Seamless deployment of IP applications over delay tolerant network (DTN)

Zoebir Bong

2015

Zoebir Bong. (2015). Seamless deployment of IP applications over delay tolerant network (DTN). Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/65357>

<https://doi.org/10.32657/10356/65357>

**SEAMLESS DEPLOYMENT OF
IP APPLICATIONS OVER
DELAY TOLERANT NETWORK (DTN)**

ZOEBIR BONG

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Master of Engineering

2015

ACKNOWLEDGEMENT

I would like to show my gratitude to my supervisor, Associate Professor Yeo Chai Kiat, for her guidance and encouragement during my Master of Engineering candidature.

I would also like to thank Xia Yang for his technical advice and knowledge sharing.

In addition, a great thank you to Ms Chua Poo Hua, senior laboratory technician of Centre for Multimedia and Network Technology (CeMNet), for her technical and equipment support.

Last but not least, I would like to express my ultimate thankfulness to my family, especially my lovely wife for her understanding, tolerance and continuous moral support to complete my post-graduate study.

ZOEBIR BONG

Nanyang Technological University

School of Computer Engineering

Academic Year 2014-2015

ABSTRACT

Delay tolerant network (DTN) is a network architecture that is capable of transporting data in a challenged environment or one with intermittent connectivity. Its store-and-forward functionality allows easy deployment of ad-hoc or distributed network infrastructure during disaster or disrupted situations, and maintains data delivery assurance over its persistent storage among DTN nodes.

Nowadays, DTN deployment using popular wireless technology (such as Bluetooth, Wi-Fi and so on) could only cover small regions of ad-hoc network owing to the short transmission range of the technology. Although the opportunistic contact of DTN is able to propagate among small ad-hoc networks to cover a wider region, vast areas such as the sea and forests still challenge DTN and hinder further propagation. The old long-range transmission (like amateur radio using UHF radio band) would extend current DTN deployment to a wider space using DTN serial convergence layer. The original serial convergence layer of DTN reference implementation was enhanced with significant improvement on multiple access control and on-the-fly neighbor discovery.

A complete replacement of existing IP networks with DTN does not bring tremendous advantages owing to global network equipment costs and DTN protocol overhead. Instead, the use of DTN as a network overlay or complement is more feasible and appropriate. Proxy agent for DTN is proposed. Initially, a redirection of IP traffic over user-aware DTN proxy was developed. The implementation of user-aware DTN proxy was tested on web browsing, which contributes significantly to the network traffic on the Internet. The user-aware DTN proxy restricts DTN overlay in terms of configurable IP applications such as web browser and email client.

DTN proxy was further developed into a seamless DTN proxy which intercepts traffic from the well-known application port and transports the traffic

over DTN. The seamless DTN proxy is totally transparent to user and requires no modification or configuration on the application layer. The underlying proxy handles the lower layer issue (such as TCP end-to-end connectivity and timeout).

Most IP applications are transported over TCP and/or UDP. To mitigate the development of different DTN proxies to support different IP applications, DTN proxy is further evolved into generic seamless type, DTN GSProxy. DTN GSProxy has TCP and UDP engines (handlers) to support one or both transport layers simultaneously. For instance, HTTP surfing (TCP) needs to resolve the IP address from the given URL name prior to the DNS protocol (UDP).

TABLE OF CONTENTS

ACKNOWLEDGEMENT	II
ABSTRACT	III
TABLE OF CONTENTS	V
LIST OF FIGURES	VIII
LIST OF TABLES	IX
LIST OF ACRONYMS	X
1 INTRODUCTION	1
1.1 Delay Tolerant Network (DTN)	1
1.2 Motivation of Thesis	3
1.3 Contribution of Thesis	4
1.4 Organization of Thesis	5
2 LITERATURE REVIEW	6
2.1 DTN-enabled web server	6
2.2 DTN mail proxy	6
2.3 DTN tunneling	7
2.4 TCP splitting	7
3 ENHANCED SERIAL CONVERGENCE LAYER (ESCL)	9
3.1 Overview	9
3.2 Features	11
3.2.1 Neighbour discovery	11
3.2.2 Media access control	11
3.3 System design	13
3.3.1 Packet structure	13

3.3.2	Roles and modes	14
3.3.3	Operation	15
3.4	Evaluation	15
3.4.1	Scenario 1: Payload transfer time	16
3.4.2	Scenario 2: Multiple access	19
3.4.3	Scenario 3: Disruption tolerance	20
3.5	Discussion	21
3.6	Summary	23
4	DTN PROXY FOR HTTP	24
4.1	Overview	24
4.2	System architecture	25
4.2.1	Bundles	25
4.2.2	ProxyClient	25
4.2.3	ProxyServer	26
4.2.4	Operation	26
4.3	Evaluation	28
4.4	Summary	29
5	SEAMLESS DTN PROXY FOR EMAIL SERVICE	30
5.1	Overview	30
5.2	System architecture	31
5.2.1	ProxyClient	32
5.2.2	ProxyServer	32
5.2.3	SMTP operation	32
5.2.4	POP3 operation	32
5.3	Evaluation and discussion	33
5.3.1	Scenario 1: Steady network connectivity	34
5.3.2	Scenario 2: Intermittent network connectivity	34
5.3.3	Scenario 3: Long haul network interruption	35
5.4	Summary	36
6	GENERIC SEAMLESS DTN PROXY	37
6.1	Overview	37
6.2	System Architecture	38
6.2.1	Policy Manager and Rule	39
6.2.2	Netfilter and Iptables	39

6.2.3	DTN API	41
6.2.4	TCP handler	42
6.2.5	UDP handler	45
6.3	Discussion	47
6.3.1	DTN GSProxy agent against dedicated seamless DTN proxy agent	47
6.3.2	Cross-boundary DTN network with priority tag and DTN ESCL	48
6.4	Summary	49
7	CONCLUSION	50
8	FUTURE WORKS	51
8.1	Potential application layer misbehavior	51
8.2	Secured protocol	51
9	AUTHOR'S PUBLICATIONS	53
10	REFERENCES	54

LIST OF FIGURES

Figure 1: Layer comparison between OSI and DTN.....	2
Figure 2: MACAW handshaking mechanism for control and data packets.	12
Figure 3: ESCL packet structures	13
Figure 4: ESCL's MACAW roles and modes.....	14
Figure 5: Test scenario for data transfer time.....	16
Figure 6: Theoretical and experimental results.....	19
Figure 7: Test scenario for multiple access control	20
Figure 8: Test scenario for disruption tolerance	21
Figure 9: Web traffic flow through DTN proxy for HTTP	26
Figure 10: Mobile email access in vehicular network	30
Figure 11: Proposed seamless DTN proxy for SMTP	31
Figure 12: DTN SMTP proxy evaluation testbed topology	33
Figure 13 : SMTP survival under steady network connectivity.....	34
Figure 14: SMTP survival under intermittent network connectivity	35
Figure 15: SMTP survival under long network disruption.....	36
Figure 16: DTN GSProxy ecosystem design.....	38
Figure 17: Packet flow in netfilter.....	40
Figure 18: Bundling events for a round-trip packet.....	41
Figure 19: TCP packet flow over DTN network	43
Figure 20: UDP packet flow over DTN network	45
Figure 21: DTN proxy agent comparison.....	47
Figure 22: Cross-boundary DTN network with different priority tag	49

LIST OF TABLES

Table 1: MACAW comparison.....	13
Table 2: Payload transfer time with different segment size	18
Table 3: Payload transfer time with different number of ACTIVE nodes	20
Table 4: Payload transfer time with and without disruption	21
Table 5: SCL and ESCL comparison.....	22
Table 6: Web page functionality test	28
Table 7: Bundle priority and delivery pattern	48

LIST OF ACRONYMS

DNS	Domain Name System
AJAX	Asynchronous JavaScript And XML
DTN	Delay Tolerant Network
EID	Endpoint Identification
ESCL	Enhanced Serial Convergence Layer
HTTP	Hyper-Text Transport Protocol
IP	Internet Protocol
LTE	Long Term Evolution
MACAW	Multiple Access with Collision Avoidance for Wireless
PEP	Performance Enhancing Proxy
POP3	Post Office Protocol version 3
SCL	Serial Convergence Layer
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UHF	Ultra High Frequency
VHF	Very High Frequency

1 INTRODUCTION

Quality of service (QoS) for data delivery becomes crucial in present day's data communication. Increasing demand for data communication or interaction among human beings causes high network bandwidth utilization and/or oversubscriptions. Oversubscriptions of network bandwidth either over long range cellular network or short distance Wi-Fi network indirectly degrades transmission due to signal interferences or inability for user to join the network. This phenomenon greatly affects the QoS for data delivery for mobile users especially during network channel handover. Thus, when this phenomenon happens, most users would start to experience disruption in the data transfer. User might need to restart data transmission from the very beginning such as HD quality video transfer over WhatApps (popular chatting mobile application), gigabytes of file attachment over email and so on.

1.1 Delay Tolerant Network (DTN)

Delay Tolerant Network (DTN) was developed by DARPA almost a decade ago and has been widely used in military and scientific industries. Store-and-forward and bundling protocol are main attractive features in getting its popularities in these area but not in commercial and residential aspects. DTN is still unpopular or not commercialized in the mass market because of the high DTN overhead for small data packet transfer and most applications mainly support Internet Protocol (IP) based traffic.

In recent years, Delay Tolerant Network (DTN) architecture [1] [2] [3] has been designed to solve the opportunistic contact communication problem. DTN network is widely deployed in military and interplanetary fields due to its capabilities, namely store-and-forward, convergence layer and bundle protocol [4]. DTN uses store-and-forward approach to resolve data delivery from source to destination. As the name implies, DTN stores DTN packets at a DTN node when the link to the next DTN node is down, and forwards it when there is a (restored) link to the next DTN node. Intermediate DTN node should have a large storage for relaying DTN packet.

