GOVERNMENT COLLEGE OF ENGINEERING, JALGAON
(An Autonomous Institute of Government of Maharashtra)
Department of Computer Engineering

Name: Vaibhav Bharat Sontakke	PRN No: 1942207
Class: T. Y. B. Tech.	Batch: L4
Date of Performance: ___________	Date of Completion: _____________
Subject: DSL	Sign. of Teacher with Date: __________

Experiment No.: 10

Title: To Study BitTorrent and End System Multicast.

Theory:

INTRODUCTIO N T O BIT T ORRENT

BitTorrent is a technology/protocol which makes the distribution of files, especially large files, easier and less bandwidth consuming for the publisher. This is accomplished by utilizing the upload capacity of the peers that are downloading a file. A considerably increase in downloaders will only result in a modest increase in the load on the publisher hosting the file.

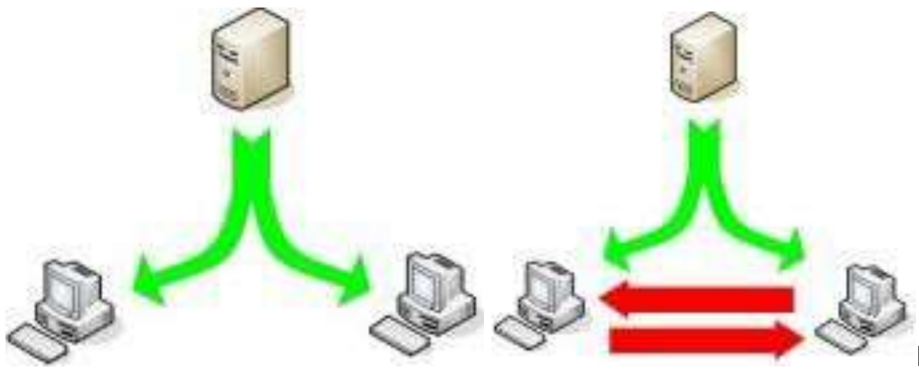


Figure 1 – The basic flow of the BitTorrent protocol.

The illustration in 1 shows the basic flow of BitTorrent. The figure on the left shows a client-server approach to download. The peers download from the server simultaneous. If we assume the upload capacity of the server is the same as the download capacity of a peer, the time for the download to finish will be two times the time if only one peer where downloading from the server. The figure on the right shows an approach similar to BitTorrent. By splitting the file and send one part to each peer, and let the peers download the part they are missing from each other, both download time and load on the server is reduced. Of course, the BitTorrent protocol is much more sophisticated than this simple example, but this shows the basic idea.

THE HISTORY OF BITTORRENT

BitTorrent is by far the most popular peer-to-peer programs ever. Analysis shows that it accounts for about 35% of all Internet traffic. How did it become so popular, and what makes it so special? In the summer of 2001, Cohen released his first beta version of BitTorrent. In 2002 he presented it at a conference. His goal with this software was to give people a quick and simple way of distributing and swapping Linux software online. But, as we all know, the movie-geeks soon saw the potential in the BitTorrent technology. In 2004, pirate copies of movies and TV-shows began dominating the BitTorrent traffic, and after that the growth has been explosive. It is projected that by the end of this year about 40 million

AREAS OF USAGE

Piracy and illegal distribution is the first thing brought to mind when you hear about peer-to peer file sharing. Not unreasonable, since the majority of traffic is illegal exchange of copyrighted material. But the BitTorrent protocol has features which makes it usable for totally legitimate purposes.

large software and patches can be done at higher speeds using BitTorrent .How often haven't you waited too long for the newest security update from Windows to be finished downloading? Using the BitTorrent protocol the spreading of this kind of software can be much quicker and more satisfactory for the end users. Within an organization one can also use the protocol to distribute applications and updates more rapidly. Improvements in the protocol facilitate this usage

THE BITTORRENT ARCHITECTURE:

