P2P Paradigm

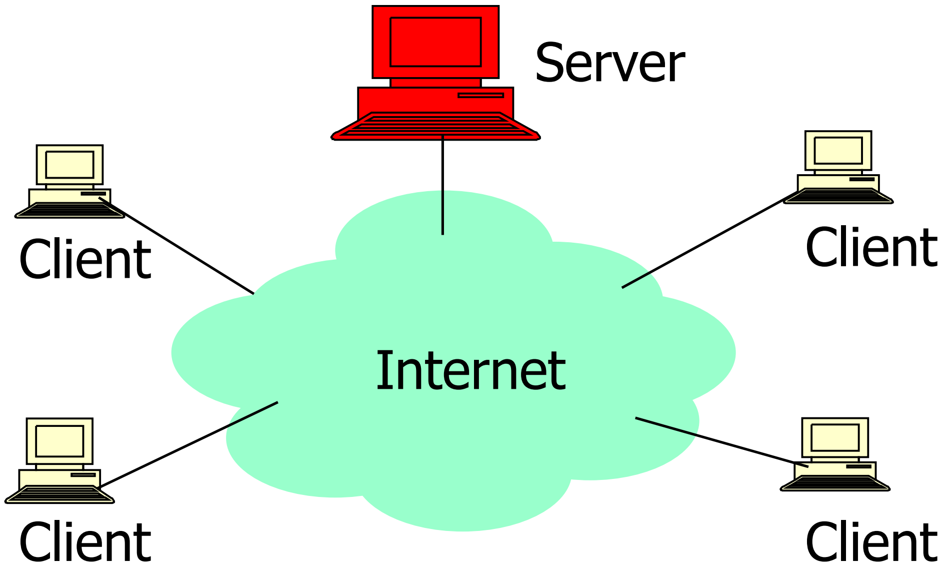
Lenuta Alboaie (adria@info.uaic.ro)
Andrei Panu (andrei.panu@info.uaic.ro)

Content

- **Peer-to-peer (P2P) paradigm**
  - Preliminaries
  - Definitions
  - Characteristics
  - Application types
  - Infrastructures
  - Instruments

Preliminaries

...let's remember the client/server model



Preliminaries

...let’s remember the client/server model

- Usually, we look at the client as at a component having low computational capabilities
- The server is maintained and managed centrally

Problems of client/server architecture:

- The lack of robustness
- Lack of safety (reliability)
- Lack of scalability
- Vulnerability to attacks

Definitions

Peer *= one that is of equal standing with another* (according to Webster)

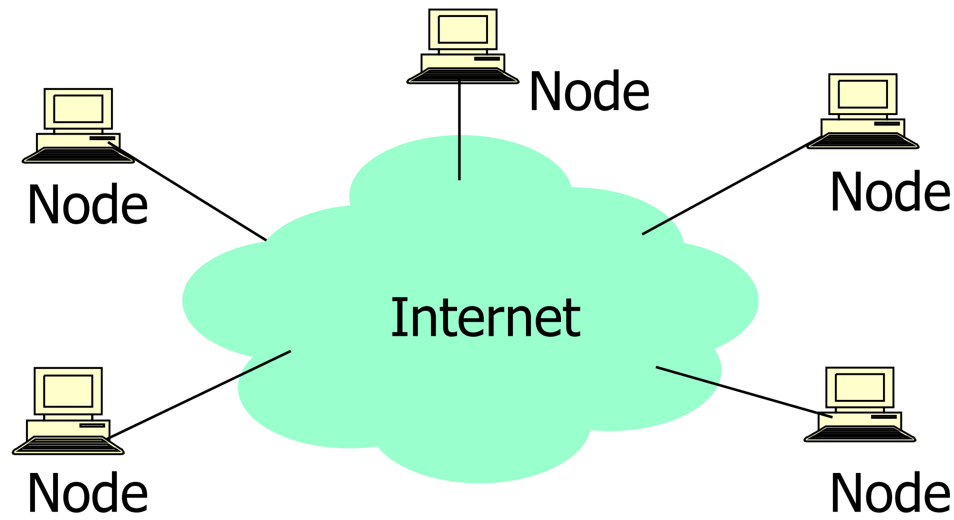
Peer-to-peer (P2P) = network architecture where nodes are relatively equal

- Meaning that each node is capable of performing specific network functions

Definitions

P2P Systems, in the strict sense, are fully distributed systems

- All nodes are fully equivalent in terms of functionality and the activities that they can perform



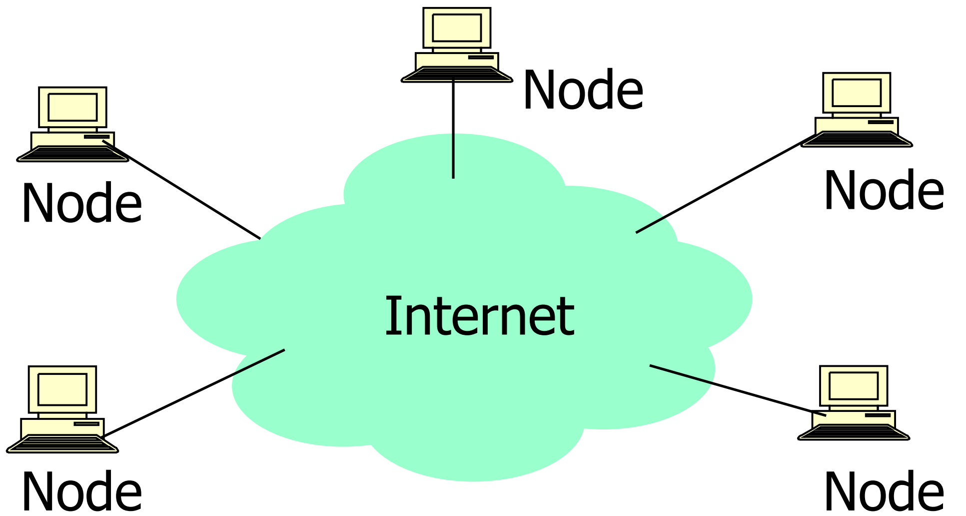
Obs.: Pure P2P systems are rare (e.g., Gnutella), most are hybrids, having super-nodes or servers with different roles (data search, control)

Definitions

- Nodes
  - can consume and provide data
  - any node can initiate a connection
- There is no centralized data source =>

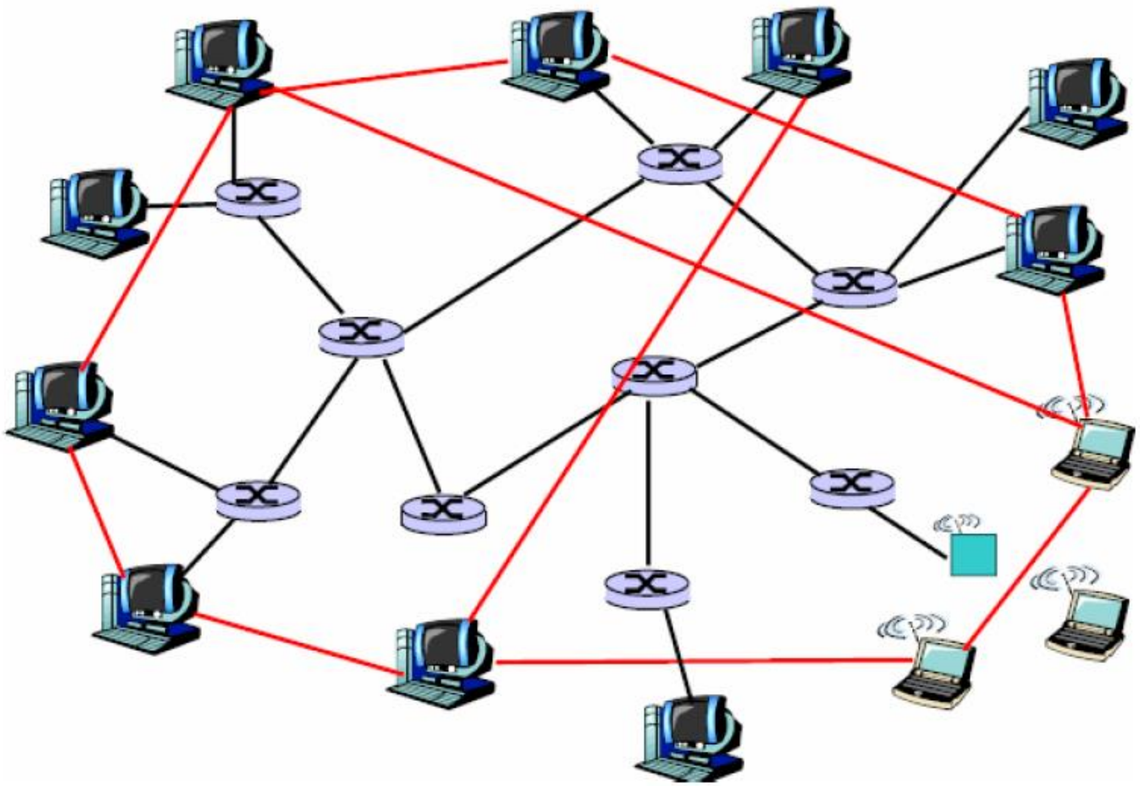
A form of democracy on the Internet

Copy-right protection threatened



Definitions

- "**P2P** is the class of applications that rely on the resources (storage, processing, content, human presence) available at the edges of the Internet



Edges of the Internet (*overlay networks*)

Definitions

"**P2P** is a class of applications that take advantage of resources – storage, cycles, content, human presence – available at the edges of the Internet. Because accessing these decentralized resources means operating in an environment of unstable and unpredictable IP addresses, P2P nodes must operate outside the DNS system and have significant, or total autonomy from central servers"

"A distributed network architecture may be called a **P2P** network if the participants share a part of their own resources. These shared resources are necessary to provide the service offered by the network. The participants of such a network are both resource providers and resource consumers"

Characterization

Characteristics:

– Sharing computational resources through interchange and less directly through mediation offered by a centralized authority (server)

  - Centralized servers can be used to perform specific activities (e.g., P2P network initialization, adding new nodes in the network, etc.)
  - Ideally, nodes participate actively in accomplishing operations such as location & caching nodes / content, routing information, transferred resource management, etc.

