A measurement-based study of the Direct Connect Network

Skand Singh Gupta, Shriram Rajagopalan
Department of Computer Science, University of California, Santa Barbara

***Abstract*—The Direct Connect peer to peer network has been steadily increasing in its popularity over the last 3 years. There are more than 20 different implementations of hubs and 30 flavours of clients. This work is a comprehensive end user study of the dynamics of the hub's behaviour, the user's communication and data sharing patterns, the loop holes in the system and protection mechanisms employed in such systems.**

I. Introduction

The Direct Connect (DC) network based on the erstwhile Neo-Modus Direct Connect (NMDC) and the more recent Advanced Direct Connect (ADC) protocols is a peer to peer (P2P) file sharing system similar to Napster. Like Napster, peers in a DC network connect to a 'hub' in order to find the content available in the network as well as to interact with other peers. However, unlike Napster, DC allows users to run their own hubs thereby promoting the formation of a large number of tightly knit "community" file sharing systems. Another key difference between DC and Napster is that DC does not store any files on the hubs. Hubs act only as a gateway for content and peer discovery, but the actual content is transferred directly between the peer without the involvement of the hub. This relieves the owner of the hub from the responsibility of the data being transferred using the hub.

These two differences were the attracting force behind the rapid adoption of DC and has also lead to more than 20 different hub and more than 30 different client implementations[1]. There also have been a number of instances of organizations banning DC, along with other p2p file sharing systems, in order to reduce the strain on the network infrastructure. For instance the University of California at Santa Barbara blocks access to DC from it's student residences. However, very few studies have empirically measured the size of the DC network, the availability of content and the amount of network traffic generated by DC.

[1]http://www.dslreports.com/faq/dc

Ideally one would like to measure and model the traffic flowing between the peers this network. However, we being an end user, we decided on using alternative mechanisms for gathering information about the DC network. We make following contributions:

- We have studied the Direct Connect Protocol and the network in detail and have identified techniques for measuring the system from an end host by identifying and exploiting the flaws in the systems to bypass hub restrictions for collecting data.
- We used these techniques to measure the DC network over two weeks and gathered data to analyze the DC network and present our findings in this paper.

The rest of the paper is organized as follows: Section II describes the related work in the area. Section III gives an overview of how the system works. Section IV and V describe the metrics used for measurement and the methodology involved, respectively. We present the results of our study in Section VI and conclude the paper with Section VII.

II. Related Work

Sen et. al. [1] measure flow-level information collected at multiple border routers across a large ISP network for three P2P systems: FastTrack, Gnutella and Direct Connect. However, their focus was evaluating the traffic generated by these P2P systems and they don't report any specific details about the dynamics of DC network itself. Besides, this study was conducted in 2002 and there haven't been any subsequent studies covering DC.

Kosunen et. al. [5] compare Gnutella and DC on parameters such as bandwidth usage, search speed, search accuracy and scalability. However this is a small scale study which covers only 30 hubs. On the other hand our measurement is based on periodic probing of more than 6000 hubs.

III. How The System Works

Direct Connect is a 'hybrid' P2P network and comprises of hubs the clients connect to, in order to discover

other clients and content. The functionality of the DC Hub is conceptually very similar to that of a physical network hub. It multi-casts search and public chat messages between all the clients connected to it. However, exchange of private messages and content download takes place between the peers with no involvement of the hub. In the latter scenario, the hub only facilitates the initial peering by disclosing a peer's IP address to the requester and acting as a identity validation server. A hub can also redirect a client to other hubs and the mechanism is similar to website redirection.

Each hub can optionally have a series of rules and regulations forming the entry barrier for the users connecting to that network. While most hubs mandate a minimum amount of data that needs to be shared by each user, before permitting the user to search and download content, hubs can also have rules for type of content, need for user registration etc.

