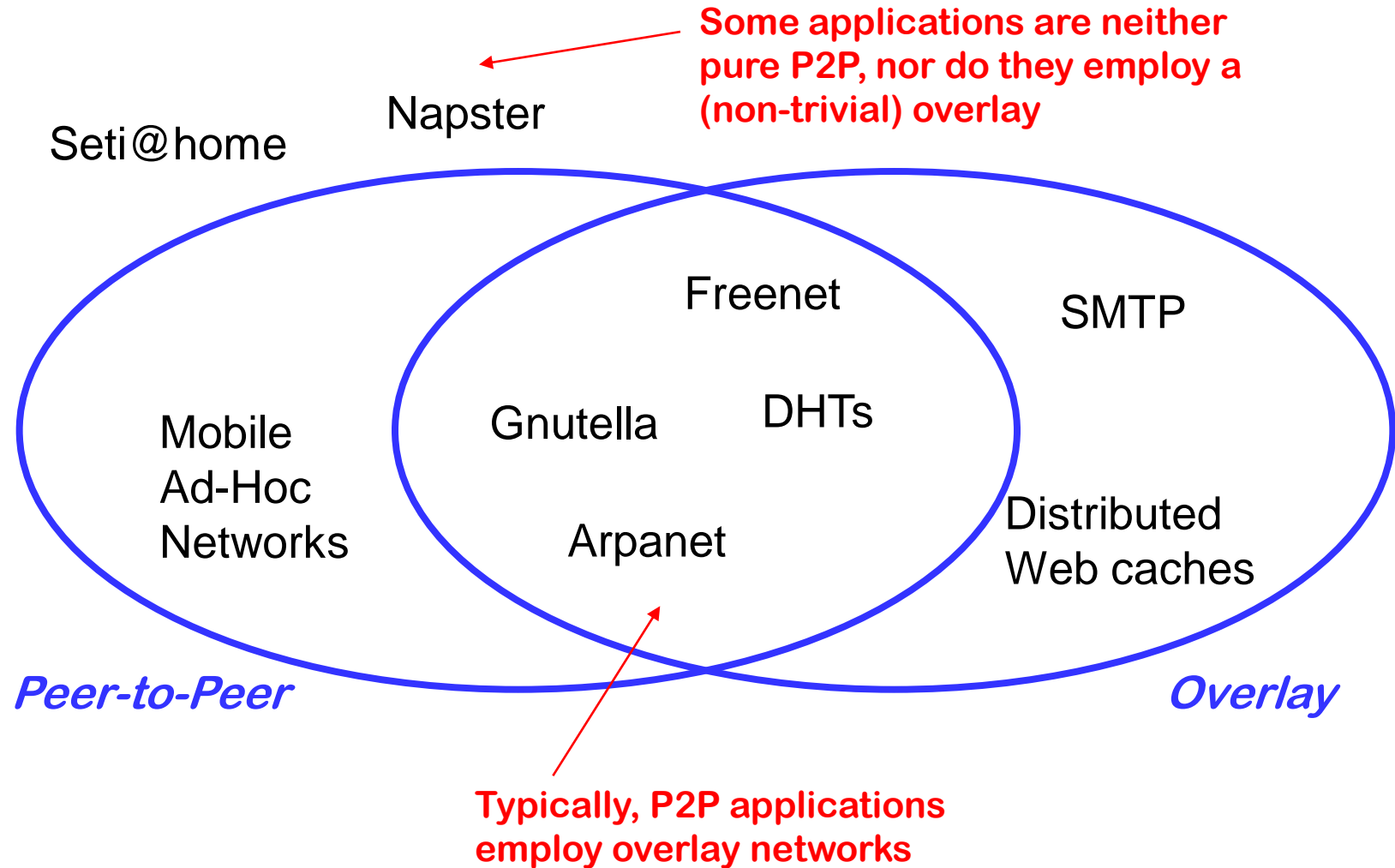
Freenet

DHT-based systems

Architectures of Peer-to-Peer Networks

Client-Server	Peer-to-Peer			
<ol style="list-style-type: none"> 1. Server is the central entity and only provider of service and content. → Network managed by the Server 2. Server as the higher performance system. 3. Clients as the lower performance systems <p>Example: WWW</p>	<ol style="list-style-type: none"> 1. Resources are shared between the peers 2. Resources can be accessed directly from other peers 3. Peer is provider (server) and requestor (client): the <i>servent</i> concept 			
	Unstructured P2P			Structured P2P
	Centralized P2P	Pure P2P	Hybrid P2P	DHT-Based
	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Central entity is necessary to provide the service 3. Central entity is some kind of index/group database <p>Example: Napster</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → No central entities <p>Examples: Gnutella 0.4, Freenet</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → dynamic central entities <p>Example: Gnutella 0.6, JXTA</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → No central entities 4. Connections in the overlay are “fixed” <p>Examples: Chord, CAN</p>
				
	1 st Gen.		2 nd Gen.	

Peer-to-Peer Networks vs. Overlay Networks



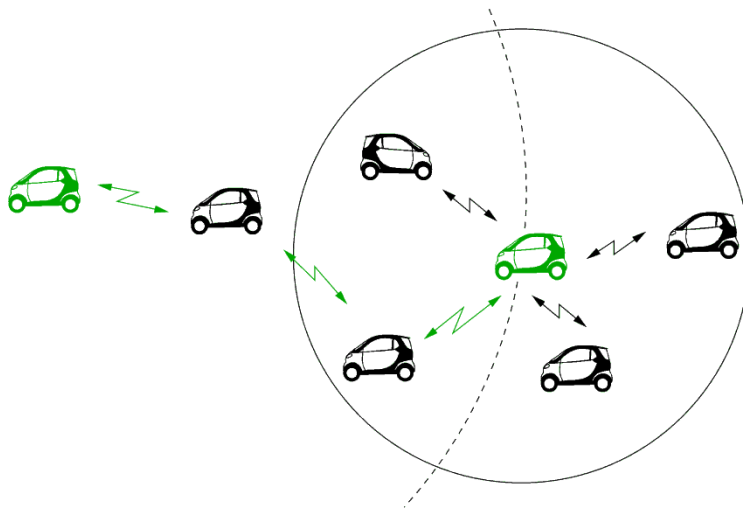
Examples: P2P vs. Overlay Networks

P2P, but no overlay:

Mobile Ad-Hoc Networks

- Each node is client and router
- No infrastructure

Sensor networks



Overlay, but not P2P ...

... Client/Server:

- Distributed Web caches

... without using distributed resources:

- Virtual private networks (VPN)

Both P2P and Overlay

- ARPANET
- Gnutella
- P2P systems based on Distributed Hash Tables (DHTs)

Conclusions

Peer-to-Peer networks

- De-centralized self-organizing systems with de-centralized usage of resources

Reasons for current usage of P2P techniques

- Today's Internet has many weaknesses
 - limited scalability, flexibility, extensibility, reliability/security
 - other deployment-specific „illnesses“ of the Internet: NAT, middleboxes, dynamic addresses ...