The BitTorrent architecture normally consists of the following entities: - a static metainfo file (a "torrent file") - a 'tracker' - an original downloader ("seed") - the end user downloader ("leecher") It's all about the torrent file, the centralized tracker and the associated swarm of peers. The centralized tracker provides the different entities with an address list over the available peers. These peers will then contact each other to download pieces of the file from each other. The first step in publishing a file using BitTorrent is to create a metainfo file from the file that you want to publish. The metainfo file is called a "torrent". The torrent file contains the filename, size, hashing information and the URL of the "tracker". The "torrent" is needed by anyone who wants to download the file the torrent is created from. The torrent file can be distributed by e-mail, IRC, http etc. The torrent is created by using a free program. This functionality is also commonly included in the BitTorrent clients. To download or "seed" a file, you need a BitTorrent client. The BitTorrent client is a free application that administrates the download procedure. There are several different BitTorrent clients available. They all support the standard BitTorrent protocol, but may differ and be incompatible with each other regarding certain features. A BitTorrent download is started by opening the torrent file in the BitTorrent client. The tracker keeps a log of peers that are currently downloading a file, and helps them find each other. The tracker is not directly involved in the transfer of data and CO455 Distributed Operating System Lab does not have a copy of the file. The tracker and the downloading users exchange information using a simple protocol on top of HTTP. First, the user gives information to the tracker about which file it's downloading, ports it's listening on etc. The response from the tracker is a list of other users which are downloading the same file and information on how to contact them. This group of peers that all share the same torrent represents a 'swarm'. When creating the torrent file from the original file, the original file is cut into smaller pieces, usually 512 KB or 256Kb in size. The SHA-1 hash codes of the pieces are included in the torrent file. The downloaded data are verified by computing the SHA-1 hash code and comparing it to the SHA-1 code of the corresponding piece in the torrent file. This way the data is checked for errors and it guarantees to the users that they are downloading the real thing. Whenever a piece is downloaded and verified, the downloading peer reports to the other peers in the swarm about its new piece. This piece is now available for other peers. We will now go deeper into piece selection and other details of the protocol.

DECENTRALIZED TRACKER

With one centralized tracker the BitTorrent network is not very fault tolerant. If the tracker goes down the file will no longer be available, since there are no way the peers could know about each other. One centralized tracker also makes the network vulnerable to denial of service attacks. In May 2005 a version 4.1 was released by Cohen and the big newsflash was support for a decentralized tracker. In this new version (a beta release), all you have to do is to publish your .torrent-file on a webpage, blog etc., and no tracker specification is necessary. Use of a dedicated tracker is still supported, and the publisher can choose the preferred mode. The tracker is distributed in the sense that every client or node in the network now acts a lightweight tracker. The solution is based on distributed hash tables (DHTs). This makes it possible to share files with minimal resources, but no guarantees can be made as respect to reliability. In addition to making it easier for a more novice user to share his files through a simple website or a blog, one of the other driving forces for the trackerless mode was economy. A website owner might have to pay (to the web server owner) according to the traffic on his website. The more people who want to download your file, the larger the traffic the website generates the lager the bill will be for the owner. Cohen thinks this is unfair as compared to broadcast by a traditional TV-station: whether

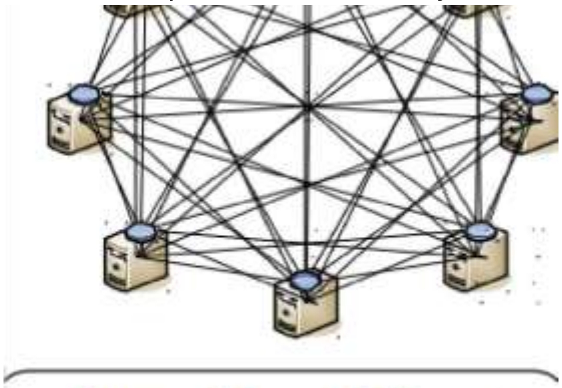


Figure 2.1 – BitTorrent with a Decentralized tracker structure

five people

DISTRIBUTED HASH TABLES (DHTS)

A DHT is a decentralized distributed system. The system consists of a set of participating nodes and a set of keys. The DHT performs the function of a hash table. Key and value pair can be stored and a value can be looked up if the correct key is provided. What separates a DHT from an ordinary hash table is that the storage and lookups are distributed among the nodes (machines) in the network. All nodes are peers that can join and leave the network freely. DHTs makes provable guarantees about performance despite the seemingly chaos created by random joining and leaving peers. Any DHT protocol emphasizes the following characteristics;

- **Decentralization**: the system is collectively created and maintained by the nodes without any central coordination.
- **Scalability**: the system performs efficiently even with thousands or millions of nodes. 🎬 **Fault tolerance**: the system should be reliable even with nodes continuously joining, leaving or failing.

HOW TO USE BITTORRENT

BitTorrent is a file-sharing protocol that lets you download content directly from other groups of people. Unlike HTTP, the download speeds vary, because you aren't receiving data from a dedicated server, instead you're downloading from other users' computers.