DTN is an overlay architecture above existing protocol stack. The DTN layer shown in Figure 1 shows DTN architecture and its comparison to the popular reference design, OSI 7 layer. DTN application layer is the top layer interacting directly with end-

user. DTN reference implementation provides ranges of DTN API for developer to design and implement DTN-based application, also known as DTN-enabled application. IP application is not possible to directly interface with DTN application. However, it can be done using DTN tunnel application developed by DTN research group community. Next layer after DTN application layer is DTN bundle layer. This layer is in charge of encapsulating (de-capsulating) of DTN Application Data Unit (ADU) into (from) DTN bundle packet. The routing decision and protocol is managed by DTN routing layer. Common used for DTN routing protocols are static routing and flooding (epidemic). This layer is where many researchers explore to study, design and implement their own routing algorithm so as to optimize routing decision on particular situation and to maintain data availability, redundancy and delivery. The next interesting DTN layer is convergence layer. This layer is equivalent to combination of transport layer, network layer and data link layer of OSI architecture. It provides the outer encapsulation of DTN transport after bundle encapsulation depending on selected convergence layer type. The current supported convergence layers of DTN reference implementation are File Convergence Layer, Bluetooth Convergence Layer, Serial Convergence Layer, Ethernet Convergence Layer, TCP Convergence Layer and UDP Convergence Layer.

Figure 1: Layer comparison between OSI and DTN

1.2 Motivation of Thesis

In DTN 2.6 reference implementation [5], there are several convergence layers available to support existing communication technology such as Ethernet, Bluetooth, Serial (RS-232) interface [6], UDP, TCP and File convergence layers. DTN convergence layer can be viewed as DTN transport layer to carry DTN application data. DTN link does not distinguish transmission type (wired or wireless) and range (short, medium or long distance). In current market trends and technology, wired connection has higher throughput and more complex transport protocol (mainly over UDP/IP and TCP/IP). However, wired connection restricts the flexibility and freedom to interconnect among nodes. The fastest trending wireless communication like 4G LTE cellular can only support optimum cell range up to 30km. An amateur radio communication can support more than 30km transmission range but at much lower speed. The equipment supporting this communication type still uses serial RS-232 interface. Serial convergence layer in DTN reference implementation supports this interface but without proper multiple access control and neighbor discovery. It was developed and implemented for simple end-to-end serial RS-232 communication. Hence, the exploitation of the enhancement to serial convergence layer is motivated to explore.

Bundle protocol, encapsulates (decapsulates) DTN application data into (from) DTN bundle packet carrying information about the source, the destination and other DTN parameters for routing and replication purposes. This encapsulation process is also called bundling. DTN reference implementation only allows DTN application data (instead of IP data) over DTN network. The seamless delivery of IP application over DTN is another possible area to exploit for future enhancements in DTN.

1.3 Contribution of Thesis

In this thesis, the author implements and proposed some new conceptual enhancements to existing DTN implementation to provide network heterogeneity and versatility of DTN. The main contributions of the thesis are as follows;

- Enhancement from existing DTN serial convergence layer with neighbour discovery and multiple access has been proposed and implemented. The neighbour discovery allows a DTN node to beacon itself to other possible adjacent DTN node so as to establish link and begin bundle forwarding. Multiple access feature manages the shared channel for data delivery within transmission range.
- A new DTN proxy agent for HTTP has been proposed and implemented to allow web surfing transport over DTN environment.
- A further improvement towards seamless version of DTN proxy agent has proposed. The existence of such DTN proxy agent is not aware by the end user. The implementation of seamless DTN proxy has been performed for SMTP to ensure the availability and delivery of email communication under disruptive environment.
- Final evolution of DTN proxy agent, namely generic seamless DTN proxy agent (DTN GSProxy) has been proposed. The UDP and TCP engine within DTN GSProxy allows seamless delivery of all well-know and customised IP applications to be processed and transported over DTN network. In addition, it can be integrated with enhanced serial convergence layer of DTN to enable smart distribution of DTN bundle according to message urgency over different transmission medium.

1.4 Organization of Thesis

The remaining chapters of the thesis are organized as follows:

- Chapter 2 gathers and reviews existing relevant DTN implementation with its capabilities and shortcomings, and multiple access protocol used in current market.
- Chapter 3 explains channel utilization of conventional channel using amateur radio for very long range but small data packet transfer.
- Chapter 4 proposes a proxy agent to encapsulate IP-based data packet into DTN bundles.
- Chapter 5 reinstates DTN proxy agent to have IP traffic transported over DTN network in a seamless approach.
- Chapter 6 proposes the evolution of DTN proxy agent towards generic and seamless mode that is able to support all types of applications through its internal engines, namely UDP and TCP handlers.
- Chapter 7 concludes the thesis.
- Chapter 8 discusses potential works that might be extended from this thesis.

2 LITERATURE REVIEW

2.1 DTN-enabled web server

A similar project related to the proposed DTN proxy for HTTP is “DTN-enabled Web Server” [7] developed by Lauri Peltola of Helsinki University of Technology. The project is a server-client based DTN proxy. The client obtains HTTP requests from web browser and encapsulates the messages into DTN bundles. Then, it forwards the DTN bundles to the server. The server receives DTN bundles (HTTP requests) from and returns DTN bundles (HTTP responses with MHTML) to the client. However, the project assumes that IP web servers are in the DTN network and has all the web contents ready. For instance, if a web client would like to browse Google website, the Google web server (DTN endpoint ID of google.dtn) has to be in the DTN network. Web client could not perform a search in the requested Google website, as it may not contain the search keyword resources available in its web server.

Therefore, the project allows web client to browse only the static and limited web contents of web servers available in the DTN network. Dynamic browsing and multimedia streaming are not supported by the project. The motivation for the author is to allow all fundamental web browsing to address all the problems in the said project.

2.2 DTN mail proxy

Refer to [8] for a comprehensive survey on recent development on DTN, DTN is not fully compatible with IP applications and this makes it difficult for IP applications to benefit from DTN. Some research efforts have been made to enable IP-based applications over DTN, for instance, email access [9], web search [10], and other web applications like Facebook [11] and Twitter [12].

Email access over DTN is the main focus. In [9] Tuomo et al proposed to use DTN Mail Proxy to interface IP mail applications with DTN. It provided standard Simple Mail Transfer Protocol (SMTP) [13] service and Post Office Protocol Version 3 (POP3) [14] service as interfaces for email application to send and receive mails via this proxy. This approach has several limitations. Firstly, it requires the installation of DTN Mail Proxy on user’s device. Secondly, email client application must be

configured to use the mail proxy as the mail server in order to use DTN mail service and switch back to the original configuration to use IP-based mail service. In addition, consider a scenario where multiple users attempt to access mail service in the same vehicle, manual mail client configuration is less efficient. As the number of users increases, the configuration is much more complex. This approach is quite similar to the author's DTN proxy in chapter 3 in which the user is aware of such a proxy to transport data over DTN.

2.3 DTN tunneling

DTN 2.6.0 reference implementation features tunneling of UDP and TCP over DTN network. UDP and TCP packets are successfully captured and tunneled over DTN network with the aid of iptables rules. The approach involves configuration of iptables rules. Application port used by source node must be known in advance so as to run a dedicated DTN tunnel command for it. In other words, this approach needs awareness of user to redirect or modify existing applications to the listening port. DTN tunnel featured in the reference implementation is unidirectional tunneling. It requires two tunneling sessions to complete a single application.

2.4 TCP splitting

TCP is a popular transport protocol used for reliable data delivery. It is normally overlaid over IP network layer and commonly known as TCP/IP. It is an end-to-end transport service to ensure data is delivered in ordered and error-checked state. For long-haul TCP transmission, a single circuit breakdown may trigger connection re-establishment. In [15] [16] [17], performance enhancing proxy (PEP) approach is used to split TCP connection in two independent TCP connections. PEP was originally developed and implemented in satellite communication. TCP splitting is one of PEP features typically used to solve TCP problems with large round trip time (RTT). PEP approach might be utilized to curb the problem faced by DTN Mail Proxy. DTN proxy transparency is crucial, which requires no installation and configuration to user applications and existing IP network infrastructure. The proxy awareness problem might be resolved using PEP approach. Random link degradation over satellite environment hinders a steady TCP-based communication between two base stations.

PEP introduces the idea of splitting TCP connection from end-to-end to hop-by-hop. This is achieved by using the orbiting satellite acting as a TCP splitter, which replies with TCP SYNC to the source base station instead of the destination base station. In other words, this PEP approach splits the original single TCP session (between base stations) into two independent TCP sessions (between satellite and base station). The author proposes a seamless DTN proxy based on PEP method to make email proxy transparent.

3 ENHANCED SERIAL CONVERGENCE LAYER (ESCL)

3.1 Overview

In 1982, a worldwide network was formed and known as the Internet, interconnecting TCP/IP based networks in different domains and countries. Today, the TCP/IP still remains as a popular protocol of the Internet, even though other protocols slowly catch up. As more and more people in this world join and contribute to Internet, networking medium and equipment evolved as well. Networking equipment like routers, switches, access points and others are developed to provide end-user connectivity. Networking medium evolves from cable to optical fibre to improve the control and management of exponentially increasing of network traffic. Higher quality (such as 4K or Ultra High Definition) video streaming becomes common and trending. Smoother high quality video traffic can be accomplished with a steady and wider network bandwidth of infrastructure. A higher speed networking also enables delay-sensitive communication such as remote medical surgery which requires near-zero latency to perform the operation.

However, as high speed network infrastructure is merely wired networking laid underground or sea, they are prone to damages caused by natural disaster like earthquake. When earthquake occurs and affects this infrastructure, communication across region or even globally is not possible. Alongside high speed wired network infrastructure, wireless networking (LTE, Bluetooth and Wi-Fi) is an easy complement to wired infrastructure. Wireless network incurs a lower cost and is faster to deploy but has a lower data rate compared to the wired networking. In the absence of wired networking, wireless networking offers alternate configuration as ad-hoc communication. However, communication disruption across islands or mountains would still be a missing connectivity gap. This cross boundary communication can be covered with the aid of conventional range communication protocol such as amateur radio.