Additional reasons

- Success of “practical” P2P approaches (publicity through copyright discussions)
- Emerging of innovative types of services (ICQ, file sharing, Skype, etc.)
- Interesting research areas:
 - Quality in peer-to-peer networks
 - De-centralized self-organization
 - Location-based routing → content-based routing

4.2 Napster

General Characteristics of 1st and 2nd Generation P2P Networks

1st and 2nd Generation P2P systems are overlay architectures with the following characteristics:

- TCP/IP-based
- Decentralized and self organizing (with possible centralized elements)
- Content (mostly music and video files)
 - distributed “randomly” on the network, with several replicas
 - Content and its descriptions are not structured.
 - Content stays at the nodes which bring it into the network.
 - Content transfer:
 - Out-of-band, i.e., on separate connections and not via signaling connections
 - Most often via http
- Generally two kinds of requests:
 - Content requests: to find content in the overlay
 - Keep-alive requests: to stay connected in the overlay

Basic Topology Characteristics

Peer

- A node actively participating in the overlay
- Identifiable by a General Unique ID (hash-value of “unique” or random ID)

Overlay Topology

- “Virtual” signaling network established via TCP connections between peers

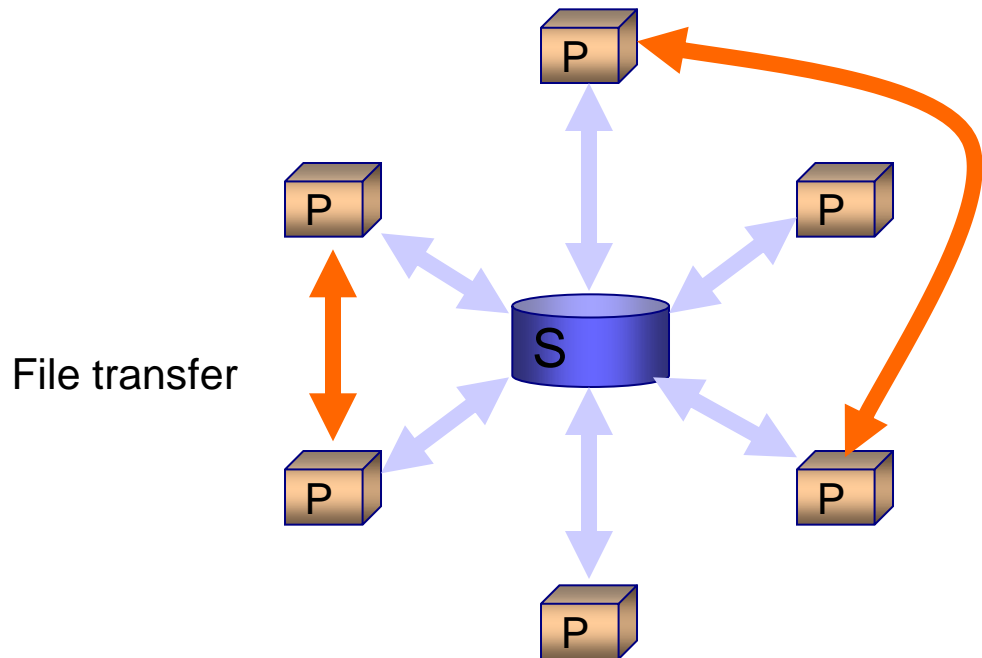
Characteristics of the overlay topology

- Completely independent of the topology of the physical network
- May include hierarchies (hub network)
- May include centralized elements (lookup server in Napster) (star network)
- May be a completely randomized network (Gnutella 0.4) which can be modeled with random graphs (randomly meshed network)
- Separate addressing scheme

Definition of Centralized P2P Networks

Basic Characteristics

- All peers are connected to the central entity that maintains a *directory*.
- Peers establish connections between each other case-by-case to exchange user data (e.g., mp3 audio data)
- The central entity is necessary to provide the service.



Basic Characteristics of Centralized P2P Networks

Bootstrapping

- Bootstrap server = the central server

Coordination

- The central entity can be established as a server farm, but with one single entry point (single point of failure).
- All signaling connections are directed to the central entity.

Peer talks to the central entity

- to find content
- to register and log on to the overlay
- to update the routing tables
- to update shared content information.

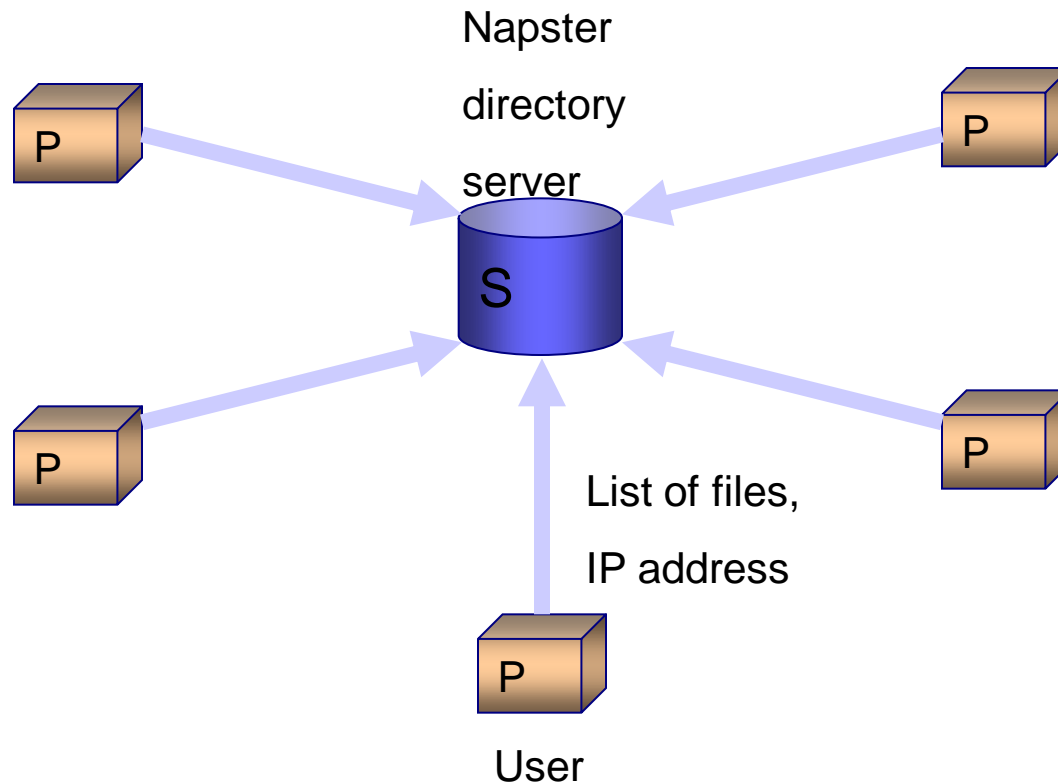
Peer talks to a peer via http

- to exchange content/data

Operation of Napster (1)

Connection setup to the directory server

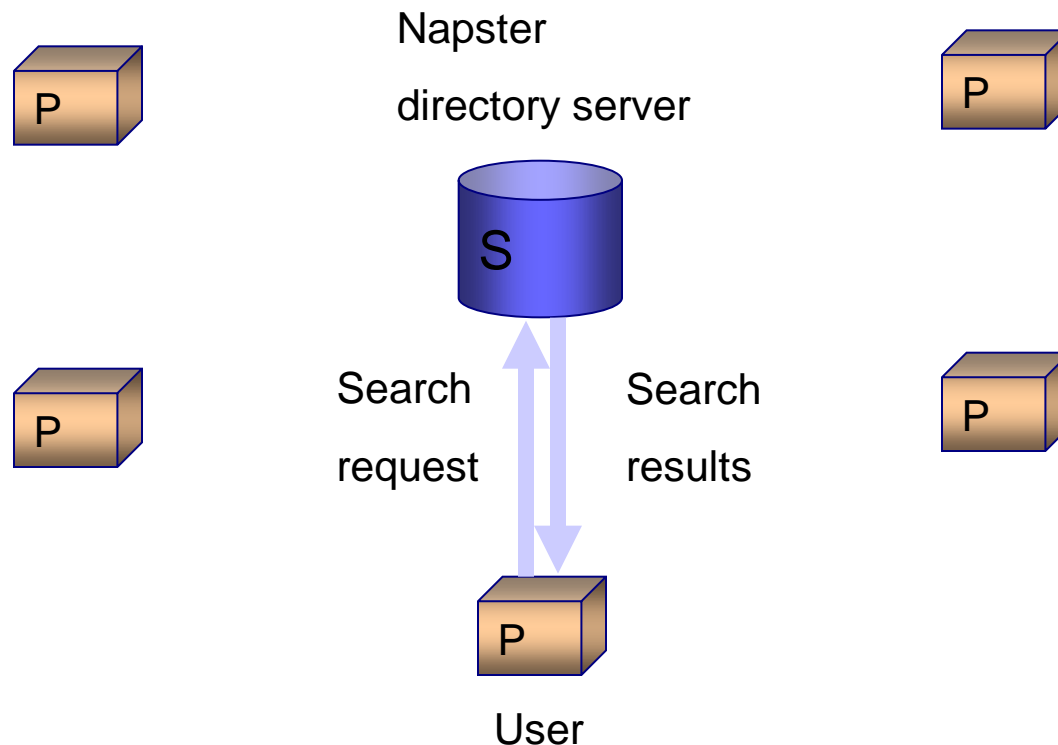
- Peers provide a list of their files (including metadata, MP3 tags) as well as their IP address to the server S