Characterization

Characteristics:

- The ability to address instability and variations of network connectivity, with adaptation for the occurred errors or for the node dynamics
  - P2P network topology is adaptive and fault tolerant; nodes are self-organized in order to maintain connectivity and network performance

Characterization

P2P network is an overlay over the physical network

- It is at Application Level => flexibility
- Virtual edges are TCP connections or pointers to IP addresses
- P2P network maintenance is done by periodically verifying connectivity (ping) or existence (messages "still alive?")
- When a node fails, the P2P system could set new edges
- Nodes proximity (physical) is not important
- P2P network can be structured or not

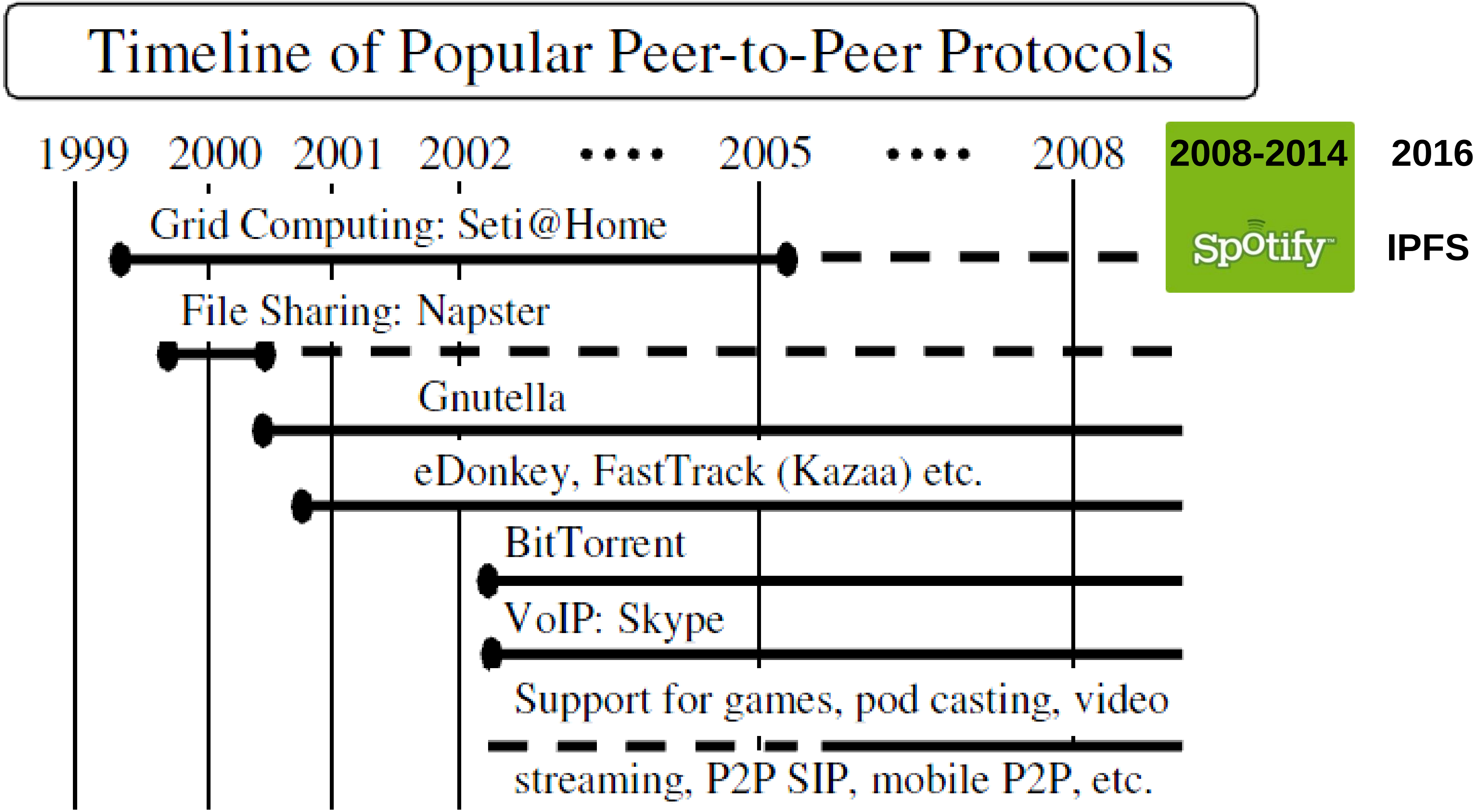
Aims and benefits

- **Efficient use of resources**
  - Unused bandwidth, storage resources, processing power available to the network edges
- **Scalability**
  - Without centralized information, without bottlenecks (communication and computing)
  - Integration of resources is done naturally during the system use
- **Reliability**
  - The existence of copies of data
  - Geographical distribution
  - No more "single point of failure“
- **Easy administration**
  - The nodes are self-organised
  - Increased fault tolerance and load balancing
  - Increased autonomy
- **Anonymity**
  - Hard to accomplish in a centralized environment
- **Dynamism**
  - Dynamic environment
  - Ad-hoc collaboration and communication

Disadvantages / Problems

- P2P architectures are probabilistic
  - Unpredictable resource location
  - Resources are volatile
- Absence of centralized control
  - Issues with imposing an authority on applications, content and users
  - Difficulties in detecting and identifying users (anti-social aspects)
- Encouraging the use of P2P systems for abusive and illegal purposes (e.g., copyright of digital content)
  - Lack of trust at the commercial/business level
  - Security issues (future course)

Evolution…



Timeline of Popular Peer-to-Peer Protocols
1999
2000
2001
2002
2005
2008
2008-2014
2016
Grid Computing: Seti@Home
File Sharing: Napster
Gnutella
eDonkey, FastTrack (Kazaa) etc.
BitTorrent
VoIP: Skype
Support for games, pod casting, video
streaming, P2P SIP, mobile P2P, etc.
Spotify
IPFS

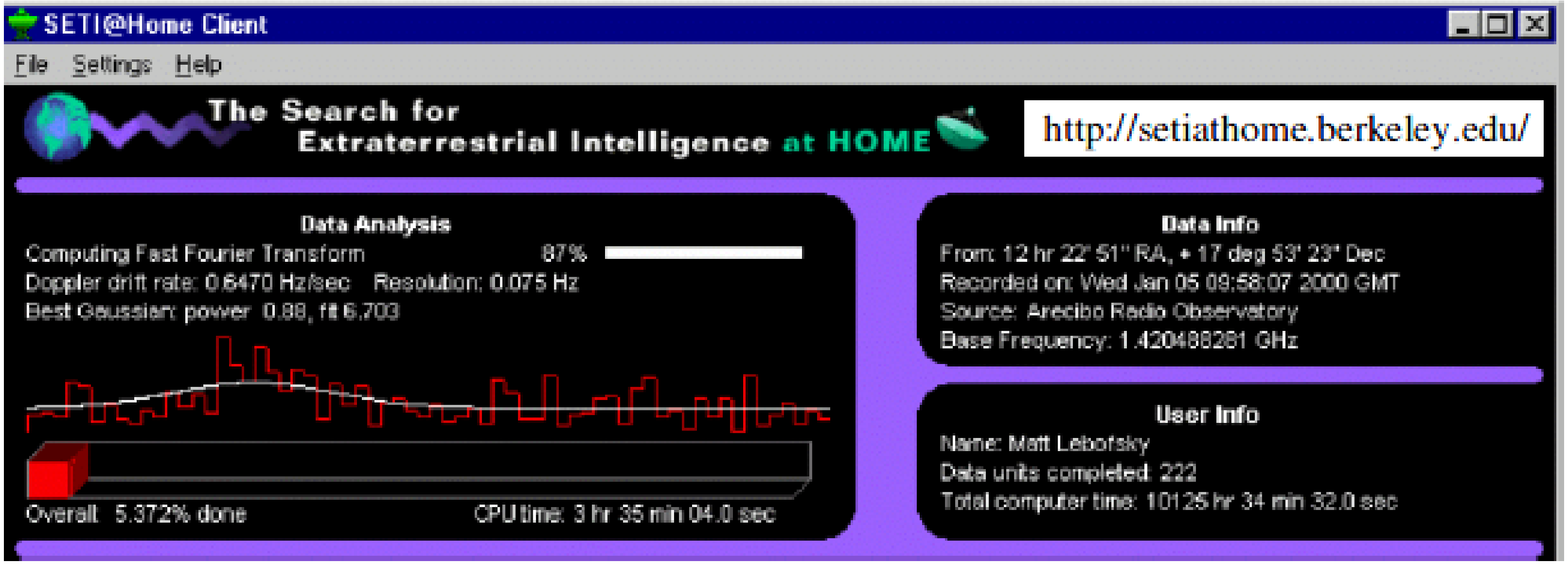
Types of Applications

- **Communication & collaboration**
  - Systems that provide an infrastructure to facilitate direct communication & collaboration, often in real time between nodes
  - Conversational systems (chat, instant messaging):