Amateur radio communication uses VHF or UHF as its transmission bandwidth to reach destinations more than 50km away and is only feasible to achieve long distance communication if the environmental conditions are met such as no channel interference, line-of-sight obstacles and opportunistic contact. These environmental factors are hard to predict and overcome. Hence, people abandoned amateur radio in favour for other wireless communication approach. Owing to its narrow bandwidth, channel contention is resolved with half-duplex communication approach but lacks a better mechanism in accessing the shared media. Furthermore, amateur radio communication is considered as live data streaming over shared channel. No data is stored or retransmitted if contention occurred.

To ensure data recovery and delivery, the store and forward capabilities of DTN allows storing data in local disk on a DTN node. If there is contention or environmental obstruction, data still remains intact in local disk and is kept until successful delivery of data to next DTN hop. Thus, DTN is an appropriate resolution to alleviate amateur radio problem due to contention and other environmental factors with DTN's store-and-forward capability to relay DTN bundles over linkable DTN nodes to reach its destination. DTN also provides Convergence Layer to interconnect heterogeneous network segments. As common interface for amateur radio equipment is serial RS-232, DTN 2.6 reference code provides convergence layer to transport messages with serial RS-232. The convergence layer is named as Serial Convergence Layer and is denoted in this thesis as SCL. SCL implementation is confined to the following:

- Only support simple RS-232 between two DTN nodes through null modem cable.
- Pre-defined destination user identification. No neighbour discovery is incorporated.
- No control protocol for multiple access over shared media channel.

Although users within the communication range make do with data transfer via automated or manual scheduling, the entry of an unknown node into the area could cause data corruption and loss which results in a high error rate. A proper and managed communication among all participating DTN nodes within the communication range is required. Thus, SCL fails to satisfy such requirements. Hence, the author is motivated to design an Enhanced Serial Convergence Layer (ESCL).

3.2 Features

ESCL is proposed and designed to address the drawbacks of SCL. Original SCL is only restricted to use one-to-one direct communication over a dedicated channel (null modem). In contrast to SCL, ESCL is designed to operate in wireless communication which more DTN nodes exist. It is impossible for a local DTN node to know all other DTN node in advance. The identification of each adjacent DTN node is required to build DTN routing table for bundle forwarding. Advertisement of a DTN node must be fired to neighboring nodes.

Besides, for a reliable communication over a free wireless space, a distributed access control must be implemented to prevent channel contention over shared media access and non-DTN signal interruption. High contention over shared channel brings impact lower data delivery rate and possible channel jamming. Detailed on which media access control protocol is implemented in ESCL will be discussed further in the subsection.

Therefore, the architecture of ESCL is based on the existing SCL with modifications to incorporate the essential features, namely neighbour discovery and media access control.

3.2.1 Neighbour discovery

This feature is in charge of periodic advertisement of local DTN endpoint identification (eid) within the transmission range so as to inform any DTN peer nodes about self-existence in the shared channel media. It also registers and tabulates the received neighbour discovery advertisements sent by neighbouring DTN peer nodes.

3.2.2 Media access control

There is a number of media access control protocols available in the market. Multiple Accesses with Collision Avoidance for Wireless (MACAW) [16] is chosen for ESCL's media access control due to its simple control mechanism and small overhead requirement for narrow bandwidth of amateur radio channel. MACAW is also widely used to avoid collision and long-haul large data transfer in IEEE 802.11 standard. MACAW handshaking mechanism uses frame sequence shown in **Figure 2** for transferring data from the source (Node A) to the destination (Node B) and solves collision problem with binary exponential back-off approach. A complete handshaking

mechanism (without retransmission/collision) involves the following frame/packet sequences.

- When Node A (source) has data for node B (destination), it requests its permission to node B using RTS (Request-To-Send) control packet.
- Node B replies with CTS (Clear-To-Send) control packet back to node A indicating its readiness for receiving data.
- Node A sends DS (Data-Sending) control packet to node B containing information about payload size to receive following this control packet.
- Node A then proceeds to send actual DATA (payload) packet to node B.
- When node B finishes receiving payload, it replies with ACK (Acknowledgement) control packet to node A indicating its successful data transfer.

Figure 2: MACAW handshaking mechanism for control and data packets.

Binary exponential back-off refers to algorithm to wait for random time slot before attempts to retransmit data. The random time slot is chosen from a range between 0 and back-off interval ($F(x)$) where x is number of collision. Back-off interval changes according to number of collisions and whether collision occurs. Upon a collision, $F(x) = \text{MIN}(1.5x, \text{BOmax})$ and upon successful transmission, $F(x) = \text{MAX}(x-1, \text{BOmin})$. BOmax and BOmin refer to maximum and minimum back-off factor respectively. Slight modifications on MACAW are made to adapt for DTN ESCL and amateur radio environment. The difference is shown in Table 1.

Table 1: MACAW comparison

	Conventional MACAW	DTN ESCL MACAW
Node identity	No	Yes. Using DTN eid for source and destination ID.
BO_{max}	Any value	2
BO_{max}	Any value	32
F(x) upon collision	MIN (1.5x, BO _{max})	MIN (1.3x, BO _{max})
F(x) upon success	MAX (x-1, BO _{min})	MAX (x-1, BO _{min})

3.3 System design

3.3.1 Packet structure

Basically, MACAW handshake mechanism involves control and data packets communicating among the nodes over shared radio channel. ESCL defines two-field and three-field packet structures for control and data packet respectively as shown in **Figure 3**.

Figure 3: ESCL packet structures

- 1) **MAGIC** field is a special word-size field in the packet. It is used to distinguish ESCL packet resemblance from unknown packet as there is no control protocol in the shared radio channel exists. It is also be used to mark the beginning of packet sequence.
- 2) **HEADER** field; is defined differently in control packet and data packet. For control packet, HEADER field contains MACAW control type (RTS, CTS, DS or ACK) and neighbour discovery flag. Neighbour discovery flag is only used for none other than RTS control packet. In other words, RTS control packet serves as normal MACAW control packet or neighbour discovery packet. For data packet, HEADER field describes DATA payload length to follow after HEADER field.
- 3) **DATA** field; contains application layer data unit.

3.3.2 Roles and modes

ESCL defines two roles (ACTIVE as red shaded node, PASSIVE as white clear node) and three modes (SENDER as blue node, RECEIVER as green node, LISTENER as grey node) for a DTN node as in **Figure 4**. A DTN node is only considered as ACTIVE node in **Figure 4(a)** if the node has data to transmit or relay and is given an opportunity to compete with other nearby ACTIVE nodes in order to secure shared radio channel. Other DTN nodes with no data on hand are PASSIVE nodes. To avoid collision, each ACTIVE node waits for a random duration (named as Back-off Algorithm) before performing the contention mechanism. Back-off algorithm and contention mechanism repeat until a winning node is found or upon a successful data transfer. The winning ACTIVE node B, illustrated in **Figure 4(b)**, from the contention mechanism deserves the right to enter SENDER mode and is allowed to transfer data packets (including MACAW handshaking control packets) to the destination node, which enters RECEIVER mode. Other PASSIVE and losing ACTIVE nodes enter LISTENER mode and only return to its original role and mode once active data transmission completes. LISTENER mode defers any received data and control packets.

Figure 4: ESCL's MACAW roles and modes.

3.3.3 Operation

ESCL transports DTN bundles from DTN application and processes them before sending raw data over amateur radio channel via serial RS-232 radio modem. The implicit process involves bundle segmentation, packet queue and encapsulation. Prior to actual data transmission, a sending node must compete and acquire the shared radio channel. The winning node holds the channel until completion or cancellation of data transfer. A completion of data transfer refers to a successful data transfer and the receipt of acknowledgement. Upon channel acquisition, the actual data will be transferred to any neighbouring nodes if flood-routing method is used. The algorithm was based on MACAW channel acquisition method.

Besides, destination address must be known and retrieved from ID table, which is obtained through periodic neighbour discovery advertisement. As explained in the earlier section, each ESCL node advertises itself to shared radio channel and tabulates the received advertisement from peer ESCL nodes.

3.4 Evaluation

Three test scenarios were set up to evaluate DTN ESCL. First scenario evaluates the payload transfer time between two DTN nodes under non-disruptive environment in comparing the original DTN SCL and the proposed DTN ESCL. The scenario also includes theoretical computation and experimental measurement of payload transfer time. Second scenario showed how DTN ESCL node is able to work with adjacent node to handle potential collision under shared channel and non-disruptive environment. The collision handling is managed in distributed manner with multiple access and neighbor discovery components of DTN ESCL. Third scenario introduces disruption component to the environment to reflect the disruption tolerance of DTN ESCL.

Each DTN node in the evaluation test bed is a laptop equipped with a SATELLINE 3AS UHF radio modem through serial RS-232 interface. The radio modem is configured as follows:

- 8 data bit, 1 start bit, 1 stop bit and 1 even-parity bit
- 38400bps bitrate
- 10mW transmit power
- -80dBm receiver sensitivity

- 458.725MHz UHF radio channel
- 10kB payload
- 30 seconds discovery advertisement

3.4.1 Scenario 1: Payload transfer time

This test scenario in **Figure 5** illustrates a direct data transfer of 10kB payload from node A to node B with no other ACTIVE nodes. The objective is to measure payload transfer time under different MTU sizes and evaluate the comparative differences. The initial plan for setting this test scenario was to benchmark the performance of ESCL to SCL. However, wireless performance of SCL was not included because of inconsistent and random measurement results caused by data collision and interleaving over the air. Therefore, both theoretical computation and experimental measurements of payload transfer time were performed.

- Case 1a: Theoretical computation of DTN SCL
- Case 1b: Theoretical computation of DTN ESCL
- Case 1c: Experimental measurement of DTN SCL in a wired setup
- Case 1d: Experimental measurement of DTN ESCL in a wired setup
- Case 1e: Experimental measurement of DTN ESCL in a wireless setup

Figure 5: Test scenario for data transfer time

Each message comprises single data byte (8-bit) and 3 stuff bits (namely star bit, stop bit and parity bit). The required time to effectively transmit a message is defined as T_{byte} . The equation for calculating T_{byte} is as follows.

$$T_{byte} = \frac{8 + stuffbit}{bitrate}$$

With radio modem's bit rate of 38400bps, T_{byte} for a single byte message is approximately 286.46 microseconds.