Operation of Napster (2)

Search for files

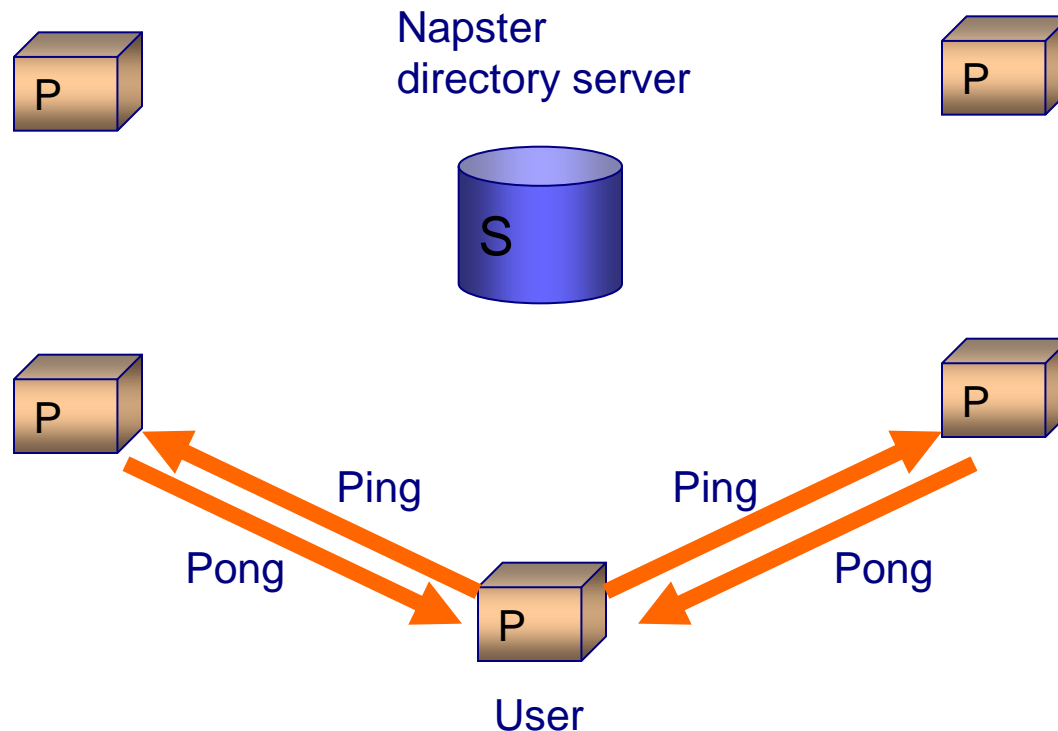
- The user issues a request to the directory server.
- The server searches through the data base, and if the search is successful it returns a list of IP addresses of peers where the data can be found.



Operation of Napster (3)

Testing of potential peers (target computers)

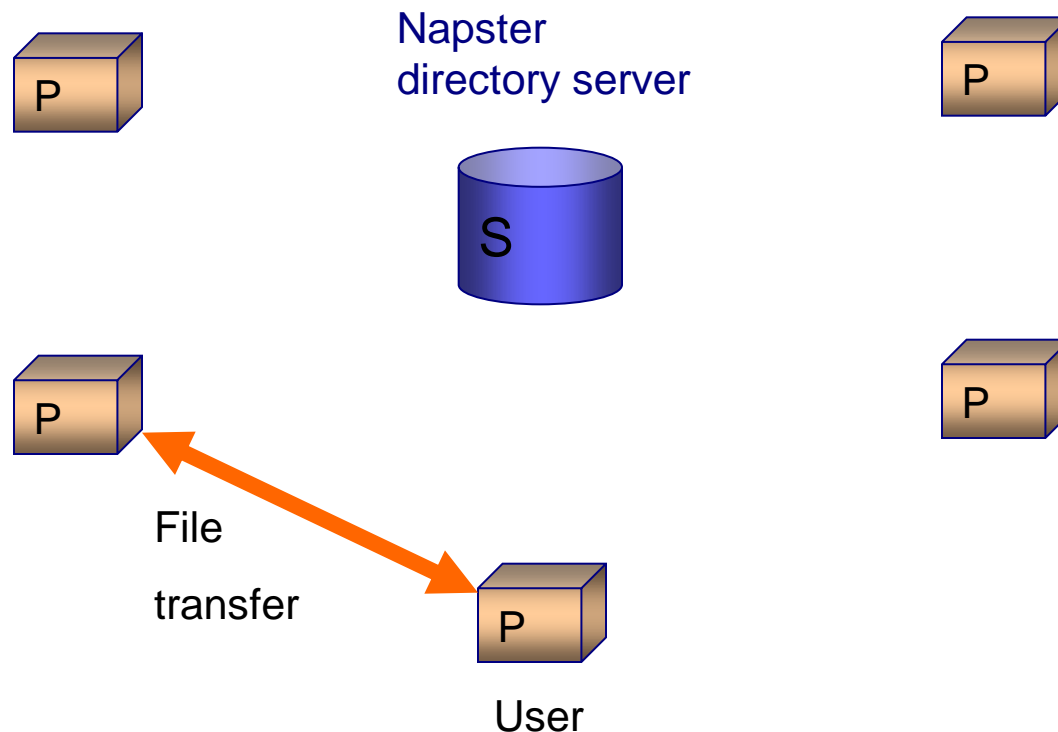
- The user tests the quality of the connections to the target computers by use of a “ping”.
- Available peers respond with “pong”.



