Web Access over Delay Tolerant Networks

Zoebir Bong, Peng Boon Sng, and Chai Kiat Yeo

***Abstract*—This paper proposes an application to support web access over Delay Tolerant Networks (DTN). The application enables web browsing in the absence of end-to-end IP connectivity and under challenged network environments where there are long delays or intermittent network connectivity. The latter cannot be managed by conventional Internet transport protocol such as TCP. DTN is resilient to delay and can avail itself to the opportunistic connectivity due to its store and forward capability. The absence of IP connectivity is thus transparent to the web user.**

***Index Terms*—Web browsing, delay tolerant networks.**

I. Introduction

The Internet with the TCP/IP protocol has become a daily essential in a connected world. There are times that the access network experienced outages or where connectivity is lost due to local disaster, partial infrastructure failure or war. Under such circumstances, an alternative network such as Delay Tolerant Networks (DTN) [1] – [3] can kick in to enable some form of connectivity to the Internet. DTN has been designed to withstand intermittent connectivity and enable opportunistic contact communication. It operates as an overlay across existing heterogeneous network infrastructure by encapsulating application data through its Bundle protocol [4]. Bundle is the basic data unit in DTN. Unlike IP, DTN does not require end-to-end connectivity and it uses a store-and-forward approach to transfer data to the destination in a hop-by-hop manner. DTN sits in the application layer of the network stack and the data transfer is carried out by the convergence layers. By using a TCP convergence layer, DTN can reuse existing TCP/IP infrastructure to perform reliable data transfer between two DTN nodes. In this way, DTN can enable communication in challenged networks experiencing frequent disruption. A DTN implementation is required to provide other functionalities like link management, convergence layer, neighbour discovery, bundling, etc. In this project, the DTN2 reference implementation (version 2.7) [5] is used.

The objective of this project is to design and build an application which leverage on the DTN technology to allow users to browse the web using their preferred browsers even though the user may not have direct access to his IP network. The DTN network will connect the client to a server which has access to the Internet. As such, the existing web contents which are accessible by IP networks do not need to be downloaded to a DTN network.

Manuscript received March 10, 2012; revised April 5, 2012

The authors are with School of Computer Engineering Nanyang Technological University, Nanyang Avenue, S 639798, Singapore.

II. Architecture of Web Access over DTN

The system architecture of our proposed web access over DTN is shown in Fig. 1 and comprises two main parts, namely, the client and the server. The client resides in a DTN network overlaid on top of the IP network. The server is situated in an area where IP connectivity is available. HTTP messages from the client are then forwarded via DTN nodes until it reaches the server.

III. Client System Design

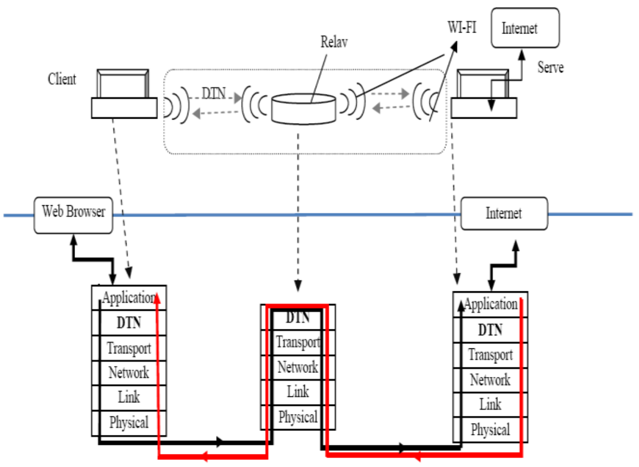

Fig. 1. System architecture

The client is generally a desktop or a laptop with WI-FI coverage to neighbouring DTN nodes and has no direct Internet connectivity. The primary function of the client system is to send bundled requests and receive bundled responses through wireless connection in DTN. Fig. 2 shows the client system design.

A. Client Sending

In order for the client to send the web request as DTN bundles, the application must have cached a copy of the requests and bundled it as a DTN bundle. The cached copy of the request can be acquired by setting up a socket and setting the web browser as a proxy to its local host on the particular socket. A unique and unused port number is used for this application, 8555. Web browser that sets its proxy to local and port 8555 will then have its request cached and ready to be sent by the application. Files containing the cached request together with its process ID are created to minimize memory space required. These files are named after its process ID with an extension of .req. Multiple threads are run to support multiple requests through either multi-tabs or parallel web request.

The ID of the thread (Process ID) will have to be sent together with the request. This is due to DTN corrupting the original Internet TCP sequence number with its own DTN

TCP sequence number. Hence an identifier is needed to match the correct thread which sends the request since multiple requests are sent and multiple replies may be received. The inevitable matching of IDs will lead to higher processing burden.