Payload transfer time, $T_{payload}$, represents the time required to transmit the data/payload bundle to its destination node. DTN SCL was designed to provide direct serial communication and hence, no overhead was introduced. $T_{payload}$ theoretical computation for DTN SCL is pretty straight forward as follows:

$$T_{payload(SCL)} = data * T_{byte}$$

In contrast to DTN SCL, the theoretical computation of DTN ESCL is complicated with additional overhead contributed by MACAW algorithm, neighbor discovery flag and MAGIC field. In section 3.2.2, a successful MACAW handshaking mechanism requires four control packets and one data packet (payload). Control packets are fixed 50 bytes long, while data packet is variable and depends on payload from DTN layer. T_{RTS} , T_{CTS} , T_{DS} , T_{DATA} and T_{ACK} refer to transfer time for RTS, CTS, DS, DATA and ACK packet respectively. Thus, computation for payload transfer time over DTN ESCL is as follows;

$$\begin{aligned} T_{payload(ESCL)} &= T_{RTS} + T_{CTS} + T_{DS} + T_{DATA} + T_{ACK} \\ &= 4 * 50 * T_{byte} + data * T_{byte} \\ &= (200 + data) * T_{byte} \end{aligned}$$

With introduction of payload segmentation in DTN convergence layer, payload transfer time for DTN ESCL is thus dependent on two parameters, namely payload segment count ($N_{segment}$) and segment size. Each segment count contains RTS, CTS, DS, segmented DATA and ACK. The smaller segment size is used, the more payload segment is required to deliver to destination. In other words, more control packets are used in order to transfer a full payload. Smaller segment size means shorter $T_{payload}$ and fairer to other ACTIVE nodes in contention but less data efficiency due to increased MACAW overhead. Theoretical computation for DTN ESCL is then given as:

$$\begin{aligned} N_{segment} &= ceiling \left\{ \frac{data}{segment\ size} \right\} \\ T_{segment(ESCL)} &= (200 + segment) * T_{byte} \\ T_{payload(ESCL)} &= N_{segment} * T_{segment(ESCL)} \\ &\text{or} \\ T_{payload(ESCL)} &= (200 * N_{segment} + data) * T_{byte} \end{aligned}$$

In this test scenario, segment size is varied to evaluate and compare the performance of cases listed earlier. Theoretical computation and experimental results of data transfer from node A to node B are tabulated in **Table 2**. Note that experimental results for segment size more than 4000 bytes under cases 1c, 1d and 1e cannot be measured due to hardware limitation. The unit under test has a maximum buffer of 4000 bytes for RS-232 transceiver. In **Figure 6**, it was expected to have same constant payload transfer time for cases 1a and 1c regardless of segment size variation since no overhead is incurred in DTN SCL. It also implies 100% baud rate utilization. Another observation from **Table 2** is a smaller segment size for DTN ESCL cases tends to take a longer time to complete data transfer. This is because when segment size is small, the number of data frame increases for the same payload. This also results in more MACAW overhead to support multiple access and becomes much more significant compared to segment size.

Note that the theoretical computation (case 1b) and experimental measurement (case 1d) of payload transfer time for DTN ESCL are different with the latter being higher. This is due to the inclusion of the sender's random delay as well as the read buffer scanning delay. In comparing case 1d results to case 1e results, payload transfer time is higher in wireless setup. This is expected since there is additional delay caused by the radio modem as well as the higher probability of wireless interferences.

Table 2: Payload transfer time with different segment size

Segment size (bytes)	N_frame	T_payload(s)				
		1a	1b	1c	1d	1e
500	20	2.86	4.01	2.90	9.10	13.20
1000	10	2.86	3.44	2.90	6.60	9.10
1500	7	2.86	3.27	2.90	5.40	8.20
2000	5	2.86	3.15	2.90	4.90	7.40
2500	4	2.86	3.09	2.90	4.80	6.90
3000	4	2.86	3.09	2.90	4.80	6.60
3500	3	2.86	3.04	2.90	4.70	6.00
4000	3	2.86	3.04	-	-	-
4500	3	2.86	3.04	-	-	-
5000	2	2.86	2.98	-	-	-

Figure 6: Theoretical and experimental results

3.4.2 Scenario 2: Multiple access

The test scenario shown in **Figure 7** illustrates both DTN SCL and DTN ESCL performances under multiple ACTIVE nodes in the same transmission region. Simultaneous data transfer from node A to node B, from node B to node C and from node C to node A are triggered to simulate a more practical multiple access event. The purpose of the scenario is to measure the capability of DTN ESCL to support multiple access over shared radio channel. In this evaluation, performance measurement for DTN SCL was not carried out due to the absence of access control mechanism. Segment size of 3500 bytes is used for the data transfer. Two test cases are used in our evaluation. A two-node data transfer in case 2a and multiple data transfer scenario is involved in case 2b.

Figure 7: Test scenario for multiple access control

Table 3: Payload transfer time with different number of ACTIVE nodes

Case	Scenario	T_payload (s)
2a	2-nodes; A->B	6.00
2b	3-nodes; A->B, B->C, C->A	7.82

From the results obtained in **Table 3**, a longer payload transfer time in case 2b is expected than case 2a. This is because additional ACTIVE node within the same transmission region poses a greater challenge among the nodes in competing for the shared media channel. Thus, it increases the probability of channel collisions which indirectly reflects longer payload transfer time.

3.4.3 Scenario 3: Disruption tolerance

The scenario in **Figure 8** demonstrates hop-by-hop data transfer with the source originating from node A to the destination node C via the relay which is node B. Under case 3a, a multiple hopping payload transfer is expected to take 13 seconds to complete, which equals to two single (per hop) payload transfer under no link disruption. In order to fully feature DTN's store-and-forward capability, a 100-second network disruption is introduced in case 3b. The network disruption includes;

- Link break-down; a manual switch-off of radio modem at node B after first two frames are received.
- Link recovery; a manual switch-on of radio modem at node B after 100 seconds from link break-down event.

Figure 8: Test scenario for disruption tolerance

Table 4: Payload transfer time with and without disruption

Case	Scenario	T_payload (s)
3a	No disruption; A->B->C	13.13
3b	100-sec disruption for B; A->B-> C	117.80

Despite the 100-second delay caused by network instability, the results in **Table 4** shows that payload in case 3b still manages to reach the destination at node C. This proves the disruption tolerance of DTN ensures guaranteed payload delivery in the absence of DTN bundle expiration.

3.5 Discussion

From the evaluation section, there are several factors contributing to the significant difference in ESCL payload transfer time between theoretical computation and experimental results. The following factors are ignored in the theoretical computation:

1. Software/hardware processing time
2. Scanning time of read buffer
3. PC-to-modem delay
4. Free air propagation delay
5. MACAW sender's random delay
6. Collision delay

The main contributing factor is due to buffer scanning time consumption, as each DTN node scans the received payload for control packet's MAGIC keyword so as to prevent interferences among DTN peers within transmission region. In addition, owing to hardware limitation on the maximum overflow buffer of RS-232 transceiver, this prevents the author from performing experimental measurement for segment size more than 3500 bytes. When the received payload exceeds the transceiver's buffer limit, it triggers the transceiver's overflow prevention process by reducing the allowable bytes to be passed from the PC to the modem. Thus, segment size of 3500 bytes is the maximum segment size supported by the proposed ESCL implementation in achieving good stability and reliability.

With the implementation of media access control (MACAW) and neighbour discovery, the proposed ESCL improves channel contention and channel utilization among different users over shared radio channel. Normally, link disruption could cause dropped or loss data after a long delay. With ESCL, although link disruption occurs in the network, it does not stop data from reaching the destination node. Unfortunately, there is a trade-off in using ESCL, which is data rate efficiency. From the experimental results in the evaluation section, ESCL delivers 18.33 kbps¹ out of a configured baud rate of 38.4 kbps in the radio modem. This only yields 47.74%, which is much lower compared to SCL, which is almost 100% data rate efficiency. However, this is understandable and acceptable because media access control and neighbour discovery are implemented in ESCL, which improves data survival in the traditional amateur radio network. A feature comparison between SCL and ESCL is tabulated in Table 5.

Table 5: SCL and ESCL comparison

Features	Original SCL	Proposed ESCL
Neighbour discovery	Manual	Automatic
Media access control	No	Yes
Disruption tolerance	Yes	Yes
Performance	Good ²	Slight lower ³
Overhead	No ²	Yes ³

¹ The data rate computation is based on 10kB payload delivery over 6 seconds. Each byte contains 1 start bit, 1 stop bit, 8 data bits and 1 parity bit.

² Point-to-point transfer

³ Point-to-point transfer and support media access control

3.6 Summary

In comparing SCL and ESCL, the latter has better payload transfer performance and better utilisation of data rate in simple point-to-point traffic but fails to support multiple accesses. ESCL is able to address the drawback with the introduction of MACAW signal messages and neighbour discovery features, thereby incurring communication overheads. Trade-off is thus expected to compensate for the multiple access capability with actual data rate utilisation hovering around 50% of the baud rate settings on the radio modem.

With a higher number of ESCL nodes being involved in the communication, a higher degree of contention fairness must be considered. To achieve a fairer MACAW contention mechanism, two factors can be refined, namely; MACAW back-off equation and segment size. More collisions take place with more ESCL nodes contending for the channel. As explained in MACAW, collision increases the back-off ranges, which indirectly affects the sender's random delay generated and thus minimizes the chance for collision in the next contention window. Smaller MTU size implies more frame transmissions but a shorter channel holding time. The following further improvements on ESCL can be considered:

- Optimum adjustment of back-off equation, delay and deferred timings.
- Error detection and correction to be included in the frame which may replace the read buffer scanning method. This will increase the data transfer efficiency.
- Use of radio modem with a higher baud rate or with multiple radio channel capability.
- Auto-adjustment of segment size in response to environmental congestion.

The development and implementation of DTN ESCL is part of preparations for allowing DTN communication over long distance using amateur radio channel. This alternate communication link can be utilized to prioritize payload with higher level of urgency to reach destination in advance such as DNS IP packet, text message and so on. Next chapter, author focuses on proxy to enable IP delivery over DTN network.