    IRC (Internet Relay Chat), ICQ (1996), YM !, MSN Messenger, Skype, P2P multicast systems (e.g., Cirrus - Adobe Flash http://labs.adobe.com/technologies/cirrus/; WebRTC) ...
- **Distributed Computing**
  - Systems that use the computational resources of available nodes (processor cycles)
  - Solving problems through *divide-et-impera*: SETI@home (*Search for Extra-Terrestrial Intelligence*-Berkeley), genome@home
    - P2P network is a kind of a computational Grid (… master course)

Types of Applications| Distributed Computing - Example

SETI@Home: A Public-Resource Computing Experiment



- Radio telescope signal analysis has insatiable appetite for computing power
- Usage of computers in homes and offices around the world has provided unprecedented computing power
- Grid computing application via peer-to-peer approach under central control

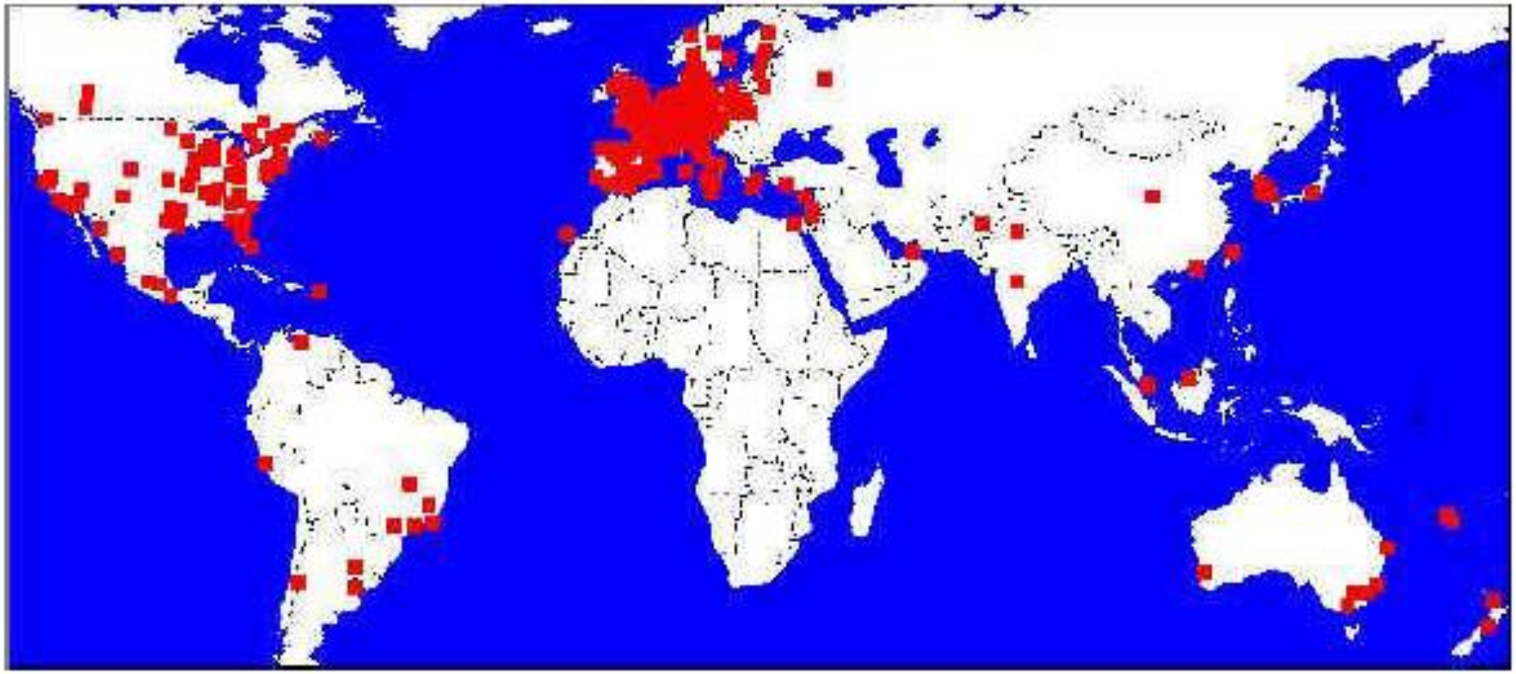
Types of Applications

- **Storage systems (databases)**
  - Design of distributed database infrastructure based on P2P
    - PIER – a scalable engine for distributed query (http://pier.cs.berkeley.edu/)
    - Edutella – open-source project for queries and *meta-data* storage (P2P for Semantic Web)
- **Distribution of digital content**
  - Systems & Infrastructure for sharing digital resources (multimedia and other data) among users
    - File sharing applications (e.g., Napster, Gnutella, KaZaA, Freenet, BitTorrent, eDonkey, etc.)
    - Distributed storage media for publishing, organizing, indexing, searching and retrieving data in secure & efficient manner (e.g., PAST, Chord, Groove, Mnemosyne, Avalanche, etc.)

Types of Applications

- **Distribution of digital content| Example**

P2P File-Sharing: Fast distribution of large files



Example: Harry Potter III early propagation after 2 hours on May 28th 2004 (Source: *www.itic.ca/DIC/News/archive.html*)

Types of Applications

- **Distribution of content through P2P**
  - P2P systems for "interchanging files"
    - Nodes transfer files one at a time
    - It offers facilities for realizing a P2P network and for searching & transferring files between nodes
    - Security, availability and persistence are not supported
    - Examples: Napster, KaZaA, Gnutella

Types of Applications

- **Distribution of content through P2P**
  - P2P systems for publishing & storing content
    Users can publish, store and distribute digital content, based on access rights (privileges)
    - Focus is on security and persistence
    - Some systems offer facilities regarding collaboration between users
    - Example: Scan, Groove, Freenet, MojoNation, Tangler

Types of Applications

- **Distribution of content through P2P**
  – Infrastructure for:
    - Routing & Localization:

      Chord, Can, Pastry, Tapestry, Kademila
    - Anonymity:

      Onion Routing, ZeroKnowledge, Freedom, Tarzan
    - Reputation management:

      EigenTrust, PeerTrust

Infrastruct.(Routing & Localization)

- Localization and routing mechanisms that can be adopted depend on:
  - Topology
  - Structure
  - The degree of centralization

  of the *overlay network*

Infrastruct.(Routing & Localization)

- **Aspects regarding centralization**
  - **Pure decentralized architectures**: all nodes perform exactly the same activities, by playing roles of servers and clients simultaneously, without the benefit of a central coordination
    - Nodes are called servents (SERVers + clieENTS)

Infrastruct.(Routing & Localization)

- **Aspects regarding centralization**
  - **Partially centralized architectures**: some nodes have a more important role (e.g., storing local indexes for shared folders)
    - Nodes become super-nodes in accordance with the policies of each P2P system
    - Super-node role is determined dynamically
  - **Hybrid decentralized architectures**: there is a central server enabling the interaction between nodes, keeping catalogs with metadata of files
    - Servers can identify and verify the storage nodes
    - The systems are called *broker mediated*

Infrastruct.(Routing & Localization)

- **Aspects regarding network structure:**
  - **Unstructured:** placing content is completely independent of overlay network topology
    - The content must be located
    - Search strategies by "brute force“: flooding the network - requests propagated via BFS / DFS
    - More sophisticated strategies: random path, probabilistic methods, etc.
  - **Loosely structured**: although the content location is not completely specified, it is affected by routing
    - Category located between structured and unstructured networks

Infrastruct.(Routing & Localization)

- **Aspects regarding network structure:**
  - **Structured:** topology is controlled, files are placed in precise locations
    - A mapping between the content (file identifier) and the location (node address) is performed
      - Like a distributed routing table
    - *exact-match queries* can be performed in a scalable way
    - The structure used to guide message routing is difficult to maintain for transient nodes (with high rates of attachment and disconnection from the network)

Infrastruct.(Routing & Localization)