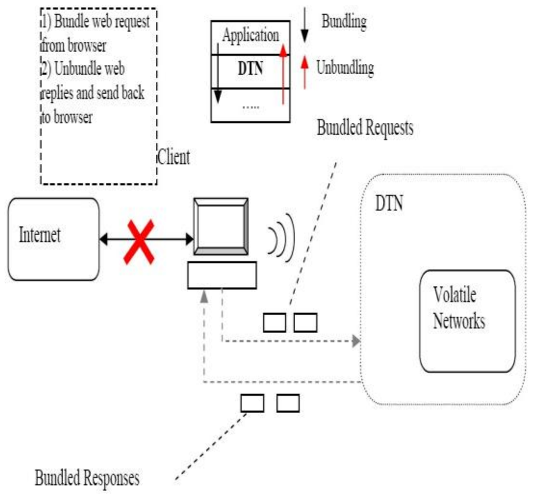

Fig. 2. Client system design

Each thread will then bundle its request and send it to the server by stating the location of its file. Relay nodes are not specified in the application as it could be declared in the DTN configuration file. The bundling process uses the existing DTN API [6] which has existing functions for bundling a packet into DTN bundle. Each thread will be dedicated to each request and thus the thread will only be killed once it has received its subsequent response.

B. Client Receiving

During the procedure of receiving a bundle, there will be one main receive thread and multiple other threads waiting for their own replies after sending its request. The task of the main receive thread is to receive a DTN bundle and unbundle it before storing it as a file in the database. Mutual exclusion is used while creating and writing the file into the database to prevent deadlock and misreading of files among the threads. Once done, the main receive thread will continue waiting for new bundles to arrive and perform the same task repeatedly for the entire duration of the application.

The files received will be named after the subsequent Process ID of the request and with extension .res. Response from the Internet is stored in the file. The other threads will then keep a lookout for files named after their own Process ID with .res extension in the database. If the file is found, the thread will send its subsequent response data to the web browser.

An HTML file of a web page usually consists of more than one object, for example its images, icons and other files. Each individual thread will serve one object request in the Internet. By doing this way, AJAX can be supported by the application. Each serving thread will be killed once they have received their subsequent reply. The replied file will also be deleted after sending it to the browser.

The reason of having a main thread is that the DTN implementation does not support multiple threads for receiving a bundle. Only one thread could be used for receiving DTN bundles. Thus multiple threads are not allowed to receive DTN bundles, rather they are to poll for its own replies that reside in the database written by the main thread.

C. Server System Design

Similar to client, the server has wireless connection and it is connected to the Internet directly to send web request and receive web responses. Vice versa to the client, server unbundles requests before sending it to Internet and bundles the responses received before sending them back to the client through DTN. The server thus acts as a proxy for the client to send the web requests and receive responses on behalf of the client to/from the Internet. Fig. 3 shows the system server design.

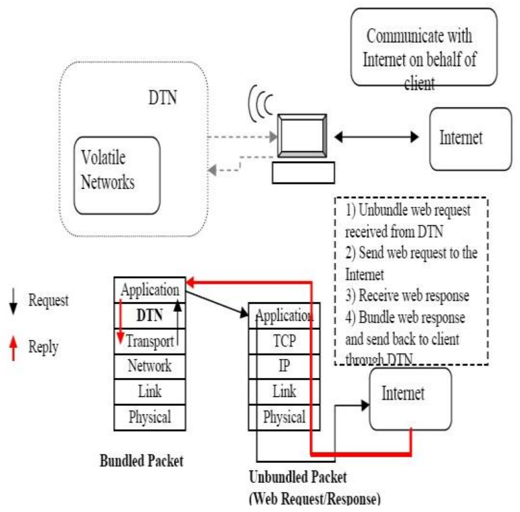
Communicate with Internet on behalf of client
DTN
Volatile Networks
Internet
1) Unbundle web request received from DTN
2) Send web request to the Internet
3) Receive web response
4) Bundle web response and send back to client through DTN
Request
Reply
Application
DTN
Transport
Network
Link
Physical
Application
TCP
IP
Link
Physical
Internet
Bundled Packet
Unbundled Packet (Web Request/Response)

Fig. 3. Server system design

Functions	Results	Remarks
Search Bar	✓	Search bar on Firefox, Opera and URL search bar on Chrome working
Downloading Files	✓	
Multi-tab	✓	
Back Button	✓	
Support Intermittent Connectivity	✓	Intermittent Connectivity are performed by switching off the relay nodes in between Client and Server

Fig. 4. Web browser functionality table

The Server application is also split into two parts, namely the main and its child processes. It runs on a main receiving thread which will spawn multiple child threads. For the Main thread, its primary job is almost similar to the receive function in the Client application. It will poll for arriving request bundles. On receiving the request bundle, it will unbundle the request and store it in the database which will then be read. New Child threads will be spawned for every request received.

The new threads produced by the Main are called Child threads. The job of these Child threads is to send the subsequent request into the Internet. As the response time may differ for different objects in the Internet, multiple threads are required to prevent bottleneck while waiting for replies from the Internet. Sockets are used for the packet injection into Internet.