The Direct Connect clients build a Tiger Tree Hash (TTH) based index of the files being shared. This indexing is used primarily for error correction during file transfer and can also include the file names and file types. Clients can connect to the hubs in two modes, active and passive. Active mode clients are those that are not running behind a proxy, i.e. they have a route able address, and can be connected to, from any other peer. Passive mode clients are the opposite of active mode clients and generally receive lower priority from other peers during file search or download requests. When a user wants to download a file or browse another peer's share of files, the client sends a

```
$ConnectToMe <TargetPeerNickName>
```

to the hub which is relayed to the target peer as

```
$ConnectToMe <SendingPeerIP>.
```

Unlike other p2p file sharing systems, the target peer is requested to connect to the requester. This gives the target peer the ability to share the contents with only the clients that it desires. However, this behavior only holds good for clients connected in active mode. When a passive mode client wants to download files from a peer, a

```
$RevConnectToMe <TargetPeerNickName>
```

is sent to the hub. The hub, after checking that the target peer is connected in active mode, sends the target peer's IP to the requester. While this mechanism allows peers behind a proxy to connect and download file from DC network, it also opens a way for clients to fake share sizes and we exploited it for our measurements (see Section V). However, this mechanism can not be be used to launch a DOS on a client because the users can dictate the number of upload/download connections depending on their bandwidth by controlling the number of upload/download slots configured in the client.

Search requests from a client are sent as regular expressions to the hub, which multi-casts them to other clients. While this increases the amount of control traffic that the hub needs to send and that each client needs to handle, it is necessary since the hubs themselves don't store any content. This technique of multi-casting all the messages can be exploited to measure the amount of traffic being generated by a hub and also to monitor the type of content that is being shared by the users. We use this mechanism to measure the control traffic generated in hubs that we were monitoring (see Section V).

On identifying a file to download, the user can chose to download it from a client or can reissue a search with the TTH of the target file (obtained from search result), to find out whether other clients have the same file. This enables a user to download a file piecemeal from different peers thereby increasing the speed of the download.

A number of bots have been written to extend the functionality of DC. Bots are pre-programmed clients or utilities that can be used to perform a variety of tasks. One could create a bot to continuously monitor the chat and search messages to see if any data of interest to the user turns up and automatically download it. Bots are also used for linking two DC networks with the approval of the hub operators (aka. administrators). These bots log into both the hubs, and serve as a logical router to relay chat, search, user join and leave messages between two hubs. When two hubs are linked in such a manner, a user in one hub can see the users of the other hub, search and download files from the peers of the other hub, etc.

Bots are also used to do lot of administrative tasks for the hub operators. Most of the hubs have a minimum data share requirement which some users bypass by faking their share size. Operators generally download the TTH index file from random clients and browse through them. When a client purports to share 120 GB of data and its index file is unavailable or filled with garbage characters or is very small or if the share size is exactly $120 * 1024 * 1024 * 1024$, which never occurs in a 120 GB hard drive, the operator identifies him/her as a fake user and bans that client from the hub for fixed to unspecified length of time. Most hubs have dedicated bot users that perform the above sanity check while operators focus on removing users with banned content, content which they deem in appropriate to that hub. It is still possible to fake the file share despite these measures as provided

by our methodology in Section V.

In the wake of spam users, several hub implementations have developed anti spam and anti flooding mechanisms to maintain a good QoS for the network. Number of messages per client are capped to prevent spamming. Message sizes from client to are generally of fixed or approximately known sizes. Any attempt to misuse the connection by sending huge chunks of data results in the hub immediately closing the connection

In the recent past, certain security loopholes in the old client implementations were taken advantage of, by inviting these older clients to join a fake redirecting hub and redirecting all requests to targeted victims. This form of DOS attack vulnerability still exists. If malicious hub masqueraded as a legitimate one, and accumulated a critical user base mass, then it could very well redirect all requests, and launch the DOS on a victim.