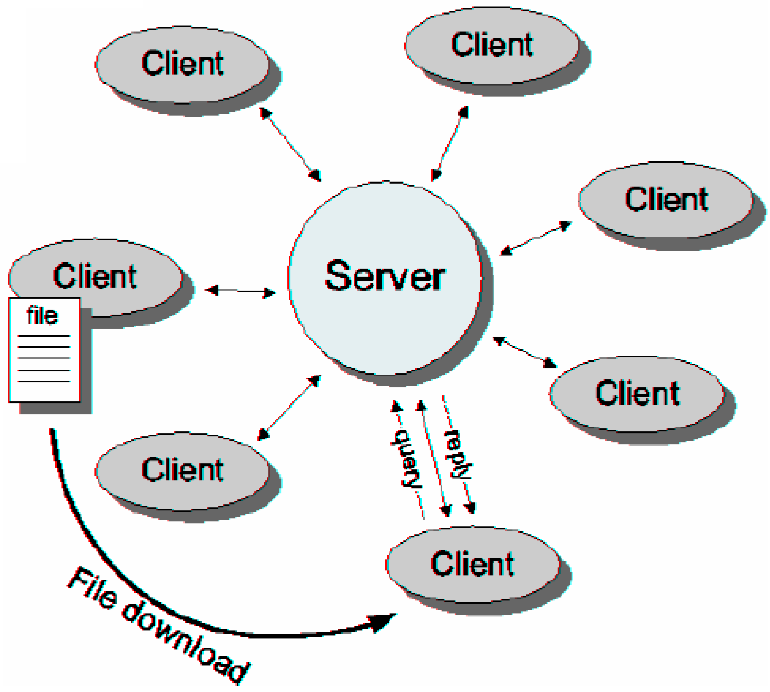
	Centralization		
	Hybrid	Partial	Absent
Unstructured	Napster Publius	KaZaA Morpheus Edutella	Gnutella FreeHaven
Structured Infrastructure			Chord, CAN, Tapestry, Pastry
Structured Systems			OceanStore Scan, PAST, Kademlia, Tarzan

Unstructured Architectures

Hybrid Decentralized

- Each client computer stores shared content (files)
- The central server keeps a table with registered users (IP, bandwidth, ...) + a table with the list of files for user & metadata

  Example: **Napster, Publius**



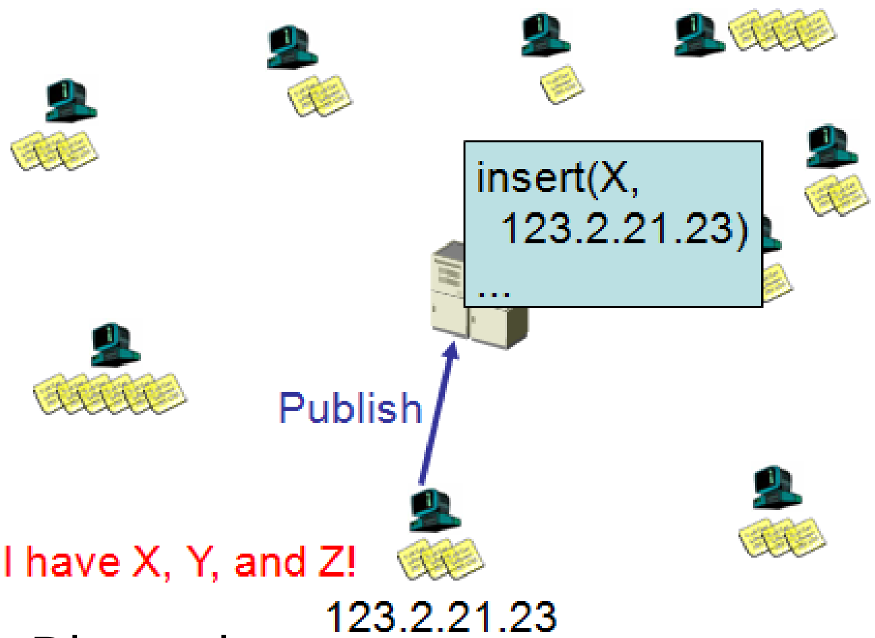
Unstructured Architectures

Napster

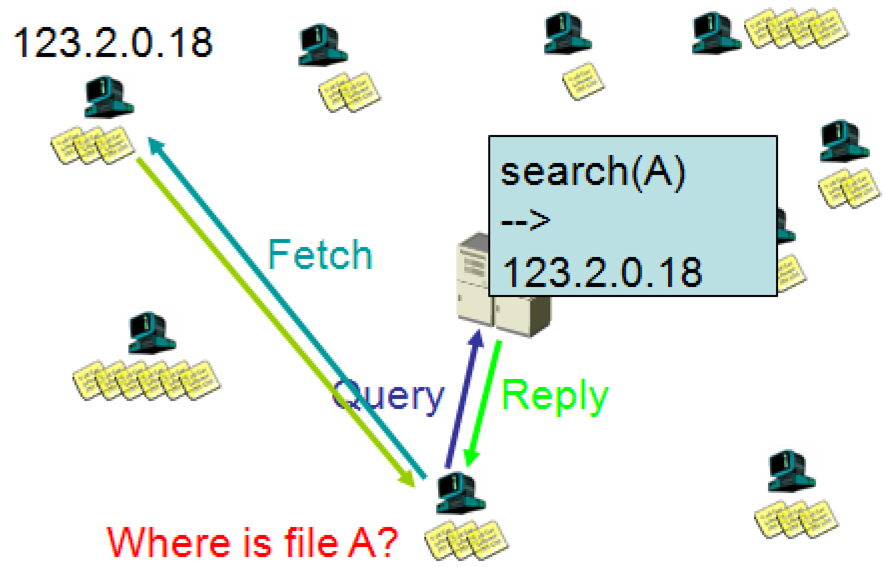
- 1999: Sean Fanning releases Napster
- It reached 1.5 million concurrent users
- Centralized database - operations:
  - **Join**: the client contacts the central server (via TCP)
  - **Publish**: report a list of files to the central server
  - **Search**: query server => it returns a peer that stores the requested file
  - **Fetch**: take the file directly from peer (the one with the best transfer rate)
- July 2001: Napster was closed

Unstructured Architectures

Napster: Publish



Napster: Search



Discussions:

- The server does all the processing
- We have "single point of failure"
- Scalability problems, some systems do not allow adding other servers (the list of available servers is static)

Unstructured Architectures

Pure Decentralized

- An overlay network is built using routing mechanisms based on IP
- There is no central coordination
- Users are connecting via an application that has a dual role – *servent*
- Communication between *servents* is based on a protocol at the application level, with 4 types of messages:
  - Ping – requires a node to announce himself
  - Pong – reply to the *ping* message (IP, port, number and file sizes)
  - Query – search request (search string + minimum speed of transfer)
  - Query hints – response (IP, port, speed, file length, file index)

Unstructured Architectures

Pure Decentralized

- Searching is done by flooding
  - If you don’t have the desired file, ask *n* neighbors
  - If they do not have the file, they will ask their neighbors in maximum *m* hops
  - On the way back, the answers will be returned (not the files’ content)
- Each message has a TTL attached
- Example: **Gnutella**

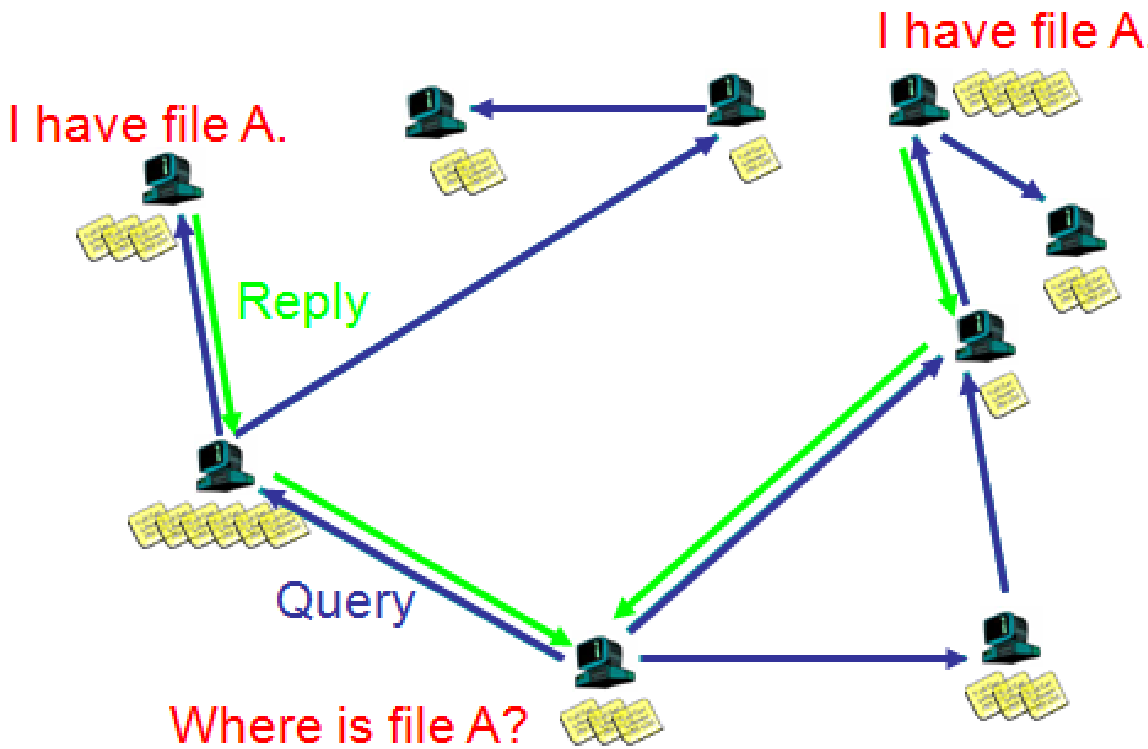
Unstructured Architectures

Gnutella

- 2000: L. Frankel and T. Pepper (Nullsoft) launches Gnutella
- Client applications appear: Bearshare, Morpheus, LimeWire
- *Query Flooding*:
  - **Join**: when joining, the client contacts a few nodes that became his neighbors
  - **Publish**: Not required
  - **Search**: asks neighbors, who ask their neighbors ...
    - There is a TTL limiting the spread
  - **Fetch**: take the file directly from peer