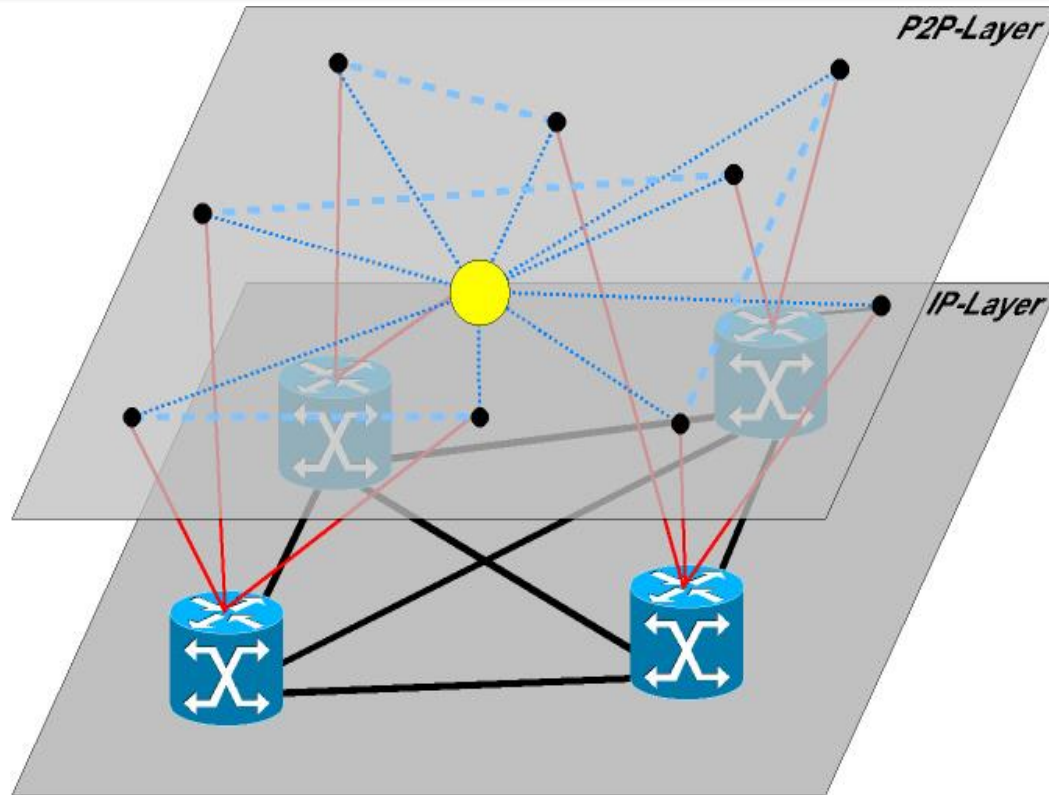
Operation of Napster (4)

Download

- The user sets up a direct connection with the peer that has the best quality.



Topology of Centralized P2P Networks



● Servent

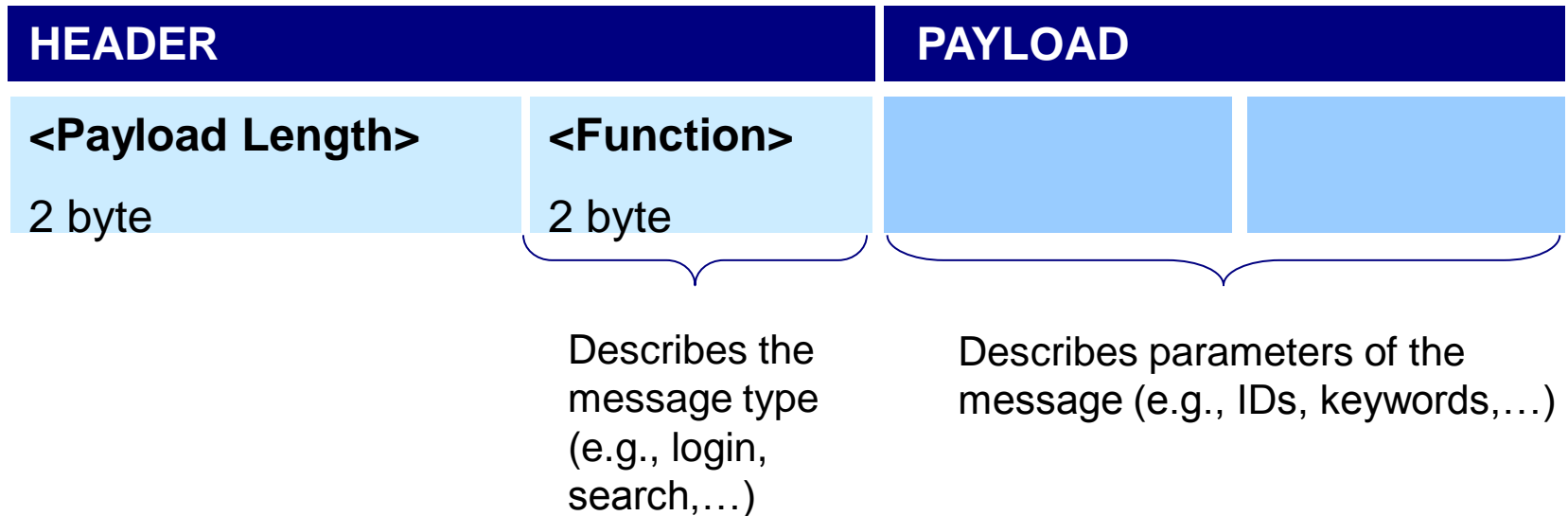
..... Connection between two servents (TCP)

— Connection between router and servent

— Connection between routers (core)

Napster Message Structure

General Header Structure



Napster: Initialization

Client/Server Service

1: LOGIN (Function:0x02)

<Nick> <Password> <Port> <Client-Info> <Link-type>

2: LOGIN ACK (Function: 0x03)

3: NOTIFICATION OF SHARED FILE (0x64)

„<Filename>“ <MD5> <Size> <Bitrate> <Freq> <Time>

Napster Host

IP: 001

Nick: LKN

NOTIFICATION(0x64)

„band-song.mp3“ 3f3a3... 5674544
128 44100 342

**Central
Napster
Index**

Napster: File Request Procedure

1: SEARCH (Function: 0xC8)

[FILENAME CONTAINS „Search Criteria“]

[MAX_RESULT <Max>]

[LINESPEED <Compare> <Link-Type>]

[BITRATE <Compare> “<Bitrate>”]

[FREQ <Compare> “<Freq>”]

2: SEARCH RESPONSE (Function: 0xC9)

„<Filename>“

<MD5>

<Size>

<Bitrate>

<Freq>

<Time>

<Nick>

<IP>

<Link-Type>

Central
Napster
Index

SEARCH(0xC8)

