DCN-2 - Tutorial work by professor.

Data Communications and Networks (New York University)

Name: Jain,Sid **Date: 02/22/18**
Section: 001

Assignment 1

Assignment Layout (25%)

- **Assignment is neatly assembled on 8 1/2 by 11 papers.**
- **Cover page with your name (last name first followed by a comma then first name), username and section number with a signed statement of independent effort is included.**
- **Answers to Questions 1 to 6 are correct.**
- **File name is correct.**

Answers to Individual Questions:

(100 points total, all questions weighted equally)

- **Assumptions provided when required.**

Total in points (100 points total): _____

Professor's Comments:

Affirmation of my Independent Effort: Jain,Sid

Problem 1 – HTTP Protocol Analysis

a. Use a packet sniffer (e.g., Wireshark) to capture the sequence of ASCII characters that are sent and received by a web browser as a result of a request of your choice to a web server. Provide a screen capture of these sequences and add carriage return and line feed characters as needed to improve readability.

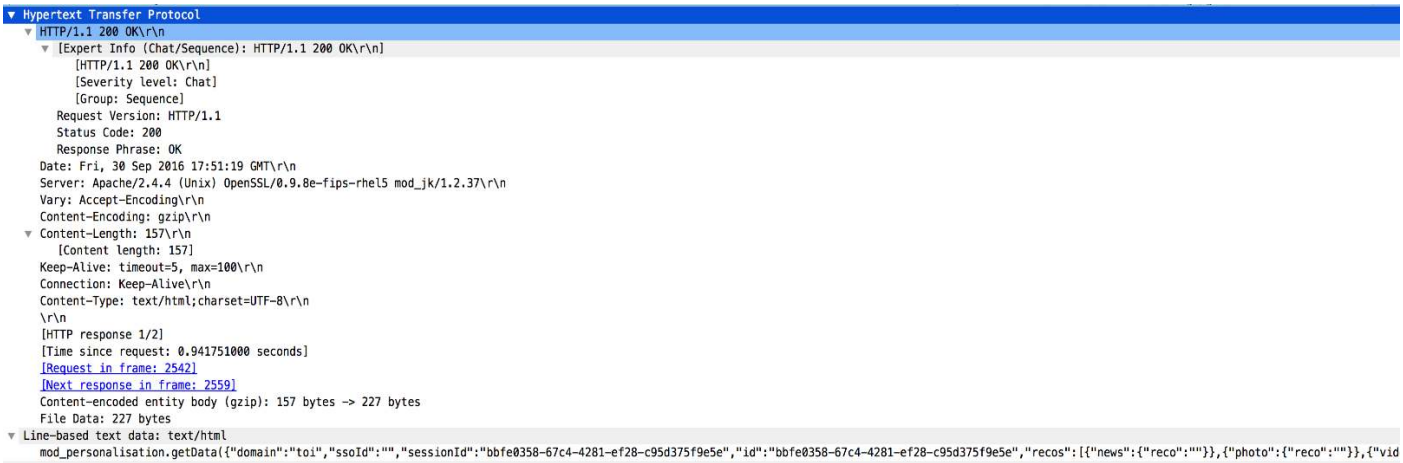
b. Identify the complete URL of the document requested, the HTTP protocol version for both the request and response, the operating system that the web browser is running on, and the kind of web server that answered the request.

c. Is the HTTP connection persistent?
d. Which web browser sent the request and why is it important for the server to know this information?
e. Was the request successful and, if so, what type of document was received by the server?

Answer 1

(a) Request Capture

Response Capture



(b) The URL requested:

http://userreco.indiatimes.com/userreco/v2/reco?
domain=toi&mediaType=news,photo,video&callback=mod_personalisation.getData

HTTP Protocol version for request: HTTP 1.1
HTTP Protocol version for response: HTTP 1.1

Operating System: Macintosh; Intel Mac OS X 10_11_6

Web Brower: Chrome/53.0.2785.116
Browser Platform: AppleWebKit/537.36
Browser Platform Details: (KHTML, like Gecko)

Server: Apache/2.4.4 (Unix) OpenSSL/0.9.8e-fips-rhel5 mod_jk/1.2.37

(c) Yes, the HTTP connection is persistent because it has Keep-Alive parameter in the response message.