Unstructured Architectures

Gnutella



Issues:

- Search time is… O(?)
- The nodes leave often => unstable network

Unstructured Architectures

Partially centralized

- Use the concept of super-node: that performs various activities in a P2P network (e.g., indexing, caching)
- The nodes are automatically chosen as super-nodes if they have enough bandwidth and computational power
- All requests are sent initially to super-nodes
- Advantages: resource discovery time is less
- Example: **KaZaA**

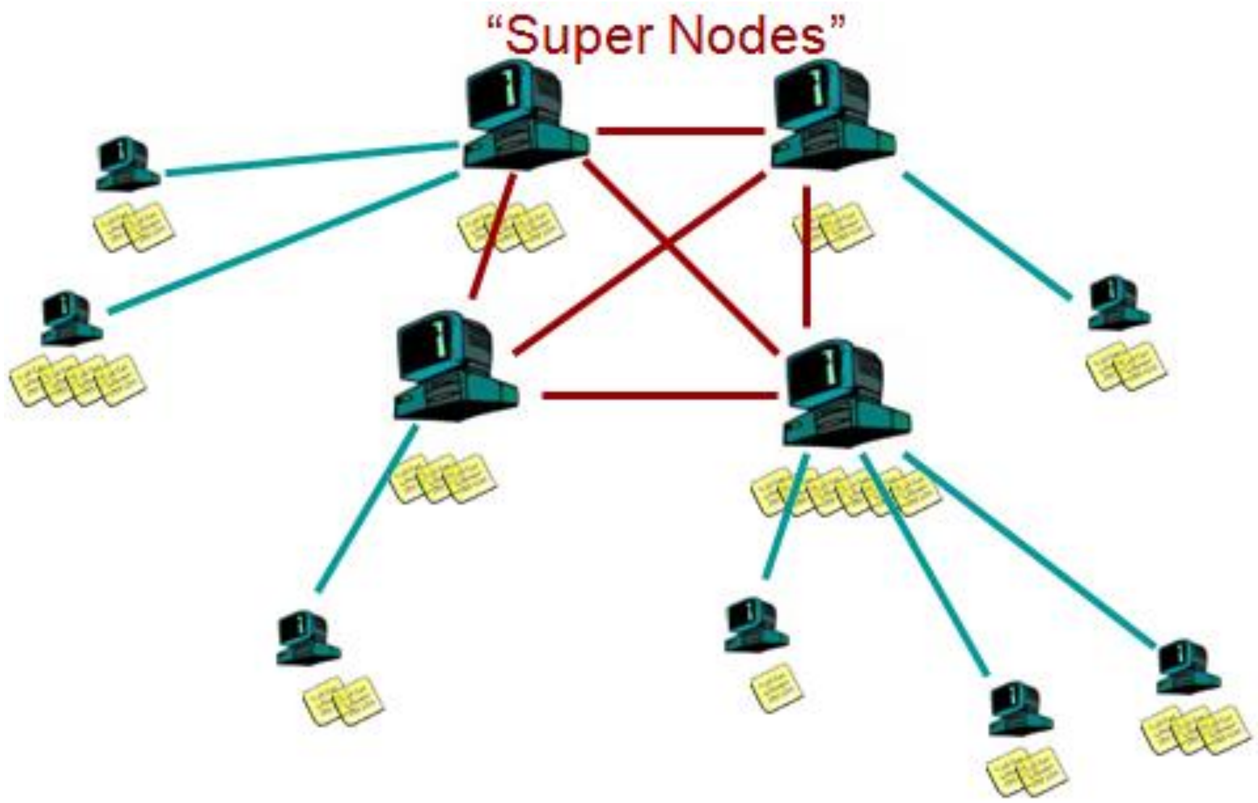
Unstructured Architectures

KaZaA

- 2001: KaZaA is launched
- Client applications appear: Morpheus, giFT
- It utilizes a mechanism of "*smart*" *query flooding*:
  - **Join**: when joining, the client contacts a *super-node* (it can become super-node later)
  - **Publish**: sends the list of files to the *super-node*
  - **Search**: send a query to the *super-node*, and *super-nodes* interrogate each other
  - **Fetch**: take the file directly from the *peer(s);* the file can be obtained simultaneously from multiple peers

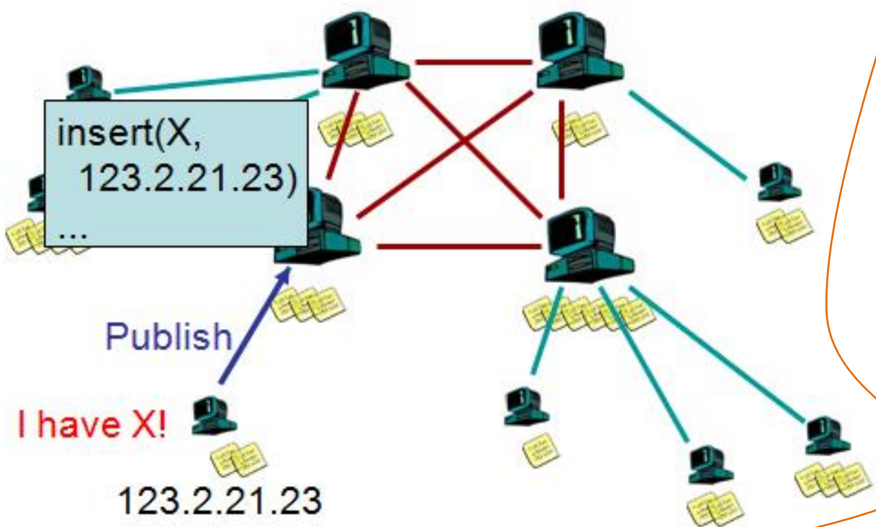
Unstructured Architectures

KaZaA: Network design

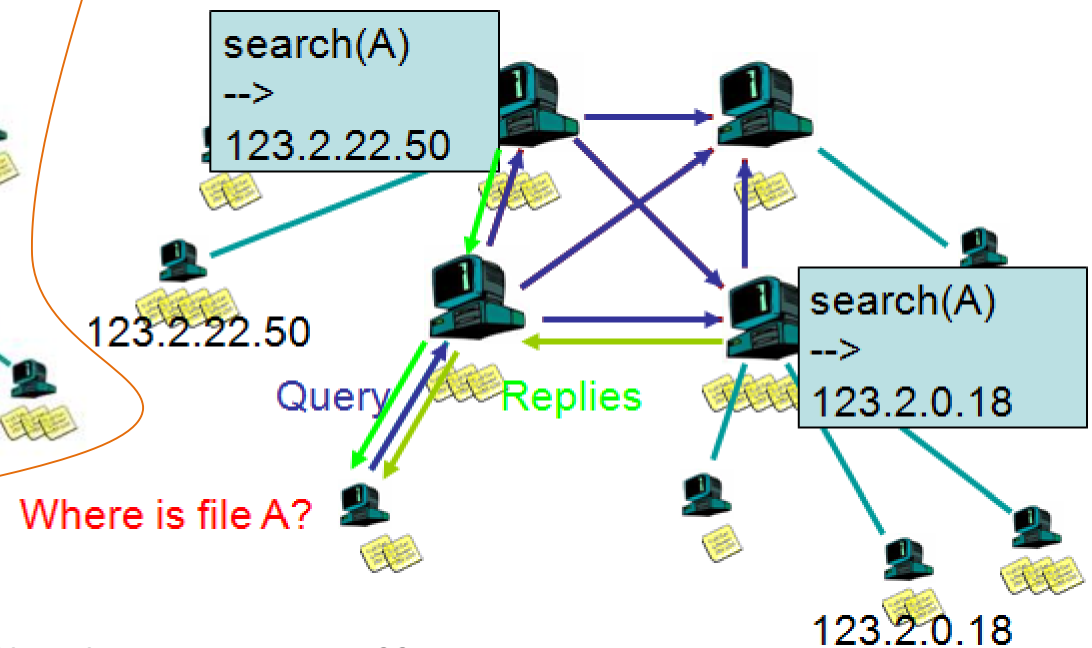


Unstructured Architectures

KaZaA: Inserting Files



KaZaA: Files Search



Issues:

- Behavior similar to Gnutella, but more efficient
- There is no guarantee on the search time or on the search area