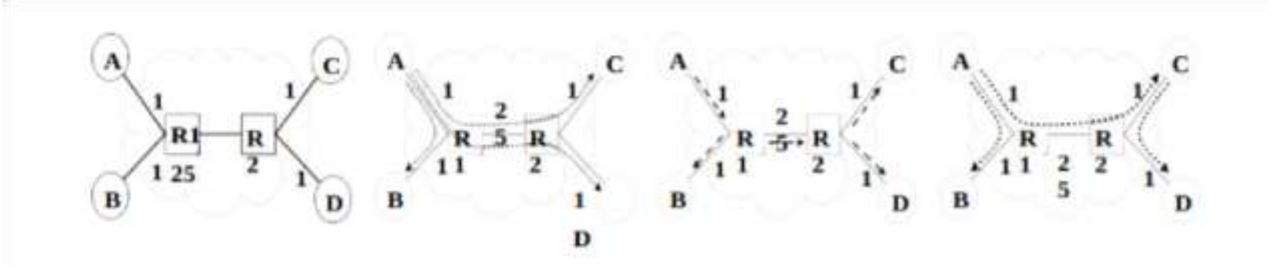
Here’s a simple 3-step guide:

1. **Get a BitTorrent client**: First you need a BitTorrent client, a program that will enable you to connect to other users (or peers) and thereby download the files you want to. Our favorites are: Windows – uTorrent, Mac – Azureus, Linux – BitTorrent. Click through one of the links and download and install the client.
Tip: Want to download Torrents anonymously? Try BTguard, a great way to download torrents securely.

2. **Find a torrent**: Now that you have a BitTorrent client installed, you'll want to find torrents you like and download them. For that you have to visit a BitTorrent directory or use a BitTorrent search engine like Torrentz.
For example, open www.torrentz.com in your browser and search for anything (Linux for example). If you see something you like, click on one of the 'Download Locations' and download the .torrent file from the site.

3. **Start the download:** Once downloaded, double-click the file and it will open and start downloading in your BitTorrent client. Some clients like Azureus will ask you to specify a location to save the file.

END SYSTEM MULTICAST
Figure 3: Example to illustrate End System Multicast



(a) **(b)** **(c)** **(d)**

ESM used a peer-to-peer network to distribute video data across all viewers of a video stream. It constructs an overlay tree to distribute data, and continuously optimizes this tree to minimize end-to-end latency. The root of the tree is the source of the broadcast. This is typically the machine that encodes the video data. This machine sends a stream of data packets to the nodes at the first level of the tree. Each of those nodes then forwards the data to the nodes connected to them, and so on, such that all nodes in the system receive the data stream. ESM allowed any user with a DSL or broadband connection or higher to broadcast good quality video to a large number of people. Since it is a peer to peer network, a broadcaster need only broadcast the video to one person for any number of people to view it. Due to the nature of peer to peer multimedia networks, skips in playback or buffering can occur.

To illustrate the di erences between IP Multicast, naive unicast and End System Multicast using Figure 3. Figure 3(a) depicts an example physical topology, where R1 and R2 are routers, while A, B, C and D are end systems. Link delays are as indicated. R1 − R2 represents a costly transcontinental link, while all other links are cheaper local links. Further, let us assume 'A' wishes to send data to all other nodes. Figure 3(b) depicts naive unicast transmission. Naive unicast results in significant redundancy on links near the source (for example, link A − R1 carries three copies of a transmission by A), and results in duplicate copies on costly links (for example, link R1 − R2 has two copies of a transmission by A). Figure 3(c) depicts the IP Multicast tree constructed by DVMRP. DVMRP is the classical IP Multicast protocol, where data is delivered from the source to recipients using an IP Multicast tree composed of the unicast paths from each recipient to the source. Redundant transmission is avoided, and exactly one copy of the packet traverses any given physical link. Each recipient receives data with the same delay as though A were sending to it directly by unicast Figure 3(d) depicts an "intelligent" overlay tree that may be constructed using the End System Multicast architecture. The number of redundant copies of data near the source is reduced compared to naive unicast, and just one copy of the packet goes

Given that End System Multicast tries to push functionality to the edges, there are two very different ways this can be achieved: peer-to-peer architectures and proxy-based architectures. In a peer to-peer architecture, functionality is pushed to the end hosts actually participating in the multicast group. Such architectures are thus completely distributed with each end host maintaining state only for those groups it is actually participating in. In a proxy-based architecture on the other hand, an organization that provides value added services de- ploys proxies at strategic locations on the Internet. End hosts attach themselves to proxies near them, and receive data using plain unicast, or any available multicast media. While these architectures have important di erences, fundamental to both of them are concerns regarding the performance penalty involved in disseminating data using overlays as compared to solutions that have native multicast support. Thus, an end system in our paper refers to the entity that actually takes part in a self- organization protocol, and could be an end host or a proxy.

Our evaluation of End System Multicast targets a wide range of group communication applications such as audio and video conferencing, virtual classroom and multi-party network games. Such applications typically involve small (tens of members) and medium sized (hundreds of members) groups. While End Sys- tem Multicast may be relevant even for applications which involve much larger group sizes such as broadcasting and content distribution - particularly in the context of proxy-based architectures.

Conclusion:

In this Practical, we studied about the Bittorrent and End System Multicast

Miss. Archana Chitte
Name & Sign of Course Teacher