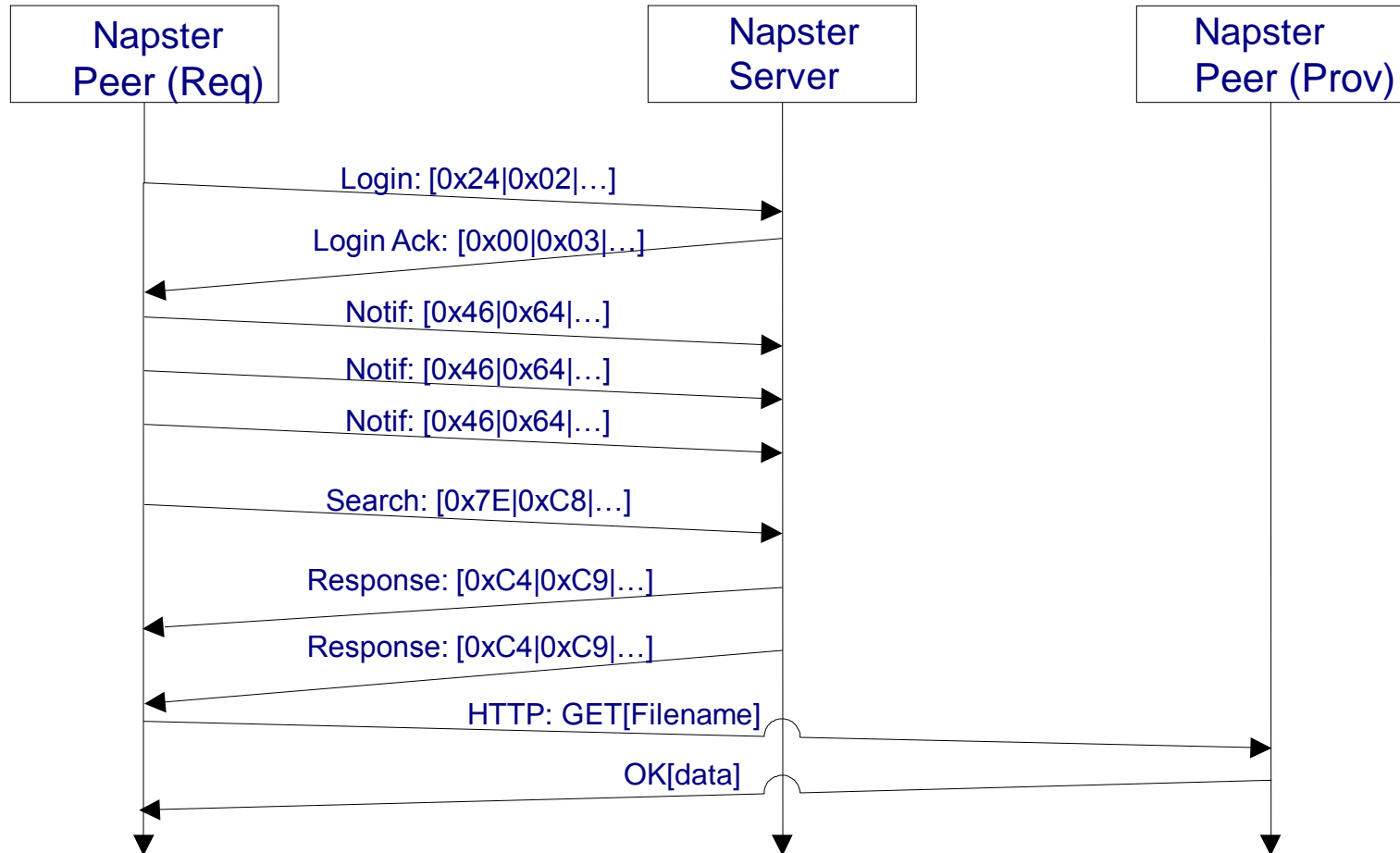
FILENAME CONTAINS „song“ MAX_RESULTS 100

LINESPEED „AT LEAST“ 6 BITRATE „AT LEAST“ „128“

FREQ „EQUAL TO“ „44100“

Napster Host
IP: 002
Nick: MIT

Summary of Napster Signalling



Sample message sequence chart for a Napster server with a requesting and a providing peer

Napster: Conclusions

Disadvantages

- Single Point of Failure
→ easily attackable
- Performance bottleneck
- Potential of congestion
- Central server in control of all peers

Advantages

- Fast and complete lookup
- Central managing/trust authority
- No keep-alive messages necessary beyond content updates

Napster-like application areas

- VoIP (SIP, H.323)
- Auctioning (Ebay)

Extensions

- Several statically connected directory servers (OpenNap networks)
- Support for arbitrary file formats (beyond MP3)

Analyses (April 2001)

- OpenNap contains 80 directory servers
- 47,000 users online
- more than 10 Mio. files available
- more than 55 TByte of data available

4.3 Gnutella

Definition of Pure P2P Networks

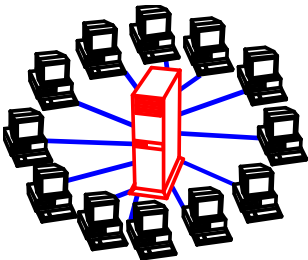
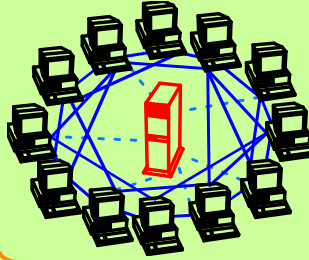
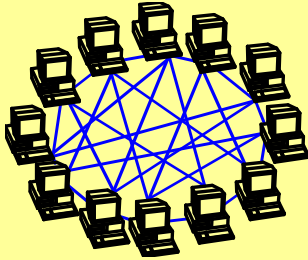
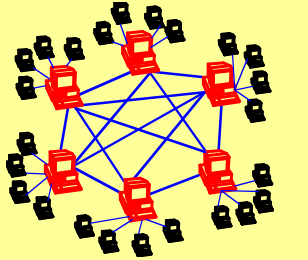
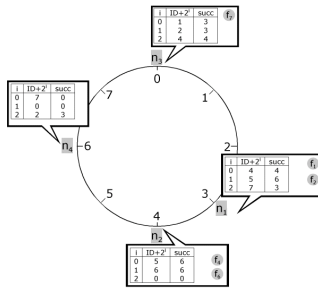
Any peer can be removed without loss of functionality.

No central entity exists in the overlay.

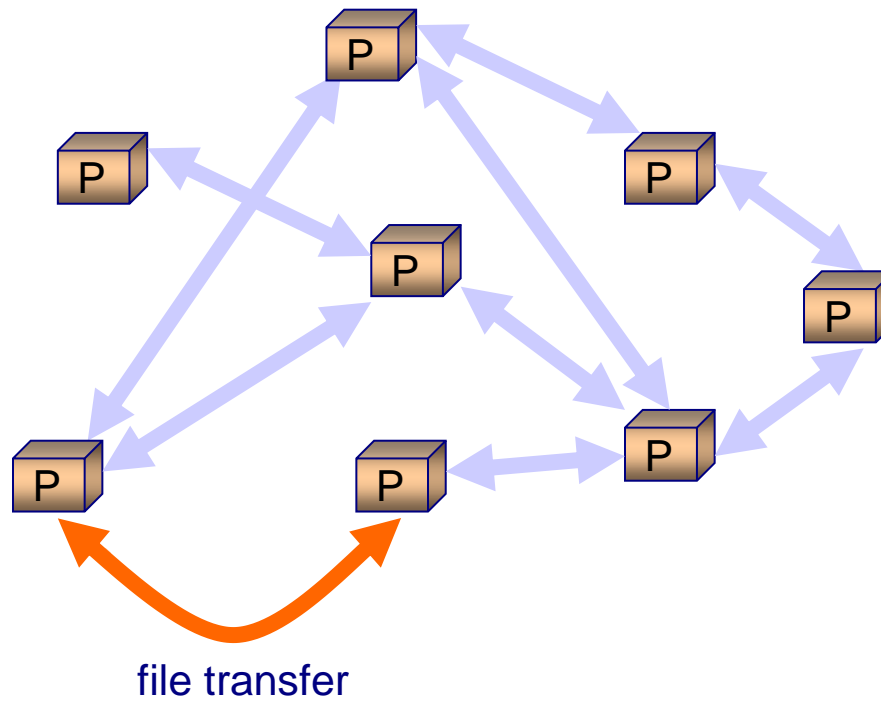
Peers establish connections between each other randomly

- to route request and response messages,
- to insert request messages into the overlay.

Architectures of Peer-to-Peer Networks

Client-Server	Peer-to-Peer			
<ol style="list-style-type: none"> 1. Server is the central entity and only provider of service and content. → Network managed by the Server 2. Server as the higher performance system. 3. Clients as the lower performance systems <p>Example: WWW</p>	<ol style="list-style-type: none"> 1. Resources are shared between the peers 2. Resources can be accessed directly from other peers 3. Peer is provider (server) and requestor (client): the <i>servent</i> concept 			
	Unstructured P2P			Structured P2P
	Centralized P2P	Pure P2P	Hybrid P2P	DHT-Based
	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Central entity is necessary to provide the service 3. Central entity is some kind of index/group database <p>Example: Napster</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → No central entities <p>Examples: Gnutella 0.4, Freenet</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → dynamic central entities <p>Example: Gnutella 0.6, JXTA</p>	<ol style="list-style-type: none"> 1. All features of Peer-to-Peer included 2. Any terminal entity can be removed without loss of functionality 3. → No central entities 4. Connections in the overlay are “fixed” <p>Examples: Chord, CAN</p>
				
	1 st Gen.		2 nd Gen.	