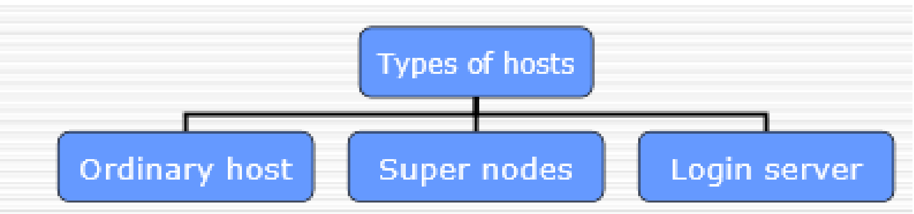
Unstructured Architectures

Partially centralized

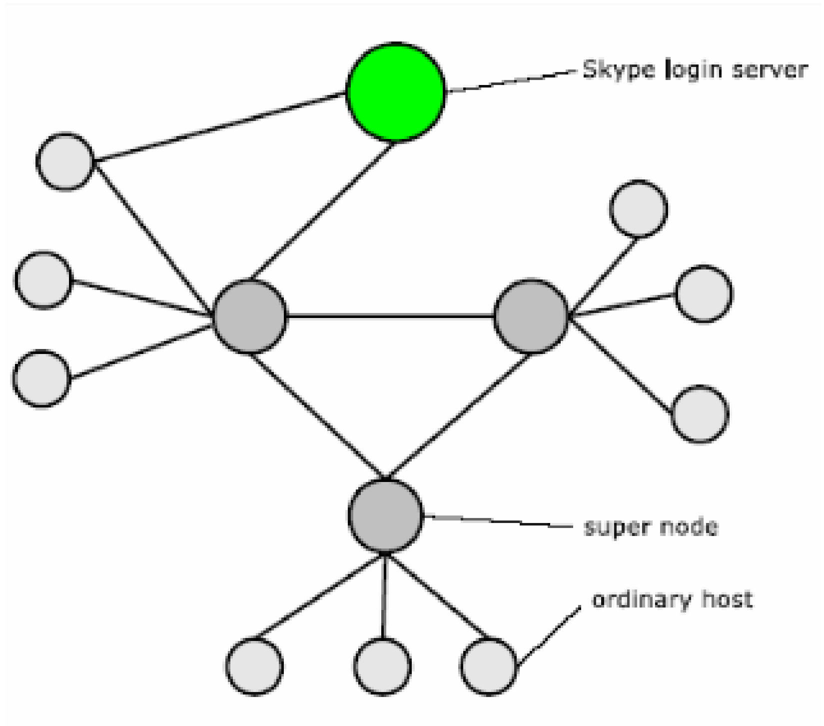
- KaZaA software is a proprietary one
- P2P control data is encrypted
- The messages use HTTP
- A node is either a super-node, or assigned to a super-node
- A super-node has 100-150 child-nodes
- A network can have ~ 30,000 super-nodes
- Each super node has TCP connections with 30-50 super-nodes
- For each file meta-data is maintained (name, size, content hash, file descriptor)
- *The content hash* is used to find another copy of a partially transferred file
- The version without *spyware* and *pop-ups*: **KaZaA-lite**

Unstructured Architectures

Skype



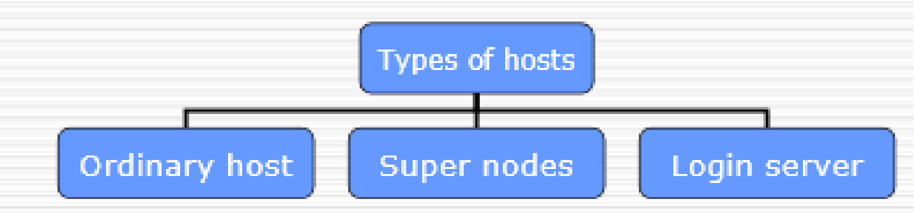
- The first P2P telephony network based on IP
- from June 2014, Microsoft announced the incompatibility with the previous Skype protocol
- Uses Microsoft Notification Protocol 24 (first use -> MSN Messenger in 1999)
- architecture was similar to KaZaA



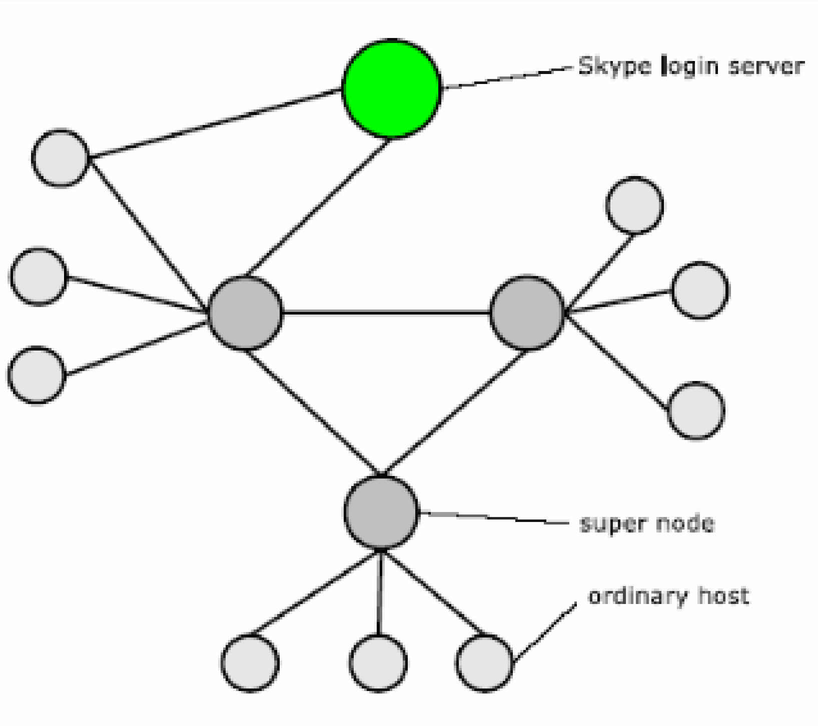
http://www1.cs.columbia.edu/~salman/publications/skype1_4.pdf

Unstructured Architectures

Skype



- Each client maintains a *host cache* with IP addresses and the port numbers associated of the accessible super-nodes
- Any customer with bandwidth (and without firewall or NAT restrictions) could become a super-node
- from 2012, Microsoft began moving super-nodes to the hosting servers in its data centers