(d) Chrome is the web browser which sent the request.
It is important for the server to know this information as this information allows the web site to customize content according to the capabilities of a particular web browser.

(e) Yes, the request was successful. The type of document that was received by the client was text/HTML.

Problem 2 – Web Application Architectures
Explain how Web architectures were developed and refined to increasingly support applications with informational, interactive, transactional, and delivery requirements? Please relate to specific architectures, their corresponding protocols, and describe the improvements that were made over time.

Answer 2

The application architecture is designed by the application developer and dictates how the application is structured over the various end systems. A developer will likely draw on one of the two architectural paradigms used in modern network applications:

1. **Peer-to-Peer**

In a P2P architecture, there is minimal (or no) reliance on dedicated servers in data centers. Instead the application exploits direct communication between pairs of intermittently connected hosts, called peers. The peers are not owned by the service provider, but are instead desktops and

laptops controlled by users, with most of the peers residing in homes, universities, and offices. Because the peers communicate without passing through a dedicated server, the architecture is called peer-to-peer. Many of today's most popular and traffic-intensive applications are based on P2P architectures. These applications include file sharing (e.g., BitTorrent), peer-assisted download acceleration (e.g., Xunlei), Internet Telephony (e.g., Skype), and IPTV (e.g., Kankan and PPstream). We mention that some applications have hybrid architectures, combining both client-server and P2P elements. For example, for many instant messaging applications, servers are used to track the IP addresses of users, but user-to-user messages are sent directly between user hosts (without passing through intermediate servers).

One of the most compelling features of P2P architectures is their self-scalability. For example, in a P2P file-sharing application, although each peer generates workload by requesting files, each peer also adds service capacity to the system by distributing files to other peers.

2. Client/Server

In a client-server architecture, there is an always-on host, called the server, which services requests from many other hosts, called clients. A classic example is the Web application for which an always-on Web server services requests from browsers running on client hosts. When a Web server receives a request for an object from a client host, it responds by sending the requested object to the client host. Note that with the client-server architecture, clients do not directly communicate with each other; for example, in the Web application, two browsers do not directly communicate. Because the server has a fixed, well-known address, and because the server is always on, a client can always contact the server by sending a packet to the server's IP address. Some of the better-known applications with a client-server architecture include the Web, FTP, Telnet, and e-mail.

Often in a client-server application, a single-server host is incapable of keeping up with all the requests from clients. For example, a popular social-networking site can quickly become overwhelmed if it has only one server handling all of its requests. For this reason, a data center, housing a large number of hosts, is often used to create a powerful virtual server. The most popular Internet services—such as search engines (e.g., Google and Bing), Internet commerce (e.g., Amazon and e-Bay), Web-based email (e.g., Gmail and Yahoo Mail), social networking (e.g., Facebook and Twitter)— employ one or more data centers. A data center can have hundreds of thousands of servers, which must be powered and maintained. Additionally, the service providers must pay recurring interconnection and bandwidth costs for sending data from their data centers.

Problem 3 – Real-Time Messaging Applications

List at least four mainstream real-time messaging applications. Document the protocols they use (along with references to corresponding IETF RFCs) and explain in detail how they differ. Please provide references and/or links to all documentation sources used to answer this question.

Answer 3

iMessage: It is an instant messaging service developed by Apple Inc. The iMessage protocol is based on the Apple Push Notification Service (APNs)—a proprietary, binary protocol. It sets up a Keep-Alive connection with the Apple servers. Every connection has its own unique code, which acts as an identifier for the route that should be used to send a message to a specific device. The connection is encrypted with TLS. iMessage provides end-to-end encryption.

Reference:
https://en.wikipedia.org/wiki/IMessage

WhatsApp: It uses a customized version of the open standard Extensible Messaging and Presence Protocol (XMPP). Multimedia messages are sent by uploading the image, audio or video to be sent to an HTTP server and then sending a link to the content along with its Base64 encoded thumbnail (if applicable).