A Pure P2P Network



Basic Characteristics of Pure P2P Networks (1)

Bootstrapping

- Via a bootstrap server (host list from a web server), or
- via the peer cache (from previous sessions), or
- via a well-known host.
- No registration necessary!

Routing

- Completely decentralized
- *Reactive protocols*: routes to content peers are only established *on demand*, no content announcements
- Requests: flooding (limited by TTL and GUID)
- Responses: routed (backward routing point-to-point with the help of the GUID)

Basic Characteristics of Pure P2P Networks (2)

Signaling connections

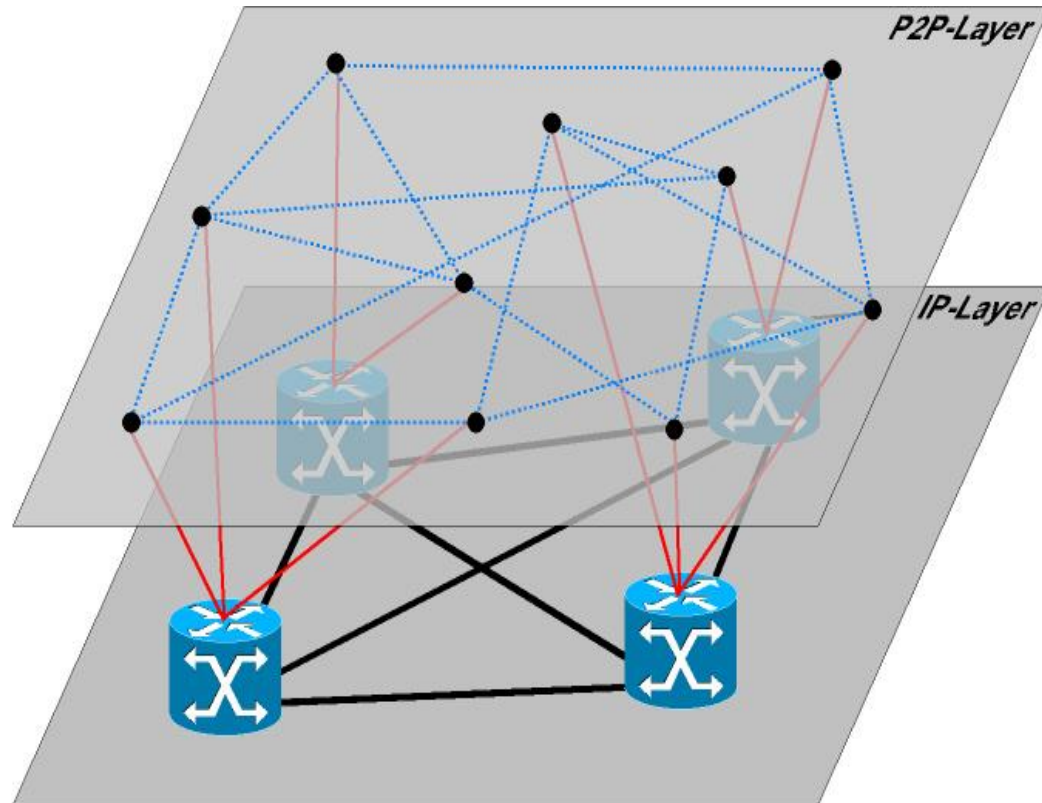
(stable as long as the neighbors do not change):

- based on TCP
- keep-alive messages
- content search

Content transfer connections (temporary)

- based on http, direct connections between the peers
- “out-of-band” transmission (outside of the Gnutella network)

Topology of Pure P2P Networks



● Servent

..... Connection between two servents (TCP)

— Connection between router and servent

— Connection between routers (core)

Basic Routing Behavior (1)

Request messages

- include a hop-counter, a GUID and a TTL (Time-to-Live) in the header
- The TTL determines over how many hops a message may be forwarded.
- Messages are *flooded* over the overlay network:
 - Every node forwards every incoming message to all neighbors except the neighbor from which it came.
- Request messages terminate early if
 - the same message type with the same GUID is received more than once (loop!)
 - Hop-counter = TTL

Basic Routing Behavior (2)

Response messages

- include a hop-counter, a GUID and a TTL (Time-to-Live) in the header.
- GUID is the same as that of the initializing request message.
- Are routed back on the same way to the requestor.
 - → Every peer has to store the GUID of each request for a certain amount of time
 - No flooding on the way back to save resources

Bootstrapping (1)

Bootstrapping

- Mostly not part of the protocol specification
- Necessary to know at least one active participant of the network

Bootstrapping (2)

Address (TCP) of an active node can be retrieved by

- a bootstrap cache
 - Nodes try to establish a connection to a node seen in a previous session.
- a bootstrap server
 - Connect to a “well known host”, which almost always participates.
 - Ask that bootstrap server to provide a valid address of at least one active node.
 - Realizations:
 - FIFO of all node addresses which recently used this bootstrap (a node that just connected is assumed to be still active)
 - Random pick of addresses which recently connected via this server to the overlay
 - + no loops
 - may be outdated
- Broadcast on the IP layer
 - use multicast channels
 - use IP broadcasting (- limited to the local network)

Example: Gnutella 0.4

A program for sharing files over the Internet

Focus: a *decentralized* method of searching for files on other hosts

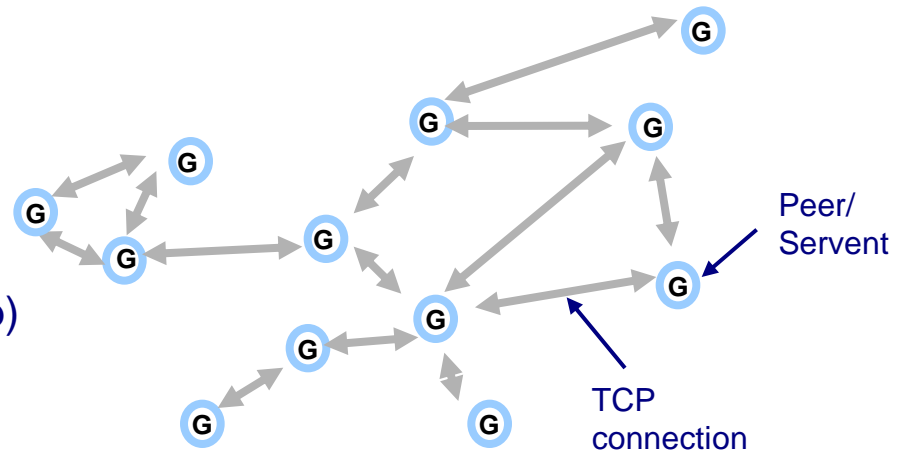
Brief History

- **March 2000:** open source release by Justin Frankel and Tom Pepper of Nullsoft, a division of AOL, and almost immediately withdrawn
- **Spring 2001:** further developments to improve scalability → Gnutella 0.6 (a hybrid P2P system)
- Since then:
 - Available in many implementations (limewire, bearshare,...)
 - Further developments of privacy, scalability, performance,...

Gnutella Operation (1)

An application-level P2P protocol over point-to-point TCP participants.