http://en.wikipedia.org/wiki/PRISM_%28surveillance_program%29

Unstructured Architectures

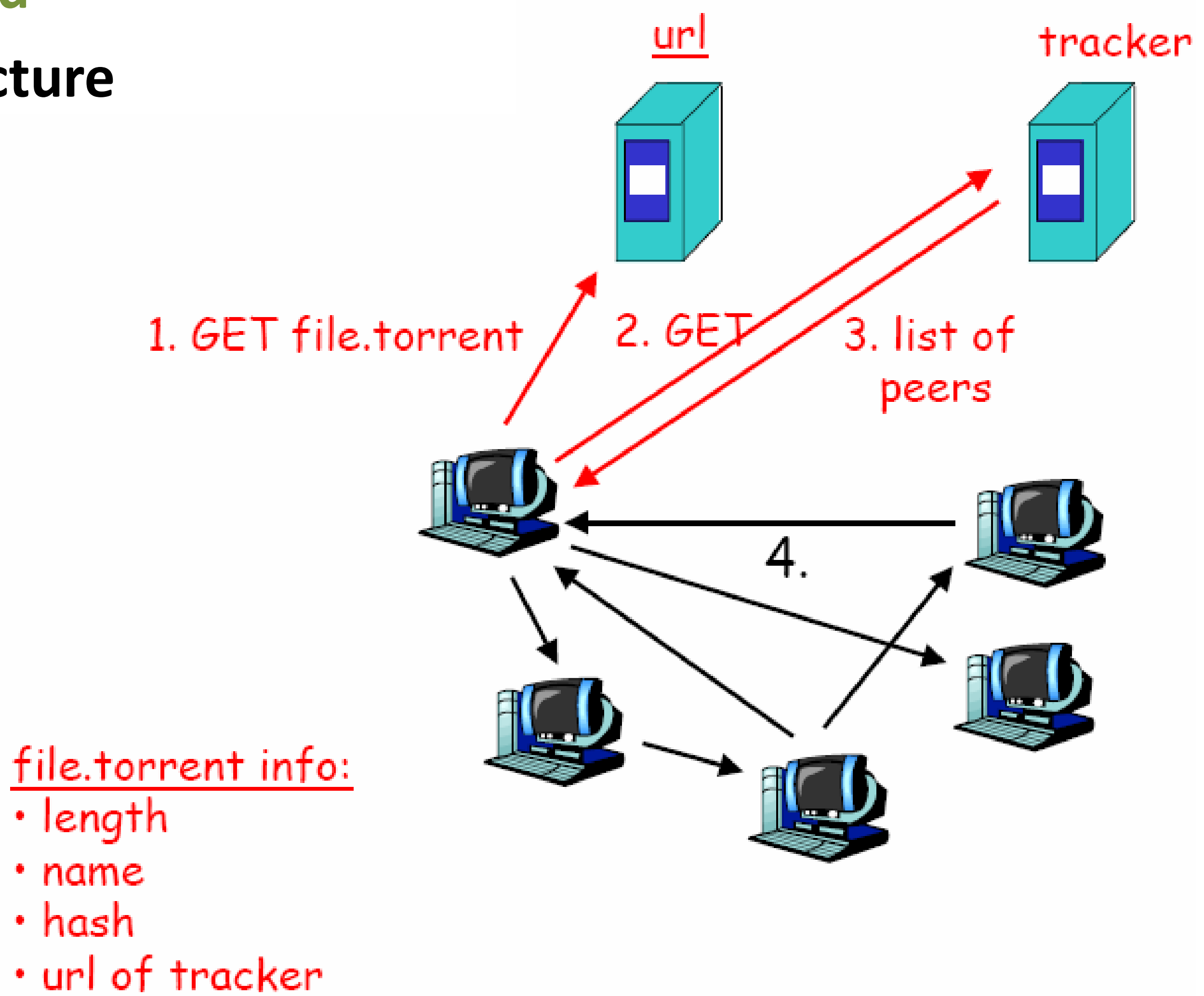
Partially centralized

- If a file is found on multiple nodes, the transfer can be done in parallel
  - The same copies are identified via *content hash*
- Different chunks of the file are transferred from different nodes
- For interrupted transfers, an automatic recovery is performed
- Example: **BitTorrent**
  - In 2002, B. Cohen released BitTorrent
  - The focus was over *efficient fetching* and not on *searching*
  - Supporters since it appeared
    - Blizzard Entertainment used BitTorrent to distribute beta versions of new games

Unstructured Architectures

Partially centralized

BitTorrent - arhitecture



Unstructured Architectures

Partially centralized

BitTorrent

- It is based on *swarming* mechanism:
  - Join: contact a centralized server (*tracker*) and get a list of *peers*
  - Publish: running a *tracker* server
  - Search: e.g., uses Google to find a *tracker* for the desired file
  - Fetch: Take pieces of files from peers;
- Obs.: The difference from Napster
  - *File Chunk Downlod*
  - Using the “tit-for-tat” strategy: if A *downloads* from other nodes, then A should allow downloads from it (*free-rider problem*)

Unstructured Architectures

Problems

- **Nodes whose IP addresses are available via NAT (with restrictions)**
    - They can not be TCP servers for P2P network
    - Partial solution: *reverse call*
        - A wants to transfer from B and B uses NAT
        - A and B establish TCP connections with C server (which has a routable IP)
        - A may request B, via C, to create a TCP connection from B to A
        - A can send a request to B, and B gives the answer
    - If A and B use NAT?
- **Flash crowd: an unexpected increase of demand for a particular resource**
    - For the wanted content, there are not enough copies
    - How much time does it take for a user to locate the file?
    - How many messages will receive a node due to searches made by other nodes?
    - A generic protocol, based on TTL can be used

Structured Architectures

- Represent academic solutions for P2P
- Scope:
  - Successful search
  - Search time is performed in known boundaries
  - Proven scalability
- Approach: **DHT (Distributed Hash Table)**
  - Pairs (key, value) are stored
    - Key – file names
    - Value – file content or a pointer to a location
  - Each *peer* stores a set (key, value)
  - *Operations: find the node responsible for a Key*
    - Mapping *key – node*
    - Efficient routing to *insert/lookup/delete* operations associated with this node
    - A wide fluctuation of nodes is allowed

Structured Architectures

- Aspects: **content localization**
- Idea: The responsibility is distributed to multiple nodes of the coverage network, in an adaptive manner
- To each resource a unique key is associated via a *hash* function: h("Computer Networks Course")->7929; The range values of the *hash function* are distributed in P2P network
- Each node must "know" the location of at least one copy of a resource for which the hash function has values in its range
- Nodes can maintain their own cache with copies of each resource that they need to "know"

Structured Architectures

- Aspects: **routing**
- For each resource, a node that "knows" the resource must be accessed through the shortest path
- Structured P2P systems approaches differ by the routing strategy
- The nodes of the system forma a distributed structure that can be: ring, tree, hypercube, etc.
- It provides an API for distributed hash tables (DHT - Distributed Hash Table)
  - Giving a key *k*, the API will return the IP address of the node responsible for the *k* key value

Structured Architectures

- **Implementations**
  - Chord [MIT]
  - Pastry [Microsoft Research UK, Rice University]
  - Tapestry [UC Berkeley]
  - Content Addressable Network (CAN) [UC Berkeley]
  - SkipNet [Microsoft Research US, Univ. of Washington]
  - Kademlia [New York University]
  - Viceroy [Israel, UC Berkeley]
  - P-Grid [EPFL Switzerland]

Structured Architectures

- **Loosely structured**
  - The nodes can estimate which nodes store the searched resources
    - Blind broadcasts are avoided
    - C*hain mode propagation* mechanism is used: each node takes local decisions regarding who will be the next node which will be queried
  - Search for a file involves using a key and a timeout mechanism
  - Example: **Freenet**

[https://en.wikipedia.org/wiki/Freenet]

Instruments

- P2P – framework for Android

  https://code.google.com/p/p2p-communication-framework-for-android/

- p2psim – simulator for p2p protocol

  http://pdos.csail.mit.edu/p2psim/

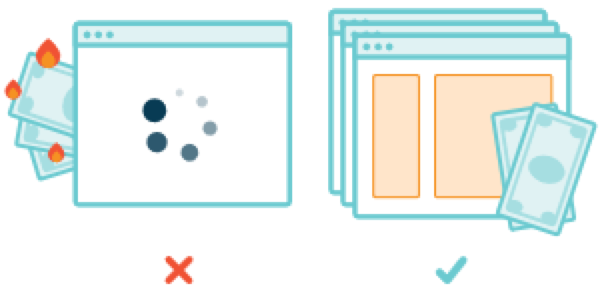
- Instruments and protocols for P2P:

  http://en.wikibooks.org/wiki/The_World_of_Peer-to-Peer_%28P2P%29/Networks_and_Protocols/Other_Software_Implementations

- " IPFS is the Distributed Web" - https://ipfs.io/

Instruments

- “ IPFS is the Distributed Web” - https://ipfs.io/
  - **A peer-to-peer hypermedia protocol to make the web faster, safer, and more open**



HTTP is inefficient and expensive

HTTP downloads a file from a single computer at a time, instead of getting pieces from multiple computers simultaneously. With video delivery, a P2P approach could save 60% in bandwidth costs.

IPFS makes it possible to distribute high volumes of data with high efficiency. And zero duplication means savings in storage.

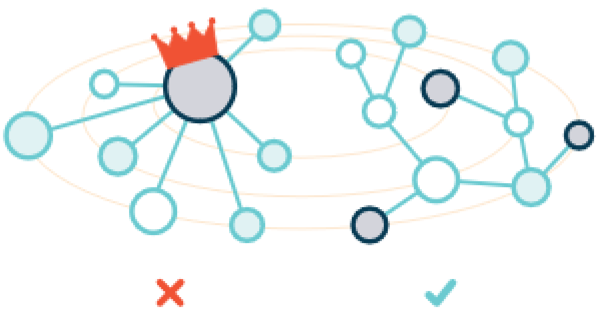
Humanity's history is deleted daily

The average lifespan of a web page is 100 days. Remember GeoCities? The web doesn't anymore. It's not good enough for the primary medium of our era to be so fragile.

IPFS provides historic versioning (like git) and makes it simple to set up resilient networks for mirroring of data.

Instruments

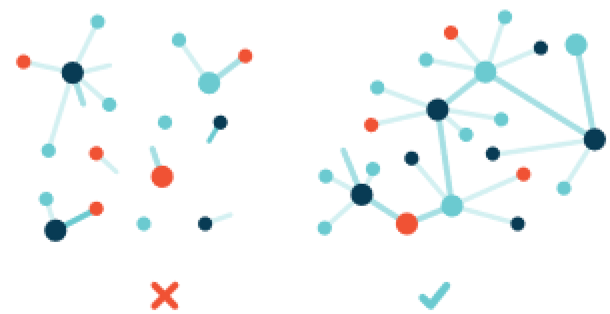
- " IPFS is the Distributed Web" - https://ipfs.io/
  - **A peer-to-peer hypermedia protocol to make the web faster, safer, and more open**



The web's centralization limits opportunity

The Internet has been one of the great equalizers in human history and a real accelerator of innovation. But the increasing consolidation of control is a threat to that.

IPFS remains true to the original vision of the open and flat web, but delivers the technology which makes that vision a reality.



Our apps are addicted to the backbone

Developing world. Offline. Natural disasters. Intermittent connections. All trivial compared to interplanetary networking. The networks we're using are so 20th Century. We can do better.

IPFS powers the creation of diversely resilient networks which enable persistent availability with or without Internet backbone connectivity.

Instruments

- “ IPFS is the Distributed Web” - https://ipfs.io/

Let's take a look at what happens when you add files to IPFS:

Each file and all of the **blocks within it** are given a **unique fingerprint** called a **cryptographic hash**.

Each **network node** stores only content it is interested in, and some indexing information that helps figure out who is storing what.

IPFS **removes duplications** across the network and tracks **version history** for every file.