IV. Metrics for Evaluation

We use following parameters to evaluate the DC network:

- *Hub Upload Bandwidth.* Hub upload bandwidth is a measure of amount of control traffic as a function of number of users in the hub. Since a large number of hubs are managed from stock PCs as opposed to server class machines. Measuring the upload bandwidth per user lets us estimate the bandwidth requirement for running a hub in order to support a large number of users.
- *Content Shared on DC.* By measuring the amount of content being shared in DC, we get an estimation of the relevance of DC as compared to other P2P file sharing systems.
- *User Persistence.* User persistence is defined as: *If a user is seen on-line during one of the probes, what is the probability of seeing the same user on-line at a subsequent probe.* Measuring user persistence is important because it is indicative of the availability of the content on DC. User persistence is measured as a ration of number of times a user was on-line vs. number of times the hub to which the user belonged was on-line and is given by:

$$p_n = \frac{X_n}{H_n} \tag{1}$$

  Where,
  p_n, persistence of a user n
  X_n, number of times the user was on-line
  H_n, number of times the hub was on-line

- *Hub Availability.* The hub availability measures the ratio of the number of times a hub was on-line over the number of times it was probed. Hub availability influences all the other parameters discussed above. Since most of the hubs are run from home PCs, it is important that we take into consideration the availability of the network itself before measuring other parameters.

V. Methodology

The architecture of DC posed some interesting challenges for our study. First and foremost is the need to find out all the DC networks that are operational. The multitude of unrelated DC networks with no central management makes it an interesting challenge. Fortunately for us there exist a few websites[2] which do a loose aggregation of public hubs. Anyone running a hub can register their hub to these websites thereby enabling discovery by the users. Apart from collecting the list of hubs, these aggregation websites probe the registered hubs periodically to gather the hubs' on line status and to collect other information such as number of users and the amount of data being shared. However, the interval between two probes can vary from a few hours to a couple of days. The unpredictable interval between two probes and the lack of confidence in the impartiality of their results lead us to design a probing mechanism of our own. However, we used these aggregation websites to make our initial list of known public hubs. Our bot gathered information about over 6000 DC networks by parsing information from these websites.

We modified a command line version of a DC client called microdc for sending the probes. This client was modified to act like a bot to log into a hub, download user information, monitor hub communications and exit the hub. In order to keep the interval between two probes small, we used the 30 node mutant cluster[3] to send out the probes. Each node was given the complete list of 6000 hubs but the hubs in each list were ordered in a way that the interval between two probes was not more than 80 minutes. This interleaving of the hubs also ensured that two successive probes to a hub went out from two different mutant node, reducing the probability of getting detected by a hub operator.

The hubs use the nick name and IP address tuple to distinguish one peer from another. Our modified client generates a random nick at every connection in order to create the illusion of a different user which reduces the chances of detection. Most of the hubs also enforce a minimum content size to be shared before a user is allowed to interact with other users in the hub. In order

[2]http://www.dchublist.com, http://www.hublist.org
[3]http://www.cs.ucsb.edu/ ravenben/ucsb/xcluster.html

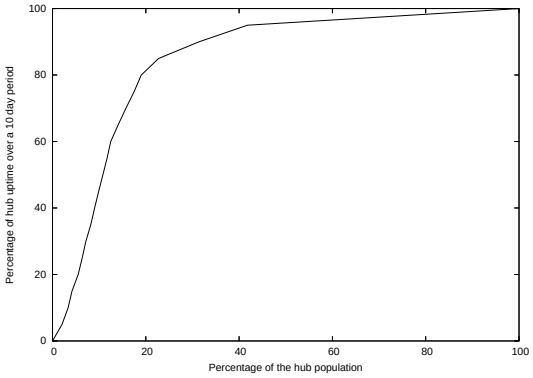

Fig. 1. CDF of hub uptime. The y-xis represents the hub uptime measured as a percentage of number of times the hub was online vs. number of times it was probed. The x-axis represents the percentage of hub population. More than 60% of the hubs are online for more than 90% of the time.