- Router Service
 - Flood incoming requests (watch the TTL!)
 - Keep-alive
 - Content
 - Route responses for other peers (based on the GUID of the message)
 - Keep alive (ping/pong)
 - Content (QUERY/QUERYHIT)
 - Data requests
 - Download requests
- Lookup Service
 - Initialize data requests
 - Initialize keep alive requests
- “Server”-Service
 - Serve the data requests (http)



Gnutella Operation (2)

Five steps

- Connect to at least one active peer (address received from the bootstrap node)
- Explore your neighborhood (ping/pong)
- Submit query with a list of keywords to your neighbors (they might forward it)
- Select the “best” of the correct answers (which we receive within a time span)
- Connect to the providing peer.

Gnutella Operation (3)

Peers are connected via TCP connections

Search requests are flooded over the Gnutella network

- TCP "broadcast" of `ping` and `query` messages

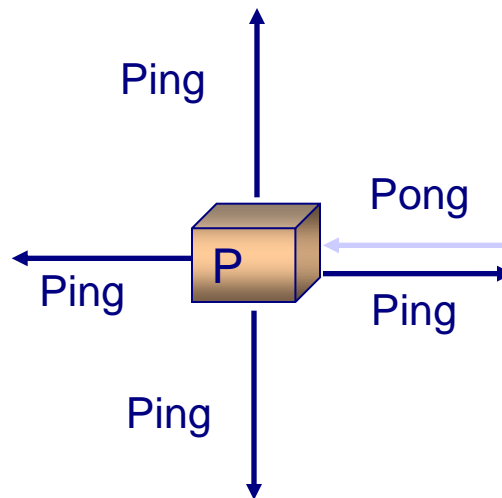
Discovery of routing loops through pseudo-unique message IDs (GUID)

- GUID (128 bit) consists of, e.g., a time stamp, a random number and the MAC address
- Two occurrences of same GUID are possible, but very unlikely.
- Temporary storage of GUIDs of received messages (soft state)
- Discard messages that have been received more than once.

Protocols of Gnutella (1)

Discovery of new peers

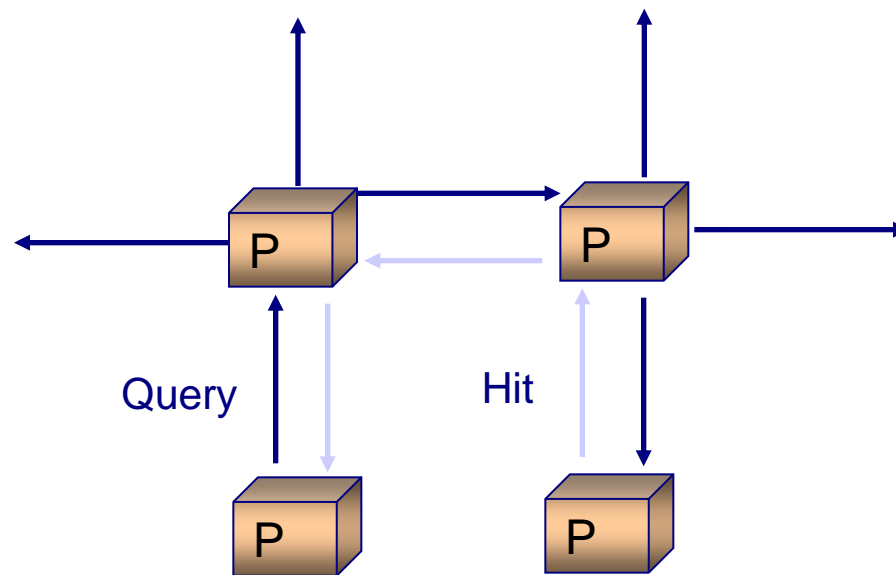
- Each peer maintains a dynamic list of its neighbors
- Discovery of new neighbors by sending out `ping` messages periodically.
 - Recipients reply with `pong` messages.
- Passive monitoring of `ping/pong` messages.



Protocols of Gnutella (2)

Searching for files

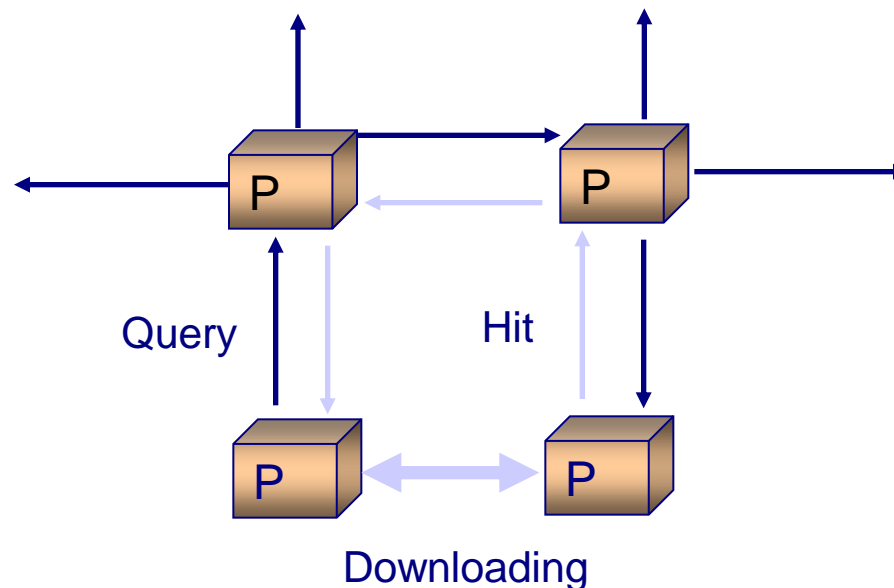
- Peer sends a `query` message.
- All recipients compare request with their local files.
- A `QueryHit` message is sent back in case of a hit.



Protocols of Gnutella (3)

Downloading files

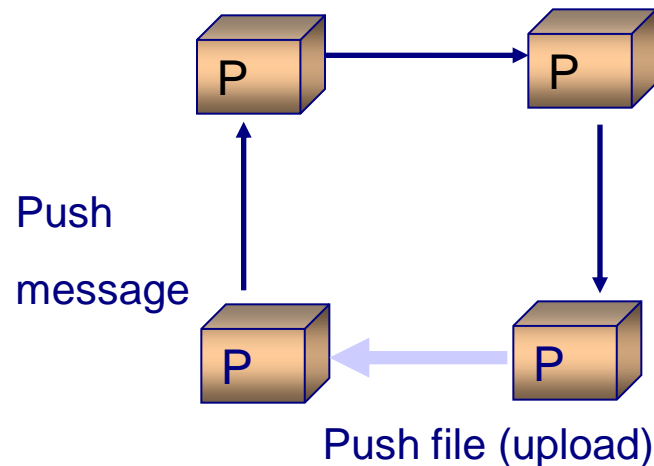
- Peer builds up a direct connection outside of the Gnutella network (“out-of-band”).
- Downloading by a `GET HTTP` request
- Problem: firewalls may block incoming connections
→ Initiation of a download by using *push* messages



Protocols of Gnutella (4)

Push message

- Recipient of a `QueryHit` message sends the *push* message to the peer behind the firewall, using the Gnutella network.
- Recipient of the *push* message initiates the upload of the requested file.
- Problem: Does not work if both peers are behind a firewall!



Protocols of Gnutella (5)

Problem: exploitation for DDoS attack

- *Spoofing* of source address of push message by using address of the site to be attacked
- Sending of such messages to many peers (reflectors)
- Reflectors try to set up data connections to the point of attack.
- Trace back is very difficult in a Gnutella network.