Once its subsequent response is received, it will store its response into the database together with the Process ID received previously. The extension of this file will be .res and its name is the previous Process ID received. The file will eventually be bundled and sent over the DTN network back to the Client.

Genre	Websites	Results	Remarks
Static Page and Search	**Google** (www.google.com.sg)	✓	AJAX in search bar working and search functionality works
	Yahoo (www.yahoo.com)	✓	AJAX in search bar working and search functionality works
	Bing (www.bing.com)	✓	AJAX in search bar working and search functionality works
	Wikipedia (www.wikipedia.org)	✓	Search function working
Blogs	**Ieatishootipost** (ieatishootipost.sg)	✓	Photos loaded in reasonable speed
Flash	**Square Enix** (www.square-enix.com/na/)	✓	Flash working
	Miniclips www.miniclips.com	✓	Flash games working
Video	**Youtube** (www.youtube.com)	✓	Video playing but no visually sequential streaming
News	**Asiaone** (www.asiaone.com)	✓	Multiple formats in website working but alignment different in Chrome and Opera
	Soccernet (soccernet.espn.go.com)	✓	Multiple objects working

Fig. 5. Test results of web pages

IV. Results and Discussion

Google reported that on average about 44 resources are needed for each web page [8]. Average network size transferred for each web page is 320KB. Images contribute most to the network size. Different Web pages contains different objects and references. The format and sizes of the objects will differ too. Hence the testing of the application involve web pages that contains different component and of different sizes. Web pages consisting AJAX, Flash, JavaScript and web browser common functions are tested to evaluate the functionality of the application as well as its capabilities to accommodate delays and intermittent connectivity. Three popular web browsers, Mozilla Firefox, Google Chrome and Opera browser are tested. Figs. 4 and 5 show the test results. All three web browsers work with the application and their main functions are all supported by the application. The application also enables surfing of most HTTP web pages with AJAX and Flash objects fully functioning and search can be performed. Under stress test with intermittent connectivity, the application is able to tolerate long delays (tested up to 5 days) by storing the messages in its database before sending it out.

V. Related Work

A project that is related to our proposed project is the DTN-enabled Web Server [7] developed by Lauri Peltola of Helsinki University of Technology. DTN-enabled web server is a server that can receive DTN packets, also known as bundles, containing HTTP requests and then returns the response bundles using MHTML. A separate proxy is created which allows client to access the DTN web servers using DTN bundles. The main difference between this project and our proposed project is that this project assumes that web servers are in DTN network instead of the Internet. Hence the entire web site, including all its resources, is downloaded into the DTN web servers in order for the client to receive the responses. For example, when a client requests for google.com, it must assume that there is an available DTN web server, google.dtn, which has the entire website resources. There is no interaction between the web and the DTN network in general since bundles are only sent within the DTN network and no packet is sent to the Internet. This will limit the clients to visiting static pages with a limited number of websites (only those downloaded) and no direct interaction with the Internet, for example querying a word using Google.

VI. Conclusion

This paper proposes a DTN application to allow users without direct access to the Internet to browse the web via Delay Tolerant Networks. The DTN is essentially overlay over the IP networks and when IP connectivity is lost, web access is still enabled via this DTN networks. Test results show that the proposed DTN application work for all the main web browsers and is tolerant of long delays and can survive intermittent loss in network connectivity unlike IP based web browsing.

REFERENCES

[1] V. Cerf, A. Hooke, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-Tolerant Networking Architecture," RFC 4838, Apr. 2007.

[2] K. Fall, "A delay-tolerant network architecture for challenged internets," in SIGCOMM '03: *Proceedings of the 2003 conference on Applications, Technologies, Architectures and Protocols for Computer Communications*, New York, NY, USA: ACM, 2003, pp. 27–34.

[3] K. Fall and S. Farrell, "DTN: an architectural retrospective," *Selected Areas in Communications*, IEEE Journal on, vol. 26, no. 5, pp. 828–836, June 2008.

[4] K. Scott and S. Burleigh, "Bundle Protocol Specification," RFC 5050 (Experimental), Nov. 2007.

[5] M. Demmer, E. Brewer, K. Fall, S. Jain, M. Ho, and R. Patra, "Implementing Delay Tolerant Networking," IRB-TR-04-020, 28 December, 2004.

[6] M. Demmer, "Architecture of DTN2," Delay Tolerant Network Research Group. [Online].Available: http://dtn.sourceforge.net/DTN2/doc/manual/arch.html#DTN2%20api

[7] Lauri Peltola, "DTN-enabled Web Server," Helsinki University of Technology, May. 2008. [Online]. Available: http://www.netlab.tkk.fi/tutkimus/dtn/web/index.html

[8] Sreeram Ramachandran, "Web metrics: Size and number of resources," Google Code, May. 2010. [Online]. Available: http://code.google.com/speed/articles/web-metrics.html