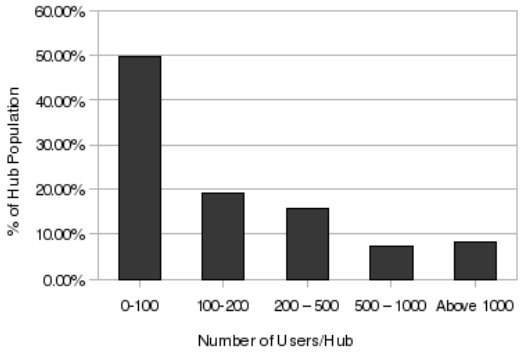

Fig. 2. User distribution. The x-axis represents the number of users and the y-axis represents the percentage of hubs with those many users. Most DC networks are small "community" networks with more than 80% of the hubs having less than 500 users.

to satisfy the minimum file share criteria, we downloaded a legitimate user's file index (index for a 120 GB hard drive) and used this as our file index whenever another user or an operator probed our client. In order to reduce the chances of getting exposed as a user faking the share, we used the passive mode connection to connect to each hub. A request to view our index of shared file was serviced promptly, but subsequent requests to download a file was ignored by feigning a lack of upload slots.

The hubs were monitored for a period of two continuous weeks. From the logs, we filtered out hubs which had no users throughout the observation period, hubs that needed prior registration for connection, private hubs that refused service and a small portion of hubs that required password based authentications. The cleaned data set had about 3000 hubs, with user ranges between 50 to 17000, data share sizes ranging from a few Mega Bytes to 15 Tera Bytes.

VI. Results

Figure 1 shows the CDF of hub uptimes. The graphs shows that over 60% of the hubs are online all the time. Given the assumption that most of the DC hubs are being operated from stock PCs, these numbers quite unexpected. This graph clearly shows that a large number of hubs are still being actively maintained by the operators.

Figure 2 shows that distribution of users per hub. The graph confirms the that most of the DC networks comprise of a relatively few users as compared to other P2P networks. Since DC doesn't impose it on the clients to connect to a centrally managed servers, the users tend to form networks around common interests.

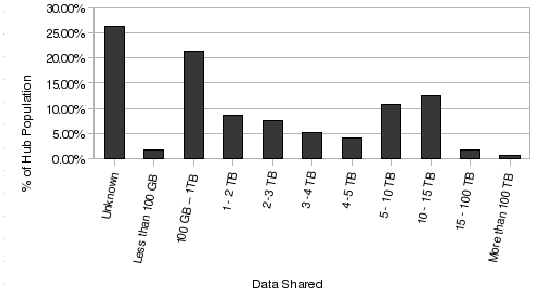

Fig. 3. Share distribution. The x-axis represents the amount of data shared and the y-axis represents the percentage of hubs. The share size of the hubs is relatively small compared to other p2p systems and is due to smaller user base of DC hubs.

The distribution of size of the data shared is plotted in Figure 3. The small size of DC network is also apparent from this graph with close to 50% of the hubs having less than 1Tb of data shared among its users. Approximately 2% of the hubs have share size over 100 Tb which translates to about 60 hubs in our sample set of 3000 hubs.

The user persistence is depicted in Figures 4 and 5. Figure 4 shows that the persistence of users sharing less than 200 Gb of data tend to hover around 20% mark. These are the users who stay in the hub for a brief interval of time just to download some content. More likely than not, these users do not upload a lot of content. Or even if they do so, they do for a short period of time. In contrast, users with share size greater than 1 Tb have lot more users who stay online for close to 100% of times. These users seem to be attracting other users into the hubs and are main contributors to the DC traffic.

