Encapsulation of OpenFlow over Delay-Tolerant Networking (DTN) Using the Bundle Protocol

draft-fan-dtn-openflow-over-bp-00

Xiaojing Fan, Huachun Zhou, Jie Qu
Beijing Jiaotong University
23111043@bjtu.edu.cn

IETF 125 Shenzhen, DTN WG

Agenda

- Introduction
- Problem Statement and Use Cases
- Architecture Overview
- Encapsulation and Delivery Rules
- Endpoint Identification and Addressing
- Experimental Verification
- Security Considerations

Introduction

- OpenFlow enables centralized control in SDN via controller–switch communication
- DTN environments also require centralized policy dissemination
- Traditional OpenFlow control channels rely on TCP/IP and stable end-to-end paths
- DTNs lack such paths, making TCP-based control channels unreliable
- The Bundle Protocol (BP) provides store-and-forward communication for disrupted networks
- BP supports naming, addressing, and transport across heterogeneous links
- This work describes the use of BP to carry OpenFlow control messages in DTNs

Problem Statement

OpenFlow assumes stable end-to-end connectivity. DTNs violate this assumption

Control messages must tolerate:

- Delay and disruption
- Intermittent links
- Non-ideal delivery behavior

Use Cases

- Intermittent connectivity

  Controller–switch links available only at scheduled or sporadic times
- Long-delay paths

  Control traffic over satellite or deep-space links
- Disrupted multi-hop forwarding

  Messages relayed via intermediate DTN nodes using store-and-forward

Architecture Overview

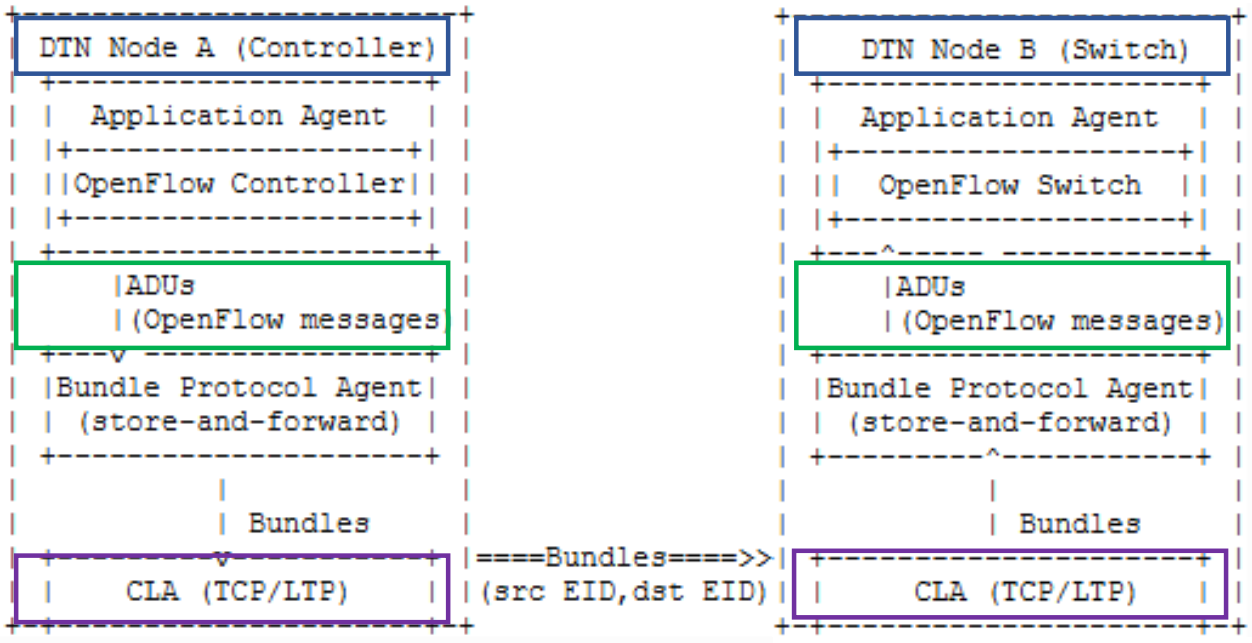


Figure 1. Architectural overview of OpenFlow control message carriage over a DTN using BP