WhatsApp follows a 'store and forward' mechanism for exchanging messages between two users. When a user sends a message, it first travels to the WhatsApp server where it is stored. Then the server repeatedly requests the receiver acknowledge receipt of the message. As soon as the message is acknowledged, the server drops the message; it is no longer available in database of server.

Reference:
https://en.wikipedia.org/wiki/WhatsApp

BlackBerry Messenger: It is a proprietary Internet-based instant messenger and video telephony application. Messages sent via BlackBerry Messenger are sent over the Internet and use the BlackBerry PIN system, it is their proprietary protocol. The communications are not end-to-end encrypted.

Reference:
https://en.wikipedia.org/wiki/BlackBerry_Messenger

Yahoo Messenger: The Yahoo! Messenger Protocol (YMSG) is the underlying network protocol used by the Yahoo! Messenger instant messaging client. Yahoo! Instant Messenger supports many features beyond just messaging, including off-line messaging, file transfer, chat, conferencing, voice chat, webcams and avatars. The YMSG protocol communicates between the client application, and a server, using a TCP/IP connection on port 5050 by default. With the exception of the login authentication details, data sent over a YMSG connection is not encrypted.

Reference:
https://en.wikipedia.org/wiki/Yahoo!_Messenger_Protocol

The RFCs for instant messaging are:

- 2000-02 RFC 2778: A Model for Presence and Instant Messaging (Informational)
- 2000-02 RFC 2779: Instant Messaging / Presence Protocol Requirements (Informational)
- 2002-07 RFC 3339: Date and Time on the Internet: Timestamps (Proposed Standard)
- 2004-08 RFC 3859: Common Profile for Presence (CPP) (Proposed Standard)
- 2004-08 RFC 3860: Common Profile for Instant Messaging (CPIM) (Proposed Standard)
- 2004-08 RFC 3861: Address Resolution for Instant Messaging and Presence (Proposed Standard)
- 2004-08 RFC 3862: Common Presence and Instant Messaging (CPIM): Message Format (Proposed Standard)
- 2004-08 RFC 3863: Presence Information Data Format (PIDF) (Proposed Standard)

Problem 4 – Email Applications

List at least five mainstream email applications. Document the protocols they use (along with references to corresponding IETF RFCs) to send and receive emails and explain in detail how they differ. Please provide references and/or links to all documentation sources used to answer this question

Answer 4

Gmail: It supports both IMAP4 and POP3 protocols to receive emails. TLS or SSL are required for security. POP uses port 995 and for IMAP port 993 is used for incoming mail and port 465 or 587 for outgoing mail. For SSL Port 465 is used and for TLS Port 587 used.
References:
coderoman.com/2013/01/pop-imap-and-smtp-settings-for-gmail/
en.wikipedia.org/wiki/Gmail
support.google.com

Outlook: It supports both IMAP4 and POP3 protocols to receive emails. SSL is required for incoming emails and TLS is required for outgoing emails. POP uses port 995 and IMAP uses port 993. Only TLS can be
used for outgoing connection and both ports 25 and 587 can be used.
References:
https://en.wikipedia.org/wiki/Outlook.com
https://support.office.com/en-us/article/POP-and-IMAP-settings-for-Outlook-Office-365-for-business-7fc677eb-2491-4cbc-8153-8e7113525f6c

iCloud: IMAP is supported for receiving email and SMPT for sending emails. SSL or TLS are required. Port 993 is used for IMAP and Port 587 is used for SMTP.
Reference:
https://support.apple.com/en-us/HT202304

Yahoo! Mail: It supports both IMAP and POP3 for incoming emails and SMTP for outgoing emails. IMAP uses port 993 and port 995 for POP. Port 465 or 587 can be used for SMTP. SSL is required for incoming emails and both SSL or TLS can be used for outgoing emails.
References:
https://help.yahoo.com/kb/SLN4724.html
https://help.yahoo.com/kb/SLN4075.html

