COMP90015 Distributed Systems Project 2 Report

Renlord Yang - rnyang - 541368   Jun Min Cheong - jmcheong - 542339

1 Ways to reduce bandwidth consumption

Using UDP instead of TCP By substituting TCP with UDP for the transport layer, it is possible to reduce the amount of bandwidth overhead attributed to the ACK responses required by the TCP protocol. By utilising UDP, UDP packets will be transmitted to the destination peer at best effort basis without any requirement to acknowledge the receipt of a UDP packet. This results in significant bandwidth savings as it is no longer necessary for peers to send ACK-SYN messages. However, as UDP packets are limited to containing 64kB worth of data, it is potentially necessary to augment the protocol to cater for splitting of image data into multiple UDP packets.

IP Multicasting As the project requires a peer to be able to transmit a stream of images to many peers at any one time, it is technically possible to perform IP multicasting if the UDP protocol is adopted. In addition, the protocol will have to be modified to support IP multicasting. By relying on one stream to multicast to many peers instead of many streams to stream the video stream to many peers, multicasting has the potential to save up to (N - 1) amount of streams, hence resulting in signifcant bandwidth savings. However, switches and routers must be able to support IP multicasting for this to occur, so ideally IP Multicasting is only potentially possible within local networks as oppose to streaming across world wide web.

Lossy Compression for Image Gzip algorithm is a lossless compression algorithm often used to to compress files and documents that requires information to be complete (ie. the compressed file must not lose any information). Lossless compression however is not entirely necessary for video streams as the information contained in image files does not have to be perfectly preserved. To further reduce the bandwidth overhead, it would be ideal to adopt a lossy compression algorithm such JPEG. The use of the lossy compression algorithm will substantially reduce the size of the images per frame, as such facillitate a better quality of service at an imperceptible degradation in video streaming quality.

2 Security Issues and Potential Fixes

Image Size Exploitation The protocol stipulates that it is possible to supply any dimensions for the image size. However, as there is no restrictions on the dimensions of the image size, an adversary can maliciously fix an arbitrary big image size that will cause severe problems on the connected peer's terminal as the behaviour of the JFrame window would be uncertain and will most likely consume most, if not all of the bandwidth capacity of the receiving peer. A possible fix would entail fixing a realistic arbitrary limit on all image sizes to prevent malicious users from exploiting this flaw. All peers can then check if the *startstream* message contains dimensions which may exceed the limits.

Non-Terminating Streams As the protocol utilises TCP as its transport layer protocol, images are transmitted from one peer to another via streams. To effectively terminate a stream, it is important that peers end *JSON Strings* with a newline character or else receiving peers may be blocked indefinitely. As the protocol does not clearly state if a stream should be ended with either a newline character or any other meta-character and it does not stipulate the minimum incoming buffer size, it is difficult to maintain consistency across different implementations of the protocol. For our implementation, if a newline character is not used to indicate the end of stream, the receiving peer will be blocked indefinitely. A possible fix would either involve the use of a meta-character (ie. newline character) to indicate end of stream or it would be prudent if a minimum buffer size be specified.

Unintentional Blocking on Peers Blocking is potentially a serious threat for this protocol. As the number of connected peers grow, it is likely a peer would encounter another peer that is running at maximum capacity. When many busy peers are connected to other busy peers, peers can potentially end up into a deadlock, where peers are left waiting for a response from the recipient peer while the recipient peer might be busy servicing other connected peers. In situations like that, deadlocks in a peer to peer network may be uncontainable and may spread across to other peers, resulting in the need to close all connections and turning them on again to resolve all deadlocks. A potential fix would involve setting the maximum number of peers that may be connected at any one time. An even better approach would involve the consideration of the bandwidth capacity before determining the maximum number of peers that may be connected at any given time.

3 Scalability and Rate Limiting

Purpose of Rate Limiting Rate limiting should be done to ensure that peers are served equally and the level of service received is of consistent. When the number of connected peers are low, the concept of rate limiting may seem counter-intuitive as it hinders the ability to send higher frame rates. The ability to control the amount of frames transmitted per second is crucial to the viability of the protocol as it ensures that no other peer will be flooded with an excessive amount of Image messages to be processed in the receving buffer. In an ideal scenario, rate limiting guarantees that peers that are connected to many peers will receive the same level of video stream quality from incoming video streams from other connected peers as the rate of incoming frames are limited by the design of the protocol. Without rate limiting, frames will be sent at such a rate where a subset of the connected peers would be capable of saturating the bandwidth capacity of the destination peer, thus resulting in an unfair bandwidth distribution scheme for all other connected peers. Rate Limiting resolves this by enforcing a rate limit for each connected peer (ie. 10 images per second with a thread suspension time of 100ms for each connected peer).

Scalability of Rate Limiting Despite the advantages of rate limiting, it is potentially possible for a peer to experience quality degradation. If there are too many incoming video streams from connected peers, the amount of bandwidth available would be insufficient to service all incoming video streams. When this occurs, video streams with connected peers would be severely compromised to the extent, the video stream will be perceptibly unappealing as TCP will throttle down the tranmission rate of the images due to network congestion. As such a scenario is possible, the maximum number of connected peers is highly dependant on the bandwidth capacity of a given peer. For example, it is possible for a peer to be capable of receiving up to 10 video streams while another peer can only receive 3 video streams due to the difference in bandwidth capacity. Support for higher numbers of connected peers is beyond the control of rate limiting.

4 Possible Improvements

Detailed Peer Negotiations As we have discussed earlier that a lot more useful information may be sought from peers to facilitate a better quality of service for users. As a result of this, it is highly advisable that peers engage in a negotitation phase prior to streaming images to other peers. The propose negotiation phase should exchange informations such as bandwidth capacity, number of currently connected peers, image size to be transmitted and the format of the image. With these information, peers would then be able to determine if a connection should be accepted or re-negotitate for a more suitable image size in the event that the bandwidth capacity was not at a desirable state. The peer negotation should also be extended so that at any time, peers may re-negotate transmission policies to be able to receive more connections when necessary. Peers can also use the negotiation information to refuse connection if it is obvious that the likelihood of blocking occuring is very high.

Usage of Different Image Format As discussed earlier, it is prudent for peers to utilise different image formats that might suit the need of different peers. For example, for peers who have very low bandwidth capacity, it would be a good idea to be able to use Bitmaps that hold image data in black and white. Other potential image formats that may be served include JPEG, PNG and JPG.

Blocking Handling Policy As dicussed earlier, blocking is a big problem for this protocol. It would be highly advisable that a blocking handling policy be included in the protocol as a counter-measure to handle the potential of deadlocks within the peer to peer network. An example of a counter-measure would involve peers restarting their connection if a timeout was reached, thus resolving all deadlocks and repeat the connection establishment procedure from start to finish again.