4 DTN PROXY FOR HTTP

4.1 Overview

The Internet is a virtual world or platform connecting the human beings around the world. Every piece of information can be found and updated in a fast pace in every day, hours, minutes and seconds. It has become a daily basic task for many people. People can surf the Internet for information, watching video, posting message in Facebook page, checking email and many more.

Unfortunately, there are always moments that connections to the Internet are disrupted due to network outages, natural disaster, partial infrastructure failure. The Internet has used TCP/IP protocol to run its handshaking and/or data transfer. It is known that TCP/IP requires end-to-end connection to provide reliable data transfer but it suffers disconnection or data loss when a long delay encountered. Under such circumstances, an alternative network allows belated Internet communication, namely Delay Tolerant Network (in short, DTN). DTN plays an important role in providing services under disrupted environment due to the following features;

- To withstand intermittent connectivity.
- To enable opportunistic contact communication.
- Hop-to-hop communication via store-and-forward approach.
- Overlay across heterogeneous network by its Bundle protocol and convergence layer.

Instead of replacing existing infrastructure (TCP/IP) to the Internet, DTN would be used as a complement. HTTP is the fundamental protocol used to serve Internet communication. A DTN proxy for HTTP was developed to allow users in isolated environment to browse the web (Internet). To enable DTN proxy for HTTP, web user configures their preferred web browser to a dedicated HTTP proxy port number, which the DTN proxy listens to.

4.2 System architecture

There are two components of DTN proxy implementation [17] for HTTP, namely ProxyClient and ProxyServer.

4.2.1 Bundles

HTTP is viewed as client-server or request-reply communication protocol with a HTTP request originating from the web browser to the HTTP server and a HTTP response replied by the HTTP server to the web browser. In this thesis, the author uses terms like BundleRequest and BundleResponse to represent DTN bundle containing HTTP request and HTTP response respectively in elaborating the flow of data. Thus, BundleRequest contains HTTP request encapsulated by ProxyClient and BundleResponse carries HTTP response replied by ProxyServer. Along with HTTP messages, task ID is also a HTTP session socket number inserted before the HTTP messages in the DTN bundle. Task ID is how HTTP response is to be forwarded to the web browser via a socket number. Both ProxyClient and ProxyServer use DTN 2.7.0 API [18] to perform bundling and unbundling of DTN bundles. DTN daemon is in charge of the routing decision for both BundleRequest and BundleResponse.

A web user in a DTN node may use different web browser and/or tabs simultaneously. Each tab session may perform multiple rounds of HTTP requests. For simplicity and scalability, each DTN bundle would only contain one HTTP message (either HTTP request or HTTP response). With store-and-forward nature, DTN is able to manage large number of HTTP messages when the hop-by-hop link is down.

4.2.2 ProxyClient

This type of DTN proxy is considered as client-side DTN proxy for HTTP. It resides in a DTN node with intermittent connectivity or has no direct access to the Internet. It is also the intermediate agent bridging the web browser and the DTN daemon. Web users configure their web browsers to redirect all HTTP traffic to its dedicated port listened by ProxyClient. The tasks performed by ProxyClient are:

- Bundles HTTP request captured at its dedicated port configured by web browser into BundleRequest.
- Sends BundleRequest to its pre-defined or known destination, which is a DTN node with a ProxyServer.

- Unbundles BundleResponse into HTTP response.
- Returns HTTP response back to the web browser based on the task ID carried in BundleResponse.

4.2.3 ProxyServer

A DTN node with connectivity to both DTN and conventional IP network is where ProxyServer situates itself. It is known as server-side DTN proxy for HTTP. Normally, ProxyServer resides near the edge of DTN network and performs the following tasks:

- Unbundles the received BundleRequest into HTTP request.
- Creates a worker for each HTTP request in establishing independent HTTP session to actual HTTP server.
- Bundles HTTP response into BundleResponse.
- Replies BundleResponse back to the originator of the web session via DTN transport.

4.2.4 Operation

Web user browses the Internet with HTTP URL. This indirectly triggers a chain of event consisting of HTTP requests and responses. In other words, a simple web browsing request involves a number of HTTP activities on the wire. DTN proxy agents (both ProxyClient and ProxyServer) handle each HTTP activity as a single job. The data flows involved in a single HTTP job by the DTN proxy agent are illustrated in Figure 9.

Figure 9: Web traffic flow through DTN proxy for HTTP

At first, the web browsing URL initiates HTTP REQUEST to be captured by ProxyClient over its listening port. JobManager of ProxyClient creates and queues a new JobWorker with socket fd number as its JobId. The JobWorker encapsulates JobId and HTTP REQUEST into BundleRequest tagged with the destination DTN eid of ProxyServer. In this illustration, one single ProxyServer resides at one isolated DTN environment. Then, JobWorker sends BundleRequest to the DTN environment via DTN daemon. DTN proxy agent has no control over bundle routing over the DTN environment. The routing is totally handled by the routing decision/protocol configured on each DTN node. Delivery time of DTN bundle is not guaranteed and depends on the DTN topology complexity, routing protocol and/or bundle size.

Once BundleRequest reaches ProxyServer, ProxyServer has its JobManager module to create a ServerWorker to handle the received DTN bundle. The ServerWorker extracts HTTP REQUEST from BundleRequest. Then, ServerWorker acts as HTTP client and sends the extracted HTTP REQUEST to the actual requested HTTP server over the Internet.

As ProxyServer has definite and steady connection to the Internet, HTTP RESPONSE is expected to be received regardless of the correctness in HTTP REQUEST message content. HTTP RESPONSE received by ServerWorker is encapsulated together with the original JobId into BundleResponse tagged with the destination DTN eid of the original ProxyClient. Similar to ProxyClient, ProxyServer injects BundleResponse into the DTN environment through the DTN daemon, and awaits for self-termination. Self-termination of ProxyServer by JobManager implies the completion of HTTP REQUEST-RESPONSE activity, the closure of HTTP socket connection from the HTTP server and the release of memory allocation for the thread/instance of ProxyServer.

The moment BundleResponse reaches the original ProxyClient, and is passed to JobManager in order to retrieve the JobId in the BundleResponse. JobManager passes HTTP RESPONSE extracted from BundleResponse to the corresponding queued ClientWorker according to the JobId. The ClientWorker sends the returned HTTP response to the web user via a socket number, which is also the JobId.

4.3 Evaluation

Google reported that on average about 44 resources are needed for each web page [19] with average transferred network size of 320kB. The factors that contribute the most are images and streaming media. As web pages are free for designer to develop, the format and size of web page differs. Hence, the author selects several web pages that contain different types of the most commonly used web components such as AJAX [20], Flash [21], JavaScript [22] and so on to evaluate the functionality of DTN proxy agent as well as its capabilities to accommodate the delay and intermittent connectivity. As DTN proxy agent was developed and implemented on Linux platform, Microsoft's Internet Explorer was not included for the evaluation. Instead, other popular web browsers are used like Google Chrome, Mozilla Firefox and Opera web browsers. These web browsers work well with the DTN proxy. The DTN proxy for HTTP also enables the surfing of HTTP web pages with AJAX and Flash enabled. Under stress test with long haul disruption [23] (tested up to 5 days), the DTN proxy is still able to tolerate such long delay by storing HTTP messages in DTN nodes before sending it back to the web browser. A full functionality test of DTN proxy for HTTP is shown in Table 6.

Table 6: Web page functionality test

Functionality	Test	Remarks
Search bar	Ok	Search engine with AJAX quick responses
File downloading	Ok	Large HTTP file downloading
Multiple tab browsing	Ok	Supports multiple web session handling capability
Intermittent connectivity	Ok	
Long haul disruption	Ok	Able to tolerate long delay
Flash	Ok	
Multimedia	Ok	

4.4 Summary

DTN proxy agent allows web users in the isolated environment or intermittent network connectivity to surf the Internet by redirecting IP traffic over DTN. Test results show that DTN proxy agent is able to work with all popular web browsers and is tolerant of long delays compared to IP based web browsing. In addition, it also highlighted the successful of IP traffic delivery over DTN network. However, pre-configured listening port must be made known to IP application in order to redirect IP packet to DTN server. Next chapter, author searches for seamless approach to address port redirection issue.

5 SEAMLESS DTN PROXY FOR EMAIL SERVICE

5.1 Overview

In recent years, mobile chipset and relevant hardware technology evolve rapidly in terms of higher performance and lower power consumption. This indirectly brings impact on richer content in mobile communication from old time simple text messaging until today's web surfing, full email access and video streaming. This rich content mobile communication could only be achieved with higher data rate between the mobile client and the service provider (Wi-Fi access point or 4G LTE network).

Although mobile network provider like 4G LTE services has wider coverage for mobility and its improved data rate, Wi-Fi access point is still favoured among mobile users. It is because Wi-Fi access point provides much higher data rate (1.69Gbps with Wi-Fi 802.11ac technology), ad-hoc connectivity and lower setup cost. Thus, free Wi-Fi access points are rapidly deployed in public area such shopping mall, transport station, café and so on in order to give mobile users a better data rate, free and steady connectivity. However, there are always scenarios that some groups of mobile users are equipped with Wi-Fi only, namely laptop and tablet users. These groups can only rely on Wi-Fi access points to obtain the Internet contents, which may encounter the short-live or intermittent connectivity in the scenario illustrated in Figure 10.

Figure 10: Mobile email access in vehicular network

In this scenario, mobile user (tablet) can access the Internet by forming ad-hoc network with mobile relay (car), which is connected to the roadside Wi-Fi access point (another relay point). Such multi-hop connection is short-lived due to many reasons such as car movement or distance between the mobile user and the mobile relay. This could cause severe performance degradation to normal IP applications, especially TCP-based application, which requires end-to-end connection. Again, DTN utilisation for such scenario is very appropriate, as its store-and-forward approach allows TCP-based application to be transported in a hop-by-hop manner. DTN encapsulates IP application data into bundles using its convergence layer.