DTN-capable nodes

- Controller and switch deployed on DTN nodes
- OpenFlow logic runs at the application layer

Message encapsulation

- OpenFlow messages treated as application data units (ADUs)
- ADUs encapsulated into Bundles by the Bundle Protocol Agent

Convergence layers

- Bundles mapped to underlying links via CLAs
- Multiple CLAs supported (e.g., TCP, LTP)

Encapsulation & Delivery Rules

Encapsulation Overview

- OpenFlow signaling carried as BP payload
- OpenFlow messages treated as application data units (ADUs)
- ADUs encapsulated into Bundles by the BP agent
- Encapsulation includes:
  - Underlying carrier headers (e.g., Ethernet / IP / TCP, UDP, LTP)
  - BP Primary Block and optional Extension Blocks
  - Bundle Payload containing OpenFlow signaling data
- BP may operate over different CLAs
- **No specific tunneling protocol** is defined or required

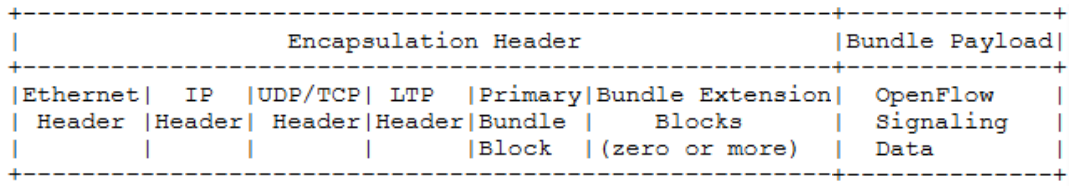


Figure 2. Illustrative example of OpenFlow control message encapsulation as a BP payload

One-to-One Message Mapping

- Each Bundle carries exactly **one** OpenFlow message
- OpenFlow messages carried in native wire format
- Payload treated as opaque data by BP forwarding nodes
- No message aggregation unless explicitly defined and agreed

Encapsulation & Delivery Rules

Fragmentation and Reassembly

- Bundle fragmentation MAY be used if supported by BP
- Fragmentation MUST NOT be used when:
  - "Do not fragment" is indicated
  - Anonymous source is used
- Fragments MUST be fully reassembled before delivery
- Partial OpenFlow messages MUST NOT be exposed

Duplicate Suppression and Ordering

- Receiver SHOULD maintain a bounded duplicate cache
- Cache keys derived from BP identifiers (source + timestamp)
- Arrival order MUST NOT be assumed to reflect send order
- Ordering control MAY use:
  - OpenFlow transaction identifier (xid)
  - Explicit sequence numbers

DTN Delivery Considerations

- No assumption of interactive request/response timing
- Duplicate delivery and reordering may occur
- OpenFlow processing MUST tolerate duplicates