	Protocol to send (User agent)	Mail Server	Protocol to download emails on client machine
GMail	SMTP	SMTP [RFC2821]	IMAP/POP
Outlook	SMTP	SMTP [RFC2821]	IMAP/POP
iCloud	SMTP	SMTP [RFC2821]	IMAP/POP
Yahoo! Mail	SMTP	SMTP [RFC2821]	IMAP/POP

Differences:

- When using POP, clients typically connect to the e-mail server briefly, only as long as it takes to download new messages. When using IMAP, clients often stay connected as long as the user interface is active and download message content on demand. For users with many or large messages, this IMAP usage pattern can result in faster response times.
- The POP protocol requires the currently connected client to be the only client connected to the mailbox. In contrast, the IMAP protocol specifically allows simultaneous access by multiple clients and provides mechanisms for clients to detect changes made to the mailbox by other, concurrently connected, clients.
- IMAP supports flags on the server to keep track of message state: for example, whether or not the message has been read, replied to, or deleted.
- POP is a much simpler protocol, making implementation easier.
- POP mail moves the message from the email server onto your local computer, although there is usually an option to leave the messages on the email server as well. IMAP defaults to leaving the message on the email server, simply downloading a local copy.

The RFCs related to email are:

- RFC 821 – Simple Mail Transfer Protocol (Obsoleted by RFC 2821)
- RFC 822 – Standard for the Format of ARPA Internet Text Messages (Obsoleted by RFC 2822)
- RFC 1035 – Domain names, Implementation and specification
- RFC 1123 – Requirements for Internet Hosts, Application and Support (Updated by RFC 2821, RFC 5321)
- RFC 2142 – Mailbox Names for Common Services, Roles and Functions
- RFC 2821 – Simple Mail Transfer Protocol (Obsoletes RFC 821, Updates RFC 1123, Obsoleted by RFC 5321)

- RFC 2822 – Internet Message Format (Obsoletes RFC 822, Obsoleted by RFC 5322)
- RFC 3696 – Application Techniques for Checking and Transformation of Names
- RFC 4291 – IP Version 6 Addressing Architecture (Updated by RFC 5952)
- RFC 5321 – Simple Mail Transfer Protocol (Obsoletes RFC 2821, Updates RFC 1123)
- RFC 5322 – Internet Message Format (Obsoletes RFC 2822)
- RFC 5952 – A Recommendation for IPv6 Address Text Representation (Updates RFC 4291)
- RFC 6530 – Overview and Framework for Internationalized Email (Obsoletes RFC 4952, 5504, 5825)

Reference:
https://en.wikipedia.org/wiki/Email_address

Problem 5 – Network Management Utilities

a. Explain what the following utilities are used for: traceroute, ping, nslookup, ipconfig, dig?
b. Identify at least three more utilities and explain what they are used for.
c. For each one of the utilities introduced in 5.a. and 5.b., provide a detailed usage scenario along with corresponding screenshots as needed to fully document your example.

Answer 5

traceroute:

The Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route, one's packets follow (or finding the miscreant gateway that's discarding your packets) can be difficult. traceroute utilizes the IP protocol `time to live' field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host.

```
traceroute: Warning: www.manipal.edu has multiple addresses; using 54.230.36.129
traceroute to www.manipal.edu (54.230.36.129), 64 hops max, 52 byte packets
 1  10.0.0.1 (10.0.0.1)  4.469 ms  4.686 ms  5.052 ms
 2  96.120.72.209 (96.120.72.209)  22.248 ms  26.855 ms  19.342 ms
 3  68.85.129.129 (68.85.129.129)  20.545 ms  34.909 ms  19.529 ms
 4  68.85.63.209 (68.85.63.209)  14.982 ms  19.407 ms  33.204 ms
 5  be-20-ar03.plainfield.nj.panjde.comcast.net (68.85.62.17)  27.065 ms  18.359 ms  25.424 ms
 6  be-33659-cr02.newyork.ny.ibone.comcast.net (68.86.90.21)  20.165 ms  24.563 ms  24.561 ms
 7  hu-0-13-0-0-pe02.111eighthave.ny.ibone.comcast.net (68.86.84.238)  23.231 ms  19.736 ms  20.263 ms
 8  as16509-1-c.111eighthave.ny.ibone.comcast.net (50.242.148.114)  19.817 ms  19.378 ms  20.601 ms
 9  * * *
10  * * *
11  54.240.217.61 (54.240.217.61)  17.588 ms  19.645 ms *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
31  * * *
32  * * *
33  * * *
34  * * *
35  * * *
36  * * *
37  * * *
38  * * *
39  * * *
40  * * *
41  * * *
42  * * *
43  * * *
44  * * *
45  * * *
46  * * *
47  * * *
48  * * *
49  * * *
50  * * *
51  * * *
52  * * *
53  * * *
54  * * *
55  * * *
56  * * *
57  * * *
58  * * *
59  * * *
60  * * *
61  * * *
62  * * *
63  * * *
64  * * *
```