5.2 System architecture

With reference to Figure 11, there are two main components of proposed seamless DTN proxy for SMTP, namely ProxyClient and ProxyServer. The functionalities of ProxyClient and ProxyServer resemble the ones proposed in the earlier chapter, DTN proxy for HTTP. ProxyClient can be deployed to access point of local network or installed to user device. The operation of email client remains unchanged. When email client at DTN node attempts to communicate with (public or private) email servers, its SMTP traffic would be redirected seamlessly to ProxyClient. ProxyClient would then perform bundling procedures for the redirected SMTP traffic and forward the DTN bundle to ProxyServer, which resides at the spot with both DTN and Internet connections. ProxyServer would then unpack DTN bundle for original SMTP traffic and forward to the actual email server.

Figure 11: Proposed seamless DTN proxy for SMTP

5.2.1 ProxyClient

ProxyClient provides seamless interception of email traffic for further processing and standard SMTP/POP3 interface to email client. In order to intercept email traffic, traffic is filtered based on the well-known ports of the email service and redirected to the service port of ProxyClient. Once TCP connection is established between email client and ProxyClient, email traffic would be received from the corresponding TCP socket. The received email traffic would then be packed into DTN bundles and sent to ProxyServer, together with information of the designated email server.

5.2.2 ProxyServer

ProxyServer resides in a location with both Internet and DTN connectivity. Its role is to process the DTN bundles received from ProxyClient and interact with email server to send and receive emails. ProxyServer also has a set of SMTP/POP3 interface and acts as email client when communicating with actual email servers.

5.2.3 SMTP operation

When ProxyClient receives data from email client (Mozilla Thunderbird), it would perform bundling of SMTP messages into DTN bundles and tag the payload with the port and address of the designated email server, which is extracted when the original connection from the mail application is redirected. These bundles are then transferred to ProxyServer.

5.2.4 POP3 operation

When email client attempts to receive emails from email server, its traffic would be redirected to ProxyClient. When the email client is checking whether there are new emails, it is difficult for local ProxyClient to reply because it does not know whether there are new emails at the user's email server. In this scheme, the local ProxyClient would note down the request, but does not reply to the email client until network connectivity resumes such that it can retrieve the information from the remote email server. During this period, the TCP connection between the email client and ProxyClient are kept active. The POP3 requests from the email client will be cached and sent as DTN bundle to ProxyServer and then conveyed to the email server. The response from the email server would also be transferred back and passed to the email client through the DTN environment. The reason for keeping the connection alive

without replying is because if the connection is closed due to reply from ProxyClient to the email client, either positive or negative, it is unknown whether the email client will initiate another round of checking when the network connectivity resumes. It is possible that email client always get replies stating “no new email” but in fact there are new emails at the email server. Such scenario is certainly not desirable.

5.3 Evaluation and discussion

Most popular protocols used in today’s email communication are SMTP/POP3 and SMTP/IMAP. To implement all those protocols in evaluating seamless DTN proxy for email access, it would require huge amount of time to develop the modules. Due to time constraint, only SMTP module in seamless DTN proxy for email was implemented. Thus, the performance benchmark of seamless DTN SMTP proxy is evaluated against traditional SMTP over IP under different network situations.

Figure 12: DTN SMTP proxy evaluation testbed topology

The network topology shown in Figure 12 is used to evaluate the performance. It involves 4 nodes, namely an email user with ProxyClient, two DTN relay nodes and a ProxyServer. In order to have consistent measurement results, a private email server is used for the benchmark evaluation and connected directly to ProxyServer. Delivery delay is the benchmark parameter used for performance measurement, and computed as the duration between the commencements of email transmission at the email client until the email arrival at the private email server. Mozilla Thunderbird is used for email client application and Postfix for private email server. The following three network scenarios are used in the evaluation.

5.3.1 Scenario 1: Steady network connectivity

This scenario assumes that there are no disruptions at all in the network and is used to measure the additional overhead or delay introduced by seamless DTN SMTP proxy. In Figure 13, significant delivery delay was observed when email content increases (referring to larger email attachment). The delay was due to store-and-forward nature of DTN.

Figure 13 : SMTP survival under steady network connectivity

5.3.2 Scenario 2: Intermittent network connectivity

This scenario assumes that there is a periodic disruption (5-second link down and 5-second link up). Measurement results obtained in Figure 14 show that sending email over IP with more than 1MB content failed to get delivered to the destination because the periodic 5-second time window is not enough to properly transfer the message. In contrast, email over DTN has no issue on delivery to the destination regardless of the message size.

Figure 14: SMTP survival under intermittent network connectivity

5.3.3 Scenario 3: Long haul network interruption

The last scenario was aimed to test the feasibility of email transmission when 10-second long disruption occurred before the commencement of the transmission. The results measured in Figure 15 showed that email delivery over IP is a total failure under this scenario because SMTP uses TCP connection where the connection establishment could not complete before the actual SMTP data transfer commences. With seamless DTN SMTP proxy, DTN is able to store the SMTP messages at ProxyClient even though the link is down, and only forwards the stored message to the next hop when the link is re-established.

Figure 15: SMTP survival under long network disruption

5.4 Summary

Seamless DTN proxy is capable of transporting IP-based ADU (Application Data Unit) over DTN network under intermittent network connectivity. It is accomplished without the awareness of user at the application layer. Therefore, no modifications or configurations are made on the user application, just as explained in earlier sections using SMTP email transport over DTN. In contrast to DTN proxy for HTTP proposed in the previous chapter, seamless DTN proxy improves the flexibility and prevents user from tedious configurations when multiple email clients are used such Mozilla Thunderbird, Microsoft Outlook and other third-party clients.

In next chapter, author addresses the issue on complexity and tediousness in creating different seamless DTN proxy dedicated for a specific protocol. Instead, generic seamless DTN proxy should be designed to curb the problem.

6 GENERIC SEAMLESS DTN PROXY

6.1 Overview

In the previous chapter, the author proposes a DTN proxy agent that seamlessly allows SMTP data to be transported between source node and destination node via DTN network. It provides a more reliable email delivery under disrupted environment with the expense of additional DTN overhead. It is achieved using iptables network filtering policy and packet redirection management. Thus, user at application layer would never notice the packet handling by another application.

Since the launch of the Internet, there have been many protocols [24] created to support various types of applications. Commonly used and standardized protocols are SMTP/IMAP/POP3 protocols for email system, HTTP for web browsing, FTP for file transfer and so on. In [24] [25], IANA has assigned and registered port number for IP traffic under UDP [26] and TCP [27] transport. There are thousands of well-known registered ports. For instance, DNS resolution uses UDP port number 53 to locate DNS [26] server so as to obtain IP address from a given email or web URL.

To implement seamless DTN proxy agent to cater to each protocol defined in the RFC1700, it would be a massive and time-consuming task. Although development workload and duration are not the issues, it is not possible for DTN node to know which DTN proxy agents to be activated without knowing which application a user would run. Of course, it is possible to activate all DTN proxy agents on a DTN node but it would require a powerful processor and huge memory to be used. In a nutshell, it is unrealistic to implement all or most commonly used protocol, as each seamless DTN proxy agent is run as one single application in the user space.

However, UDP and TCP are the main and fundamental protocols used under OSI transport layers. These two prudent transport protocols are heavily utilized and overlaid by other upper OSI layer. For instance, email system uses SMTP (Simple Mail Transport Protocol) which overlays on TCP/IP protocol. The other examples are file transfer protocol (FTP) over TCP/IP, domain resolver (DNS) over UDP/IP and so on. The trend for IP packets contributing to today's Internet has evolved from static workstations to mobile handheld devices. Social media, resource sharing and cloud platform have heavily used RESTful web API. RESTful web [25] API is an abstraction

of the architecture of the World Wide Web (WWW) and has been applied to the development of web services.

Thus, the author is motivated to propose a simpler and generic DTN proxy agent, and refers to it as Generic Seamless DTN proxy or GSProxy. With GSProxy design, only two protocols need to be developed, namely UDP-based and TCP-based.

6.2 System Architecture

DTN GSProxy system architecture shown in Figure 16 reuses most modules implemented in seamless DTN proxy agent in the previous chapter. The reused modules are IPTABLES, NETFILTER, TCP HANDLER AND DTN API. DTN GSProxy must support UDP packet handling (UDP HANDLER) as well as TCP packets. Another feature, POLICY MANAGER, is developed to provide creation and removal of TCP and UDP handlers automatically and seamlessly depending on the policy rules defined in the configuration file. Each module is elaborated in detail in the subsequent sub-sections. These modules are considered as front-end parts of the entire DTN GSProxy ecosystem. It still requires back-end support, namely DTN daemon (server) must start before initiating DTN GSProxy. DTN daemon is in charge of DTN transmission, DTN reception, DTN neighbor discovery, DTN beacon advertisement, DTN link detection, DTN routing and DTN store-forward capabilities.

Figure 16: DTN GSProxy ecosystem design

6.2.1 Policy Manager and Rule

Policy Manager reads a configuration file residing at the DTN proxy node to define its policy rules. Each line in the configuration file defines a policy rule. A policy rule contains three parameters; protocol, port number and remarks with comma delimiter. Protocol can only be **udp** or **tcp**. Port number ranges between 1 and 65536. Remarks field is double-quoted string and can be used to describe human-readable policy rule. Duplicate policy rule in the configuration file will be skipped. A sample of the configuration file is as follows:

```
udp,53,"proxy for dns"  
tcp,80,"proxy for http"  
udp,53,"duplicate proxy for dns"  
udp,8000,"proxy for unregistered and private port"
```

Policy Manager instantiates an appropriate (UDP or TCP) handler, reserves ports (from a pool of listening port ranging between 30000 and 40000) and defines iptables rules.

6.2.2 Netfilter and Iptables

Netfilter is a framework in Linux Kernel which allows various networking functionalities. Linux operating system segregates system memory into two distinct spaces, namely kernel space and user space. Kernel space is where the core of operating system executes and provides its service to user space. Processes running under user space can only access a limited part of memory, whereas the kernel has full access and higher privileges to all memory. User space processes can only access kernel space's resource via system call.

Packet manipulation is more efficient under kernel space than user space as the former has more access to full memory. Implementation of packet manipulation under kernel space is possible through loadable kernel module (LKM). With LKM, packet manipulation (such as dropping packet) can be quickly verified rather than passed and verified under user space. LKM modules can be created with Linux kernel programming. However, there are some hindrances such as;

- Assumption that all DTN node (machines) are running Linux operating system with same version of Linux kernel.
- Level of difficulty in writing a program under kernel space than user space.