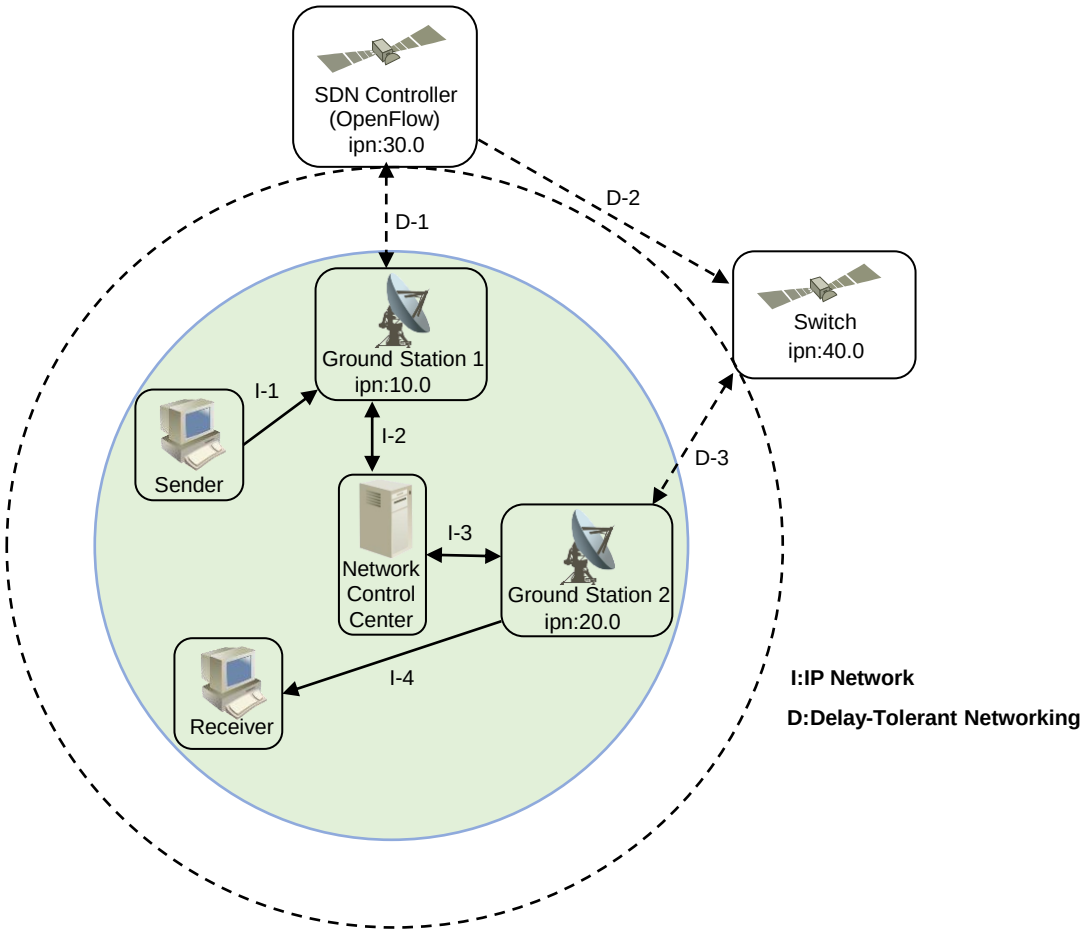
Bundle Lifetime

- Each Bundle MUST have a finite lifetime
- Expired OpenFlow messages MUST NOT be applied

Endpoint Identification and Addressing

- OpenFlow controllers and switches identified by BP EIDs
- Each OpenFlow entity uses a stable, unique destination EID
- Naming is deployment-specific
- Dedicated service component MAY be used (e.g., /of)
- Singleton (unicast) endpoints SHOULD be used
- Communication is bidirectional (control, telemetry)
- Payloads delivered to the local OpenFlow entity based on EID

Experimental Verification



- Prototype implemented based on **ION-DTN**
- OpenFlow (v1.0) control messages carried over DTN using BP
- Validated in satellite–terrestrial scenarios:
  - Long delay and intermittent connectivity
  - Store-and-forward message delivery

- **Verified**:
  - Feasibility of OpenFlow signaling over DTN
  - Correct encapsulation and delivery semantics
- No modifications required to OpenFlow protocol
- Experimental results reported in prior published work

[1] Fan X, Zhou H, Zheng T, and Yan J. Service orchestration strategy in STINs: A multi-task SFC mapping approach[J]. IEEE Transactions on Vehicular Technology, 2025, (Under Review).
[2] Li T, Zhou H, Luo H, et al. SERvICE: A software defined framework for integrated space-terrestrial satellite communication[J]. IEEE Transactions on Mobile Computing, 2017, 17(3): 703-716.
[3] Feng B, Zhou H, Zhang H, et al. HetNet: A flexible architecture for heterogeneous satellite-terrestrial networks[J]. IEEE network, 2017, 31(6): 86-92.

Security Considerations

- Does not modify OpenFlow or BP security properties
- DTN store-and-forward may increase exposure
- Deployments should use existing BP security mechanisms

Thanks!

Questions? Comments?