ping:

The ping utility uses the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway. ECHO_REQUEST datagrams (``pings'') have an IP and ICMP header, followed by a ``struct timeval'' and then an arbitrary number of ``pad'' bytes used to fill out the packet.

```
C:\Users\arpit>ping www.google.com

Pinging www.google.com [172.217.3.100] with 32 bytes of data:
Reply from 172.217.3.100: bytes=32 time=7ms TTL=56
Reply from 172.217.3.100: bytes=32 time=10ms TTL=56
Reply from 172.217.3.100: bytes=32 time=7ms TTL=56
Reply from 172.217.3.100: bytes=32 time=13ms TTL=56

Ping statistics for 172.217.3.100:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 13ms, Average = 9ms
```

Nslookup:

Nslookup is a program to query Internet domain name servers. Nslookup has two modes: interactive and non-interactive. Interactive mode allows the user to query name servers for information about various hosts and domains or to print a list of hosts in a domain. Non-interactive mode is used to print just the name and requested information for a host or domain.

```
C:\Users\arpit>nslookup www.google.com
Server:  FIOS_Quantum_Gateway.fios-router.home
Address:  192.168.1.1

Non-authoritative answer:
Name:    www.google.com
Addresses:  2607:f8b0:4006:80e::2004
          172.217.3.100


C:\Users\arpit>
```

ipconfig:

ipconfig is a utility that communicates with the IPConfiguration agent to retrieve and set IP configuration parameters. It should only be used in a test and debug context. Using it for any other purpose is strongly discouraged. Public API's in the SystemConfiguration framework are currently the only supported way to access and control the state of IPConfiguration.

```
C:\Users\arpit>ipconfig

Windows IP Configuration


Ethernet adapter Tunngle:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : fios-router.home
   Link-local IPv6 Address . . . . . : fe80::e9d5:9c9d:958f:35a7%5
   IPv4 Address. . . . . . . . . . . : 192.168.1.168
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 13:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
```

arp:

The arp utility displays and modifies the Internet-to-Ethernet address translation tables used by the address resolution protocol (arp(4)). With no flags, the program displays the current ARP entry for hostname. The host may be specified by name or by number, using Internet dot notation.

```
C:\Users\arpit>arp -a

Interface: 192.168.1.168 --- 0x5
  Internet Address      Physical Address      Type
  192.168.1.1           48-5d-36-78-c3-0a     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Netstat:

Netstat (Network Statistic) command display connection information, routing table information etc.