The above listed difficulties for implementation are also encountered if it is under user space. Fortunately, there is a helpful tool beside the stated implementations, namely iptables. Iptables is a user-space software tool that allows a Linux system administrator to configure IP tables provided by Linux kernel firewall and the chains and rules it stores. It can be used to configure Netfilter chains shown in Figure 17 too.

Figure 17: Packet flow in netfilter

6.2.3 DTN API

DTN API from DTN 2.6.0 reference implementation provides lots of DTN functions. However, DTN GSProxy utilizes only two functions from DTN API;

- DTN Bundling. A process where IP packet is encapsulated into DTN bundle.
- DTN Unbundling. A process where IP packet is extracted from DTN bundle.

DTN bundle comprises DTN header and DTN payload. DTN header must at least be given with information about DTN source (ingress node) EID, DTN destination (egress node), bundle type and payload size. DTN EID has been mentioned many times in earlier chapters as DTN endpoint ID. DTN payload is where application (IP) packet resides at. Application packet has different data structures for TCP and UDP handlers.

There are four bundling events happening for a round-trip packet shown in Figure 18. Bundling event processes differently for UDP and TCP packet by its dedicated handler respectively.

- Local bundling: transforms forwarding IP traffic to DTN traffic at DTN ingress node.
- Remote unbundling: transforms DTN traffic to forwarding IP traffic at DTN egress node.
- Remote bundling: transforms returning IP traffic to DTN traffic at DTN egress node.
- Local unbundling: transforms DTN traffic to returning IP traffic at DTN ingress node.

Figure 18: Bundling events for a round-trip packet

6.2.4 TCP handler

TCP/IP traffic requires an end-to-end connection from source to destination. TCP data packet transmission could only commence after successful 3-way handshaking (SYN, SYN/ACK and ACK) between source node and destination node. The author refers to 3-way handshaking process as TCP control packet. Under disruptive environment, TCP control packet delivery over DTN transport might take a much longer duration to complete the handshaking process as there might be intermittent link among DTN node from source to destination node. It is unwise to transport TCP control packet from source to destination and vice versa. TCP control packet is mainly used to trigger the commencement of TCP data packet transfer so the former could be handled and acknowledged locally at DTN ingress node. Referring to PEP approach, DTN ingress node manages TCP control packet with IP source node by acting as surrogate IP destination node. The following definitions are used to illustrate nodes involved in TCP control and the data packet traffic flow diagram is shown in Figure 19.

- IP Source node. Node where the data originates from the application layer.
- DTN ingress node. Node where DTN GSProxy resides and receives the TCP packet from the IP Source node. It acts as a gateway between the IP network and DTN network.
- DTN node. Any intermediate DTN node with or without DTN GSProxy activated. It handles normal DTN routing, DTN link detection and recovery.
- DTN egress node. Node where DTN GSProxy resides and sends TCP packets out either directly to the IP destination node or indirectly via the IP network (Internet). This node is also a gateway and the DTN EID must be known by the DTN ingress node.
- IP destination node. Node where the data sinks to the target's application layer.

Figure 19: TCP packet flow over DTN network

TCP local bundling

In order to handle TCP control and data packets separately, each TCP policy rule requires two listening ports for different packet types. TCP data packet only contains socket ID and application data unit and does not have transfer information such as source IP address, source port, destination IP address and destination port. These information can be obtained from TCP control packet and saved in a TCP hash table with socket ID as key parameter. DTN bundle comprises DTN header and DTN payload. DTN payload for TCP/IP delivery must include TCP info (Socket ID, Source IP address, Source port, Destination IP address and Destination port) and TCP data. TCP info is retrieved from TCP hash table by matching socket ID found in TCP data packet. TCP info entry of TCP hash table is removed by either graceful termination via TCP BYE message or forced termination from the IP Source node. In other words, each TCP info entry is mapped to a TCP session.

TCP remote unbundling

When DTN bundle reaches DTN egress node, TCP information is extracted to clone the same TCP hash table as the DTN ingress node. Destination IP address and port number from the TCP info are used to create a new direct TCP session to the actual IP Destination node. TCP data is inserted to the new created TCP session.

TCP remote bundling

Direct TCP session opened during TCP remote unbundling is maintained until the end of transmission from the IP Source node. Each returned TCP packet is then processed through bundling event and accompanied with the same piece of TCP info used to create the TCP session. DTN bundle is returned to the DTN ingress node.

TCP local unbundling

When returned DTN bundle is back at the DTN ingress node, TCP data is pushed to the active TCP session matched with the socket ID found in the TCP info of the DTN bundle.

Iptable rule

Two iptable rules are needed for each TCP session. The sample iptable rules are as follows with the destination IP address within 192.168.1.0/24 network, netfilter queue number 0 assigned and netfilter redirection port number of 30000.

```
iptables -A PREROUTING -t mangle -d 192.168.1.0/24 -p tcp -flags syn -j NFQUEUE
-queue-num 0
iptables -A PREROUTING -t nat -d 192.168.1.0/24 -p tcp -j REDIRECT -to-port
30000
```

6.2.5 UDP handler

In contrast to TCP handler, UDP handler manages UDP data packet in a much simpler way as UDP is best-effort data delivery protocol. UDP is hop-by-hop packet delivery and does not require acknowledgement from the IP destination node to initiate data transfer. UDP handler only needs one single listening port and netfilter queue for each policy rule. The same definitions are used to illustrate nodes involved in UDP data packet traffic flow diagram which is shown in Figure 20.

Figure 20: UDP packet flow over DTN network

UDP local bundling

Bundling event at DTN ingress node is pretty straight-forward. The entire UDP/IP is marked and dumped into DTN payload before being transformed into DTN bundle.

UDP remote unbundling

During unbundling event at DTN egress node, DTN payload containing original IP header, UDP header and UDP data is injected into local network and forwarded as normal IP packet. Assuming same policy rules at all DTN GSProxy node, the injected UDP/IP packet is caught by iptable rules before being forwarded to the next hop. The reason behind this phenomenon is because the injected packet at the DTN egress node has the exact content as the original packet entering the DTN ingress node. This has a deep impact in forwarding the original packet to the IP destination node as the packet is caught and injected in an infinite loop at the DTN egress node and would cause a

memory crash. In order to prevent an infinite loop of the injected packet at the DTN egress node, the original packet at the DTN ingress node must be marked (with 0x1 value) before encapsulation into the DTN bundle. Thus, the injected (marked) packet would not be caught into an infinite loop anymore near the DTN egress node and successfully be forwarded to the IP destination node.

UDP remote bundling

This event is the same as UDP local bundling but with the returned UDP data and the reversed IP address and port number.

UDP local unbundling

This event is the same as UDP remote unbundling but with the returned UDP data and reversed IP address and port number. Again, this UDP/IP packet must not be captured into netfilter by iptables rules.

Iptable rule

One iptable rule is needed for each TCP session. The sample iptable rule is as follows with the destination IP address within 192.168.1.0/24 network and netfilter redirection port number of 30001.

```
iptables -A PREROUTING -t nat -d 192.168.1.0/24 -p udp -m mark --mark 1 -j  
REDIRECT -to-port 30001
```

6.3 Discussion

6.3.1 DTN GSProxy agent against dedicated seamless DTN proxy agent

Single DTN GSProxy agent in Figure 21 replaces a number of dedicated seamless DTN proxy agent (such as DTN SMTP proxy agent implemented in the previous chapter). Author uses common email and web surfing applications to compare the number of proxy agents required to connect to its respected servers. DNS server is contacted prior to email/web server in order to resolve human-readable URL to IP address. Thus, there are three dedicated seamless DTN proxy agents to be activated compared to one single DTN GSProxy agent. The former requires more CPU and memory resource to be allocated for individual agent instances, while the latter needs only a single instance of proxy agent with auto-creation and removal of UDP or TCP handler. DTN GSProxy offers more flexibility to cater to new future protocol, instead of developing a new dedicated DTN proxy agent.

Figure 21: DTN proxy agent comparison

6.3.2 Cross-boundary DTN network with priority tag and DTN ESCL

Potential use of DTN GSProxy proposed in this chapter together with enhanced serial convergence layer (DTN ESCL implemented Chapter 3) may be applied to wide area networking (WAN) disruption such as submarine cable disruption in 2008 [32] and 2011 [33]. The incidents caused global Internet access disruption and affected millions of users. Priority field in DTN bundle header can be utilized to match the level of urgency, delivery throughput and transmission range. Currently, the author uses the available value for DTN bundle priority according to the DTN bundle specification defined in RFC5050. There are four priority values with higher values being of higher priority tabulated in Table 7.

Table 7: Bundle priority and delivery pattern

Priority value	Message urgency	Payload size	Delivery throughput	Transmission range	Transmission medium
0x3 (reserved)	Very high	Extremely small	Extremely small	Farthest	• Amateur radio (VHF) via DTN ESCL
0x2 (expedited)	High	Small	Small	Far range	• Amateur radio (UHF) via DTN ESCL
0x1 (normal)	Normal	Medium	Medium	Mid-range	• Cellular network via DTN UDP or TCP convergence layer
0x0 (bulk)	Low	Very large	Very large	Short range	• Bluetooth via DTN Bluetooth convergence layer • Wi-Fi (a/b/g/n/ac) via DTN Ethernet convergence layer

Smart distribution and delivery of DTN bundle depending on message urgency is possible to be deployed by using DTN bundle priority together with DTN GSProxy and DTN ESCL. Under normal circumstances, message with higher urgency has very brief information and should be delivered as fast as possible to reach the destination. For instance, a powerful earthquake took place in mountainous region, and rescue team deployed several DTN node as shown in Figure 22. Command center is a DTN node that has Internet access. Information about the missing, casualty and survivors must be regularly updated and reported to the command center. This piece of information can be tagged with “Very High” urgency. In addition, information on personal particulars

for affected people are commonly described in text and can be delivered with “High” message urgency. Photo, audio and video about affected area are generally huge in file size and must be transported in higher speed transmission medium, and should not affect messages with “Very High” urgency. Therefore, these high throughput information must be transported on separate DTN link/route.