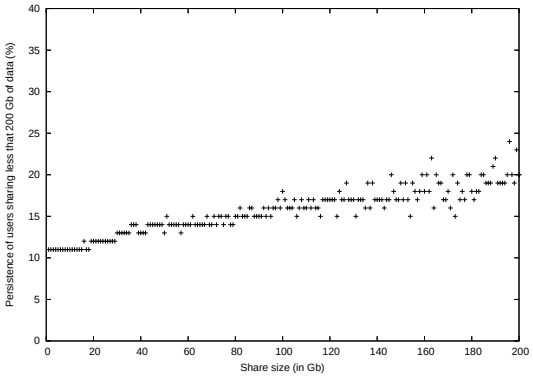

Fig. 4. User persistence for users sharing less than 200 Gb of data. The x-axis is the size of the share and the y-axis represents the persistence of user measured as a percentage of number of times the user was online against the number of time the hub was online. The persistence for most of the users sharing less than 200 Gb of data is less than 20%

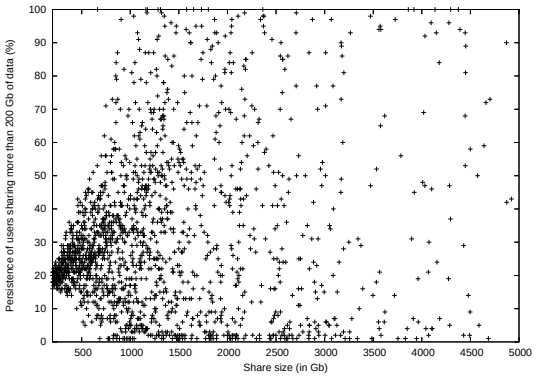

Fig. 5. User persistence for users sharing more than 200 Gb of data. There is lot more variation in the persistence of users sharing larger amount of data with a lot of users staying on line for 100% or little less.

In order to find the bandwidth requirements for each hub, we connect to the hub for a fixed duration of time and record all the traffic from the hub. The analysis of this data is represented by the graph in Figure 6. The bandwidth consumed by each hub is not directly proportional to the number of users in the hub, but is rather a function of the activity of the users in the network. Hubs with lesser number of users performing a lot of searches will record higher traffic than with hubs with lot of relatively inactive users. This can be justified by the fact that the hubs multi-cast each request to all the clients.

VII. Conclusions

We have studied the Direct Connect system in detail and have identified several behavioral patters that are not present in other peer to peer systems. Leeching is not as prevalent in DC as it is in other popular contemporary peer to peer networks like BitTorrent, Gnutella, etc. Also, unlike other systems,we have noticed a significant level of sophistication in the hub implementations like anti spam , anti flood mechanisms, primararily due to community efforts and a continuous monitoring of the system by volunteering operators to eliminate malicious users.

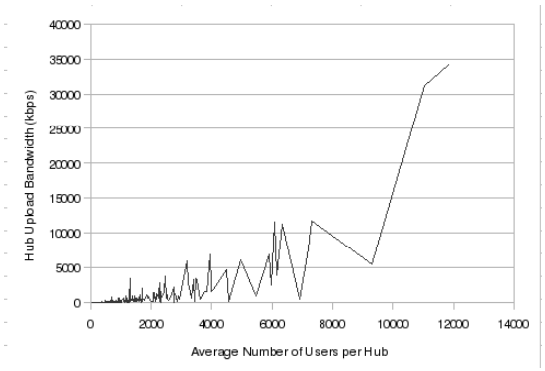

Fig. 6. Hub Upload Bandwidth. The x-axis shows the average number of users per hub and the y-axis shows the total data transferred by the hub to each user. The bandwidth required for the hub is not directly related to the number of users in the hub but more on how active the users are in hub.

References

[1] Subhabrata Sen, Jia Wang *Analyzing Peer-to-Peer Traffic Across Large Networks*, Second Annual ACM Internet Measurement Workshop, 2002

[2] Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble *A Measurement Study of Peer-to-Peer File Sharing Systems*

[3] Magnus Kolweyh and Ulrike Lechner *Towards P2P Information Systems* April, 2006

[4] Georgios Portokalidis, Evangelos P. Markatos, Manolis Marazakis *Studying and Bridging of Peer-to-Peer File Sharing Systems*, Institute of Computer Science, Foundation for Research and Technology Hellas, Technical Report, October 2002

[5] Matti Kosunen, Anna Larmo, Pablo Cesar *Content Search in Ad hoc P2P Networks*

[6] *http://www.nongnu.org/microdc/*