When **looking up files**, you're asking the network to find nodes storing the content behind a unique hash.

Every file can be found by **human-readable names** using a decentralized naming system called **IPNS**.

- https://github.com/ipfs/papers/raw/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf

Statistics

Global Consumer Internet Traffic 2005–2011

Consumer Internet Traffic 2005–2011							
	2005	2006	2007	2008	2009	2010	2011
By Sub-Segment (terabytes per month)							
Web, e-mail, file transfer	362,084	505,996	692,812	948,425	1,233,172	1,603,615	2,756,415
P2P	1,060,226	1,329,770	1,772,403	2,379,025	3,111,891	4,040,403	5,269,360
Gaming	66,844	91,943	133,367	188,680	250,574	318,212	386,832
Video Communications	11,629	15,575	24,932	36,638	47,173	66,101	92,453
VoIP	10,965	23,035	39,339	57,653	75,575	92,815	110,456
Internet Video to PC	53,074	174,427	484,027	838,154	1,232,461	1,726,114	2,331,908
Internet Video to TV	0	12,727	110,692	353,095	620,197	936,580	1,342,482
By Geography (TB per month)							
North America	534,236	618,765	917,365	1,287,026	1,698,700	2,242,841	2,861,772
Western Europe	334,600	505,329	814,015	1,281,041	1,856,310	2,515,070	3,458,721
Asia Pacific	565,782	819,072	1,201,277	1,742,834	2,315,755	3,049,294	4,663,774
Japan	60,080	98,747	147,733	223,120	319,788	436,057	556,631
Latin America	19,917	33,755	57,083	90,765	130,466	189,992	268,559
Central Eastern Europe	40,773	59,097	86,196	122,272	165,387	222,895	294,901
Middle East and Africa	9,435	18,708	33,904	54,613	84,637	127,689	185,549

Statistics

Table 8. Global Consumer Internet Traffic, 2011–2016

Consumer Internet Traffic, 2011–2016							
	2011	**2012**	**2013**	**2014**	**2015**	**2016**	**CAGR 2011–2016**
By Network (PB per Month)							
Fixed	20,121	29,095	35,943	45,372	57,991	74,247	30%
Mobile	402	879	1,717	3,116	5,213	8,313	83%
By Subsegment (PB per Month)							
File sharing	6,013	7,403	9,153	11,569	14,758	18,892	26%
Internet video	10,423	16,880	20,904	26,722	34,755	45,280	34%
Web, email, and data	3,863	5,422	7,274	9,783	13,119	17,583	35%
Online gaming	77	115	170	251	404	630	52%
Voice over IP (VoIP)	147	154	159	163	169	174	3%
By Geography (PB per Month)							
North America	5,394	8,041	9,022	10,294	12,499	15,646	24%
Western Europe	5,132	7,463	9,311	11,822	14,796	18,233	29%
Asia Pacific	8,220	11,795	15,266	20,204	26,515	34,553	33%
Latin America	780	1,196	1,833	2,835	4,352	6,506	53%
Central and Eastern Europe	817	1,211	1,686	2,398	3,392	4,699	42%
Middle East and Africa	180	328	542	935	1,649	2,783	73%
Total (PB per Month)							
Consumer Internet traffic	20,523	29,974	37,660	48,488	63,204	82,560	32%

Source: Cisco VNI, 2012

Statistics

Table 15. Global consumer internet traffic, 2017-2022

Consumer Internet Traffic,2017-2022	2017	2018	2019	2020	2021	2022	CAGR 2017-2022
By Network (EB per Month)							
Fixed	67	86	111	141	179	225	27%
Mobile	10	16	25	36	50	68	47%
By Subsegment (EB per Month)							
Internet video	56	77	105	140	184	240	34%
Web, email, and data	12	15	19	23	27	31	22%
Online gaming	1	3	4	7	11	15	59%
File sharing	8	7	7	7	7	7	-3%
By Geography (EB per Month)							
Asia Pacific	27	38	54	74	99	130	37%
North America	25	32	40	50	61	74	25%
Western Europe	12	15	19	24	30	37	25%
Central and Eastern Europe	5	7	9	11	15	19	29%
Middle East and Africa	3	4	6	9	13	18	46%
Latin America	5	6	8	9	11	14	21%
Total (EB per Month)							
Consumer Internet traffic	77	102	136	177	229	293	31%

Source: Cisco VNI, 2018

Statistics

Global Consumer Peer-to-Peer Traffic 2005–2011

Consumer Peer-to-Peer Traffic 2005–2011							
	2005	**2006**	**2007**	**2008**	**2009**	**2010**	**2011**
By Geography (TB per month)							
North America	381,746	378,538	462,356	560,817	673,083	852,483	1,080,979
Western Europe	223,519	304,988	411,057	540,032	757,818	991,817	1,330,885
Asia Pacific	391,235	550,664	762,276	1,074,759	1,401,028	1,811,094	2,327,648
Japan	28,621	42,883	58,463	87,446	117,967	154,868	206,803
Latin America	8,732	14,358	23,247	37,284	53,587	80,043	117,731
Central Eastern Europe	22,075	31,009	43,117	59,928	79,589	106,543	141,282
Middle East and Africa	4,297	7,329	11,886	18,759	28,819	43,553	64,033
Total (TB per month)							
Peer-to-Peer Traffic	1,060,226	1,329,770	1,772,403	2,379,025	3,111,891	4,040,403	5,269,360

Statistics

Table 10. Global Consumer File-Sharing Traffic, 2011–2016

Consumer File Sharing, 2011–2016							
	2011	**2012**	**2013**	**2014**	**2015**	**2016**	**CAGR 2011–2016**
By Network (PB per Month)							
Fixed	5,967	7,337	9,093	11,524	14,658	18,698	26%
Mobile	46	66	84	106	142	194	33%
By Subsegment (PB per Month)							
P2P file transfer	4,656	5,401	6,234	7,314	8,627	10,215	17%
Other file transfer	1,357	2,002	2,942	4,315	6,174	8,677	45%
By Geography (PB per Month)							
North America	785	935	1,119	1,349	1,636	2,006	21%
Western Europe	1,609	1,845	2,154	2,552	2,918	3,181	15%
Asia Pacific	2,997	3,789	4,794	6,201	8,100	10,660	29%
Latin America	210	266	347	485	786	1,276	43%
Central and Eastern Europe	363	504	690	937	1,266	1,699	36%
Middle East and Africa	49	64	73	105	96	71	8%
Total (PB per Month)							
Consumer file sharing	6,013	7,403	9,177	11,629	14,801	18,892	26%

Source: Cisco VNI, 2012

Table 10. Global Consumer File-Sharing Traffic, 2015–2020

Consumer File Sharing, 2015–2020							
	2015	**2016**	**2017**	**2018**	**2019**	**2020**	**CAGR 2015–2020**
By Network (PB per Month)							
Fixed	5,942	5,909	5,829	5,713	5,616	5,939	0%
Mobile	22	28	29	29	29	35	9%
By Subsegment (PB per Month)							
P2P file transfer	4,798	4,550	4,224	3,840	3,438	3,633	-5%
Other file transfer	1,166	1,388	1,634	1,902	2,207	2,340	15%
By Geography (PB per Month)							
Asia Pacific	2,335	2,269	2,186	2,098	2,004	2,098	-2%
North America	1,015	1,137	1,260	1,371	1,478	1,576	9%
Western Europe	1,124	1,105	1,096	1,075	1,053	1,131	0%
Central and Eastern Europe	829	763	691	646	621	666	-4%
Latin America	554	573	558	514	454	463	-4%
Middle East and Africa	107	91	68	39	34	39	-18%
Total (PB per Month)							

File Sharing
This category includes traffic from P2P applications such as BitTorrent and eDonkey, as well as web-based file sharing. Note that a large portion of P2P traffic is due to the exchange of video files, so a total view of the impact of video on the network should count P2P video traffic in addition to the traffic counted in the Internet video-to-PC and Internet video-to-TV categories. Table 10 shows the forecast for consumer P2P traffic from 2015 to 2020. Note that the P2P category is limited to traditional file exchange and does not include commercial video-streaming applications that are delivered through P2P, such as PPStream or PPLive.

- http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

Summary

- **Peer-to-peer (P2P) paradigm**
  - Preliminaries
  - Definitions
  - Characteristics
  - Application types
  - Infrastructures
  - Instruments

Bibliography

- P2P Networking and Applications, John F. Buford, Heather Yu, Eng Keong Lua, 2009, Elsevier
- http://www.cisco.com/en/US/docs/cable/serv_exch/serv_control/broadband_app/protocol_ref_guide/01_p2p.pdf
- http://pdos.csail.mit.edu/p2psim/
- Statistics: http://www.hbtf.org/files/cisco_IPforecast.pdf
- Statistics: https://ec.europa.eu/digital-single-market/en/news/cisco-visual-networking-index-forecast-and-methodology-2011%E2%80%932016
- Statistics: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html
- http://en.wikibooks.org/wiki/The_World_of_Peer-to-Peer_%28P2P%29/Networks_and_Protocols/Other_Software_Implementations
- https://www.kirsle.net/blog/entry/skype-switched-to-the-msn-messenger-protocol

Questions?