Figure 22: Cross-boundary DTN network with different priority tag

Additional priority field in DTN GSProxy’s configuration file must be inserted before the remarks field. The allowed values for the priority field are **0, 1, 2** and **3** corresponding to the values listed in Table 7.

```
udp,5000,3,"count for missing, casualty and survival"
udp,5001,2,"personal particulars of affected people"
udp,5002,1,"photo or audio"
udp,5003,0,"video record"
```

6.4 Summary

DTN GSProxy is designed to transport all UDP/IP and TCP/IP data packets over DTN network under intermittent network connectivity. It is accomplished without the awareness of user at the application layer. Administrator at DTN GSProxy may easily add or remove policy rules in the local configuration file for which protocol and port number DTN GSProxy will process the data. With DTN GSProxy implementation, it is redundant to develop dedicated seamless DTN proxy agent for each protocol which will greatly involve tremendous development time and design complexity.

7 CONCLUSION

The implementation of neighbor discovery and multiple access in DTN Serial Convergence Layer allows DTN communication via alternate longer range communication besides the preferred short range Wi-Fi or Bluetooth communication. It comes with the trade-off of lower throughput. This enhanced convergence layer is not proposed to replace the current communication medium but to complement as bridge communication between two distant and isolated area such neighboring islands in archipelago.

DTN proxy agent allows transport of IP packets from existing IP-based applications over a more disruptive environment due to its store and forward capabilities. This saves user's time to develop DTN-based application to capture, handle and deliver a particular IP-based application such as HTTP traffic described in Chapter 4.

However, the user is aware of the existence of DTN proxy. To address this issue, a seamless mode of DTN proxy is developed and implemented based on PEP approach. The main difference between DTN proxy and seamless DTN proxy lies in the packet capturing from the source node and the packet injection to the destination node.

DTN proxy is further evolved to a generic and seamless type dubbed as DTN GSProxy. This type of proxy gives more flexibility to seamlessly transport UDP-based and/or TCP-based IP packets over DTN network. There is no need to develop different dedicated seamless DTN proxies for different protocols, as DTN GSProxy has built-in UDP and TCP handler engines to be deployed in as many threads as the DTN node can support to handle multiple applications at the same time. With integration of DTN ESCL and use of DTN bundle priority tag, this allows smart distribution of different levels of message urgency over different data throughput and transmission range.

8 FUTURE WORKS

8.1 Potential application layer misbehavior

Although DTN GSProxy agent is able to manage data delivery properly, there are still some unexpected actions in the application layer. For example, a web server expects to receive HTTP requests in a certain period of time after the TCP connection is established. If it takes too long to send the message to the web server, the server will reply with a “**408 Request Timeout**” message to the client. Similarly, web browsers can also impose timeouts on the requested contents from the webserver. These timeouts are configurable settings of the application and can be adjusted according to the needs. Similarly, web browser can also impose timeouts on the requested contents from the web server. These timeouts are configurable settings of the application and can be adjusted according to the needs.

Another example is UDP-based video streaming using VLC media player. Under disruptive situation, all video packets would still manage to reach client side. However, video viewing may not be as smooth as the original video content at the IP source node. VLC media player at the client side has default configuration to drop delayed and irregular arrival time of a video packet. This misbehavior is beyond the control of DTN GSProxy. The delay buffer in VLC media player can be configured accordingly to mitigate the problem.

8.2 Secured protocol

Some protocol has added an additional security layer to secure message exchange. Support for secured application is still not resolved in DTN GSProxy due to the key exchange during connection setup. For example, HTTPS is HTTP with added security layer. HTTPS web server sends its certificate signed by trusted organizations and the browser verifies the certificate to confirm the identity of the server. As the certificate contains the domain name of the server, each domain has its own certificate. In order to use HTTP/DTN proxy to support HTTPS, the proxy must be able to generate a valid certificate for each HTTPS web server it is communicating with. This essentially resembles man-in-the-middle attack and might not be a desirable solution. A better solution might be using plain HTTP protocol to transfer encrypted data. Alternatively,

the plain HTTP messages can be encrypted at the lower layers before sending out to the network. On the other hand, it is possible to generate a self-signed certificate and use it for all the websites. This will trigger the warnings from the browsers but it can be a temporary solution for testing purpose.

9 AUTHOR'S PUBLICATIONS

The following research papers has been published by authors in various conferences.

1. Z.Bong, C.K. Yeo, "DTN Serial Convergence Layer with Multiple Access Control and Neighbour Discovery", IEEE 74th Vehicular Technology Conference (VTC), Sep 2011.
2. Z.Bong, P.B. Sng, C.K. Yeo, "Web Access over Delay Tolerant Networks", International Conference on Advancements in Information Technology (ICAIT), Jun 2012.
3. Y. Xia, Z. Bong, S. Ishikawa, C. K. Yeo, Y Mao, "Enabling Email Access under Intermittent Connectivity", IEEE Wireless Communications and Networking Conference (WCNC), Apr 2013.

10 REFERENCES

- [1] V. Cerf, A. Hooke, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-Tolerant Networking Architecture," in *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*, New York, NY, USA, 2003.
- [3] K. Fall and S. Farrell, "DTN: an architectural retrospective," in *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," in *RFC 5050 (Experimental)*, Nov. 2007.
- [5] "Delay Tolerant Networking Research Group - Code," [Online]. Available: <http://www.dtnrg.org/wiki/Code>.
- [6] "EIA Standard RS-232-C Interface between Data Terminal Equipment and Data Communication Equipment Employing Serial Data Interchange," Aug. 1969.
- [7] Lauri Peltola, "DTN-enabled Web Server," in *Helsinki University of Technology*, May. 2008.
- [8] M. Khabbaz, C. Assi, and W. Fawaz, "Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges," in *Communications Surveys & Tutorials, IEEE*, vol. PP, no. 99, pp. 1–34, 2011.
- [9] T. Hyyryläinen, T. Kärkkäinen, C. Luo, V. Jaspertas, J. Karvo, and J. Ott, "Opportunistic email distribution and access in challenged heterogeneous environments," in *CHANTS '07: Proceedings of the second ACM workshop on Challenged networks. ACM Request Permissions*, Sep. 2007.
- [10] A. Balasubramanian, Y. Zhou, W. B. Croft, B. N. Levine, A. Venkataramani, "Web search from a bus of the second ACM workshop on Challenged networks. ACM Request Permissions," in *CHANTS '07: Proceedings*, Sep. 2007.
- [11] A. Lindgren, "Social networking in a disconnected network: fbDTN: facebook over DTN," in *CHANTS '11: Proceedings of the 6th ACM workshop on Challenged networks. ACM*, Sep. 2011.
- [12] K. Zaragoza, N. Thai, and T. Christensen, "An implementation for accessing twitter across challenged networks," in *CHANTS '11: Proceedings of the 6th ACM workshop on Challenged networks. ACM*, Sep. 2011.
- [13] J. Klensin, "Simple Mail Transfer Protocol," in *RFC 2821 (Proposed Standard)*, Apr. 2001.
- [14] J. Myers, M. Rose, "Post Office Protocol - Version 3," in *RFC 1939 (Standard)*, May 1996.

- [15] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations," in *RFC 3135 (Informational)*, Jun. 2001.
- [16] C. C. a. R. Firrincieli, "TCP Hybla: a TCP Enhancement for Heterogeneous Networks," in *International Journal of Satellite Communications and Networking*, vol. 22, n. 5, pp. 547-566, Sep. 2004.
- [17] C. Caini, P. Cornice, R. Firrincieli, M. Livini and D. Lacamera, "TCP, PEP and DTN performance on disruptive satellite channels," in *Satellite and Space Communications, 2009. IWSSC 2009*, 2009.
- [18] V. Bharghavan, "MACAW: A Medium Access Protocol for Wireless LAN's," in pp. 212-225, Aug. 1994.
- [19] M. Demmer, E. Brewer, K. Fall, S. Jain, M. Ho, R. Patra, "Implementing Delay Tolerant Networking," in *IRB-TR-04-020*, 28 December, 2004.
- [20] M. Demmer, "Architecture of DTN2," Delay Tolerant Network Research Group, [Online]. Available: <http://dtn.sourceforge.net/DTN2/doc/manual/arch.html#DTN2%20api>.
- [21] Sreeram Ramachandran, "Web metrics: Size and number of resources," in *Google Code*, May. 2010.
- [22] Garrett, Jesse James, "Ajax: A New Approach to Web Applications," AdaptivePath.com, 2 July 2008.
- [23] "Usage of Flash for websites," 4 March 2013. [Online]. Available: <http://w3techs.com/technologies/details/cp-flash/all/all>.
- [24] "Server-side Javascript Back with a Vengeance," Read Write Web, 17 December 2009. [Online]. Available: http://www.readwriteweb.com/archives/server-side_javascript_back_with_a_vengeance.php.
- [25] Artemios G. Voyiatzis, "A Survey of Delay- and Disruption-Tolerant Networking Applications," in *Journal of Internet Engineering*, Vol 5, No 1, Jun 2012.
- [26] "Service Name and Transport Protocol Port Number Registry," in *RFC 6335*, Aug 2011.
- [27] J. R. a. J. Postel, "Assigned numbers," in *IETF RFC1700*, Oct 1994.
- [28] J. Postel, "User Datagram Protocol," in *IETF RFC 768*, Aug 1980.
- [29] I. S. I. - U. o. S. California, "Transmission Control Packet," in *IETF RFC 793*, Sept 1981.
- [30] P. Mockapetris, "Domain Names - Implementation and Specification," in *IETF RFC 1035*, Nov 1987.
- [31] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern Web Architecture," in *ACM Transactions on Internet Technology (TOIT)*, May 2002.
- [32] "Severed cables disrupt Internet," BBC News, 31 January 2008. [Online]. Available: <http://news.bbc.co.uk/1/hi/technology/7218008.stm>.
- [33] "Cable cuts could slow Emirates Internet," Yahoo, 26 December 2011. [Online]. Available: <http://www.emirates247.com/news/emirates/cable-cut-slows-down-internet-in-uae-2011-12-26-1.434621>.