Gnutella Message Structure (1)

General Header Structure:

MESSAGEHEADER: 23 Byte

GnodeID
16 Bytes

Function
1 Byte

TTL
1 Byte

Hops
1 Byte

Payload Length
4 Bytes

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

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

- **GnodeID**: unique 128 bit identifier of a host
- **TTL** (Time-To-Live): number of servants a message may pass before it is destroyed
- **Hops**: number of servants a message has already passed

Gnutella Message Structure (2)

PING (Function:0x00)

No Payload

PONG (Function:0x01)

Port

2 Bytes

IP Address

4 Bytes

**number of shared
files (4 Bytes)**

**number of Kbytes shared
4 Bytes**

QUERY (Function:0x80)

Minimum Speed

2 Bytes

Search Criteria

n Bytes

QUERY HIT (Function:0x81)

of hits

1 Byte

Port

2 Bytes

IP Address

4 Bytes

Speed

1 Byte

Result Set

n Bytes

GnodeID

16 Bytes

File Index

4 Bytes

File Name

n Bytes

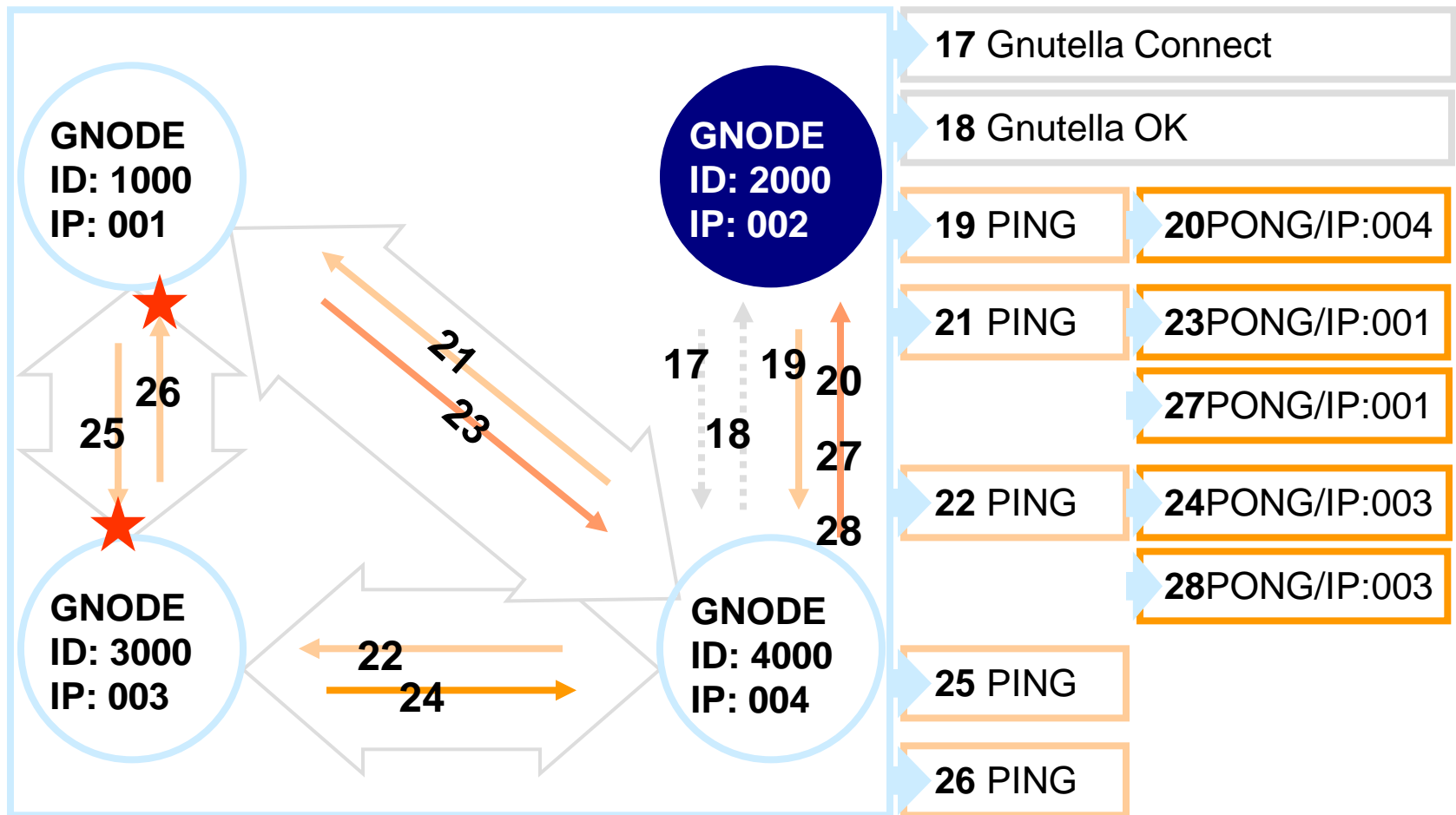
Gnutella Routing

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 **QUERYs** 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 Gnode 4000.



Gnutella Does Not Scale (1)

Reachable users

	<i>T=2</i>	<i>T=3</i>	<i>T=4</i>	<i>T=5</i>	<i>T=6</i>	<i>T=7</i>	<i>T=8</i>
<i>N=2</i>	4	6	8	10	12	14	16
<i>N=3</i>	9	21	45	93	189	381	765
<i>N=4</i>	16	52	160	484	1,456	4,372	13,120
<i>N=5</i>	25	105	425	1,705	6,825	27,305	109,225
<i>N=6</i>	36	186	936	4,686	23,436	117,186	585,936
<i>N=7</i>	49	301	1,813	10,885	65,317	391,909	2,351,461
<i>N=8</i>	64	456	3,200	22,408	156,864	1,098,056	7,686,400

N = number of connections open

T = number of hops


Gnutella default

Gnutella Does Not Scale (2)

Load generated in bytes (a message has 83 bytes)

- Searching for an 18 byte string

	<i>T=2</i>	<i>T=3</i>	<i>T=4</i>	<i>T=5</i>	<i>T=6</i>	<i>T=7</i>	<i>T=8</i>
<i>N=2</i>	332	498	664	830	996	1,162	1,328
<i>N=3</i>	747	1,743	3,735	7,719	15,687	31,623	63,495
<i>N=4</i>	1,328	4,316	13,280	40,172	120,848	362,876	1,088,960
<i>N=5</i>	2,075	8,715	35,275	141,515	566,475	2,266,315	9,065,675
<i>N=6</i>	2,988	15,438	77,688	388,938	1,945,188	9,726,438	48,632,688
<i>N=7</i>	4,067	24,983	150,479	903,455	5,421,311	35,528,447	192,171,263
<i>N=8</i>	5,312	37,848	262,600	1,859,864	13,019,712	91,138,648	637,971,200

N = number of connections open

T = number of hops


Gnutella default

Gnutella Does Not Scale (3)

First Gnutella generation

- Same number of connections for all peers
 - Peers with low bandwidth couldn't deal with the traffic load.
 - A 56K modem is usually already overloaded for network sizes of 1000 nodes or more.
 - Fragmentation of the Gnutella network with more than 4000 nodes in August 2000

Second Gnutella generation

- Number of peer connections are adapted to the bandwidth available.
- Connections to overloaded peers are terminated.
- Scalability still not satisfying

Gnutella Does Not Scale (4)

Third Gnutella generation

- Introduction of *hierarchies*
- „Super Peers“ take over load of peers with low bandwidth
- Not really a pure peer-to-peer network anymore
- Architecture similar to Fast-Track networks

Fast Track

- Being used by Morpheus, KaZaA, ...
- An extension of Gnutella

Gnutella 0.4: Conclusions

Disadvantages

- High signaling traffic because of decentralization and flooding
- Modern nodes may become bottlenecks
- Overlay topology not optimal as
 - no complete view available,
 - no coordinator
- If not adapted to the physical structure, delay and total network load increase

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