```
C:\Users\arpit>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:50379        DESKTOP-MQJ5BC5:55268  ESTABLISHED
  TCP    127.0.0.1:55260        DESKTOP-MQJ5BC5:65001  ESTABLISHED
  TCP    127.0.0.1:55268        DESKTOP-MQJ5BC5:50379  ESTABLISHED
  TCP    127.0.0.1:65001        DESKTOP-MQJ5BC5:55260  ESTABLISHED
  TCP    192.168.1.168:55291    52.165.231.192:https   ESTABLISHED
  TCP    192.168.1.168:55766    lga25s55-in-f14:https  TIME_WAIT
  TCP    192.168.1.168:55787    216.165.47.12:https    TIME_WAIT
  TCP    192.168.1.168:55788    216.165.47.12:https    TIME_WAIT
  TCP    192.168.1.168:55789    216.165.47.12:https    TIME_WAIT
  TCP    192.168.1.168:55796    144.2.0.1:https        TIME_WAIT
  TCP    192.168.1.168:55797    130:https              TIME_WAIT
  TCP    192.168.1.168:55798    144.2.1.1:https        TIME_WAIT
  TCP    192.168.1.168:55799    144.2.0.1:https        TIME_WAIT
  TCP    192.168.1.168:55800    144.2.1.1:https        TIME_WAIT
  TCP    192.168.1.168:55803    192.229.163.180:https  TIME_WAIT
  TCP    192.168.1.168:55805    144.2.193.69:https     TIME_WAIT
  TCP    192.168.1.168:55808    HERCULES:https         LAST_ACK
  TCP    192.168.1.168:55809    HERCULES:https         LAST_ACK
  TCP    192.168.1.168:55812    137.116.77.120:https   TIME_WAIT
  TCP    192.168.1.168:55819    13.107.21.200:https    TIME_WAIT
  TCP    192.168.1.168:55822    13.107.21.200:https    ESTABLISHED
  TCP    192.168.1.168:55823    13.107.21.200:https    ESTABLISHED
  TCP    192.168.1.168:55824    13.107.21.200:https    ESTABLISHED
  TCP    192.168.1.168:55826    qj-in-f189:https       ESTABLISHED
  TCP    192.168.1.168:55827    lga25s62-in-f4:https   ESTABLISHED
  TCP    192.168.1.168:55828    a173-223-201-241:https  ESTABLISHED
```

Problem 6 – Overlay Networks

a. How is peer churn managed in P2P applications such as file-sharing, conferencing, and content distribution?
b. Provide specific examples of P2P applications, explain how they specifically handle churn, and estimate the performance improvements achieved in each case. Please provide references and/or links to all documentation sources used to answer this question.

Answer 6

(a) In P2P systems, a peer can come or go without warning. To handle peer churn, we require each peer to track (that is, know the IP address of) its first and second successors. We also require each peer to periodically verify that its two successors are alive, by periodically sending ping messages to them and asking for responses.

(b) The Gnutella and Kad overlays provide a keyword-search function that allows users to locate files that have the keyword in the filename. Gnutella performs these lookups using an

unstructured topology, while Kad uses a DHT. BitTorrent, on the other hand, forms an overlay to facilitate the rapid transfer of large files to a large number of peers.

Gnutella: Gnutella is currently one of the most popular P2P systems, with more than 1 million simultaneous users. It uses a two-tier overlay structure, similar to FastTrack and eDonkey. Most peers are leaf peers, while a small fraction of peers act as ultrapeers. The leaf peers connect to a handful of ultrapeers, which index the content of the leaves. Searches spread out from ultrapeers using a modified expanding-ring search.

Kad: Kad is a Kademlia-based P2P search network used by the eMule P2P file-sharing software. To our knowledge, Kad is the largest deployed DHT, with more than 1 million simultaneous users. Similar to other DHTs, each peer has a persistent, globally-unique identifier of length b bits (in Kad's case b = 128 bits). Keywords are hashed to b bits and stored on the peer with the closest matching identifier. Each peer stores a structured routing table pointing to other nodes in the network such that the expected number of overlay hops to perform any lookup is O(log n), where n is the population size.

BitTorrent: BitTorrent is a popular P2P application for distributing very large files (100+ MB) to a large group of users. Unlike most P2P systems, which form one large overlay, BitTorrent has a distinct overlay for each file. To download a file, peers exchange different blocks of the content until each peer has the entire file. The peers locate one another using a rendezvous point, called a tracker, whose address is provided to new members out of band. Each new peer contacts the tracker via HTTP, periodically sends an update of its progress to the tracker, and informs the tracker when it departs. Each peer may receive the entire file across multiple sessions, i.e., it may obtain only a subset of blocks in one session and resume the download later. Many peers may give up without downloading the whole file. The tracker logs its interactions with peers, providing the arrival and departure times of peers with one second resolution. While the tracker records the arrival time of all peers, the departure time is only captured for peers which depart gracefully.

Source:
http://conferences.sigcomm.org/imc/2006/papers/p19-stutzbach2.pdf