17th International Conference on Space Operations, Dubai, United Arab Emirates, 6 - 10 March 2023.

SPACEOPS-2023, ID # 338

Compressed Bundle Reporting for Reliable Bundle Transmission in Disruption-Tolerant Networking

Aida S. Manole[a], Felix Flentge[b], Stefan Schmid[c], Arash Pourdamghani[d], Max Frank[e]

[a] *European Space Agency, Robert-Bosch-Straße 5, Darmstadt, Germany, aida.manole0910@gmail.com*
[b] *European Space Agency, Robert-Bosch-Straße 5, Darmstadt, Germany, felix.flentge@esa.int*
[c] *Technical University of Berlin, Einsteinufer 17, Berlin, Germany, stefan.schmid@tu-berlin.de*
[d] *Technical University of Berlin, Einsteinufer 17, Berlin, Germany, pourdamghani@tu-berlin.de*
[e] *Technical University of Berlin, Einsteinufer 17, Berlin, Germany, mfranke@inet.tu-berlin.de*

Abstract

Regarded as one of the main communication protocols implementing the Disruption-Tolerant Networking (DTN) paradigm, Bundle Protocol (BP) establishes an overlay network over heterogeneous networks. With the introduction of the new protocol version, Bundle Protocol version 7 (BPv7), there was a need for a new mechanism that would ensure reliability at the bundle layer for scenarios where the lower layer stack does not offer such capabilities. In this paper we develop a BP extension, Compressed Bundle Reporting (CBR), which provides aggregated status reporting of bundles, and enables re-transmission strategies, including a possible integration with a Custody Transfer-like mechanisms. The solution is prototyped using ESA's BP implementation and validated through experiments. We study the behaviour of CBR during four experiments, with two types of link reliability (1% and 5% bundle loss, respectively) and two different configurations for the protocol-specific parameters. During the worst link conditions, we manage to use reduce the number of bytes required to report on the status of a bundle by a factor of 10 with the help of aggregated reporting, considering the size of BPv7 bundle status reports as a baseline. Naturally, we report the best results on the more reliable link, managing to reduce the number of bytes per reported bundle by a factor over 100. A qualitative study of CBR and previously existing mechanisms for reliable data transmission at the bundle layer is performed to assess the benefits of the proposed solution.

Keywords: Bundle Protocol, Disruption Tolerant Networking, Aggregated Reporting, Reliable Bundle Transmission, Custody Transfer

Acronyms/Abbreviations
E2E – End-to-End
CLA – Convergence Layer Adapter
UDPCL – User Datagram Protocol Convergence Layer
TCPCL – Transmission Control Protocol Convergence Layer
LTPCL – Licklider Transmission Protocol Convergence Layer
AA – Application Agent
BPA – Bundle Protocol Agent
PDU – Protocol Data Unit
RTT – round trip time
BIBE – Bundle-in-Bundle Encapsulation
ACS – Aggregated Custody Signal
CBOR – Concise Binary Object Representation
ADU – Application Data Unit
CCSDS – Consultative Committee for Space Data Systems
CBR – Compressed Bundle Reporting
CRS – Compressed Reporting Signal
CREB – Compressed Reporting Extension Block

1. Introduction

This work is motivated by the lack of efficient bundles status reporting and the removal of custody transfer as a mechanism for reliable bundle transfer in BPv7. In the following we will discuss the DTN field in general, then focus on BP, which is a core protocol of an implementation of DTN, and finally related work on bundle reliable transmission mechanisms and their shortcomings that form the foundation of our solution.

1.1 Disruption-tolerant Networking (DTN)

As future space missions will require more complex network topologies to support networked communication, with data being transferred over multiple hops, space agencies acknowledged that custom protocol implementations and manual communication scheduling are not a long-term solution for complex mission scenarios. Hence, the idea of an Internet-like networked communication architecture accommodating space missions emerged [1].

A DTN network can exhibit any combination of the following characteristics*: long delays, discontinuous end-to-end connectivity, high bit error rates and variable data rate*s [2]. Table 1.1 better illustrates the key differences between traditional and challenged networks.

Table 1. Key differences between the assumptions on network conditions for traditional and DTN networks [3]

	Traditional Networks	DTN
End-to-End (E2E)	Continuous	Frequent disconnections
Propagation delay	Short	Long
Transmission reliability	High	Low
Link data rate	Symmetric	Asymmetric

1.2 Bundle Protocol and relevant terminology

BP is now the main communication protocol for DTN in space that standardizes the means of communication between heterogenous networks by forming a store-and-forward overlay network [4]. For a conceptual overview of the components that constitute a BP node, please refer to Figure 1. As an interface between BP and the lower level protocols in the networking stack, Convergence Layer Adapters (CLA) allow the necessary flexibility to the protocol to interconnect networks of different types. A CLA (e.g., UDPCL, TCPCL and LTPCL [5]) provides the means to bind to different convergence layer protocols. The other two key components of a BP node are the **Application Agent (AA)** and the **Bundle Protocol Agent (BPA)**. The purpose of the AA is to utilize the BP services to fulfil a user request. The functionality of this component is divided into two elements:

- **Application Specific Element**: processes application-specific data units by requesting their transmission and accepting their delivery
- **Administrative Element**: "constructs and requests the transmission of administrative records" [6] and processes any records a node receives

The BPA is the entity that offers the services of the protocol as defined in the standard.

A bundle is the PDU the bundle protocol operates with and can consist of two or more blocks. Each block serves different purposes. Each single bundle will have at least the primary and the payload block. The primary block specifies information required to forward a bundle to its destination. Hence, among the fields of the primary block there will be the protocol version, the CRC type, the destination endpoint Id, etc. "Extension Blocks" is the term used for all the other blocks aside from the primary and payload blocks. For instance, a hop count block defines the hop count limit of a bundle. The novel reliability mechanisms we propose requires specifying such a block, the Compressed Reporting Extension Block (CREB), that will be described in detail in Section 2.

Another term relevant to compressed bundle reporting is "administrative record". Administrative records are contained in the payload block of a bundle. This message type provides administrative information to a bundle node, such as reporting on the status of a bundle.

Additional relevant terminology concerning node identification is necessary, especially when describing the implemented mechanism in Section 2. The most recent protocol standard defines the following terms: **bundle node**, **endpoint Id**, **singleton endpoint** and **node Id**.

A **bundle node** is an entity that can send and receive bundles. An **endpoint Id** is described by the following characteristics:

- An endpoint identifies a bundle destination,
- It is a common identifier that describes a set of zero or more bundle nodes. For instance, a sensor network might constitute a set of bundle nodes registered in a single common endpoint and will receive any data delivered to that endpoint,

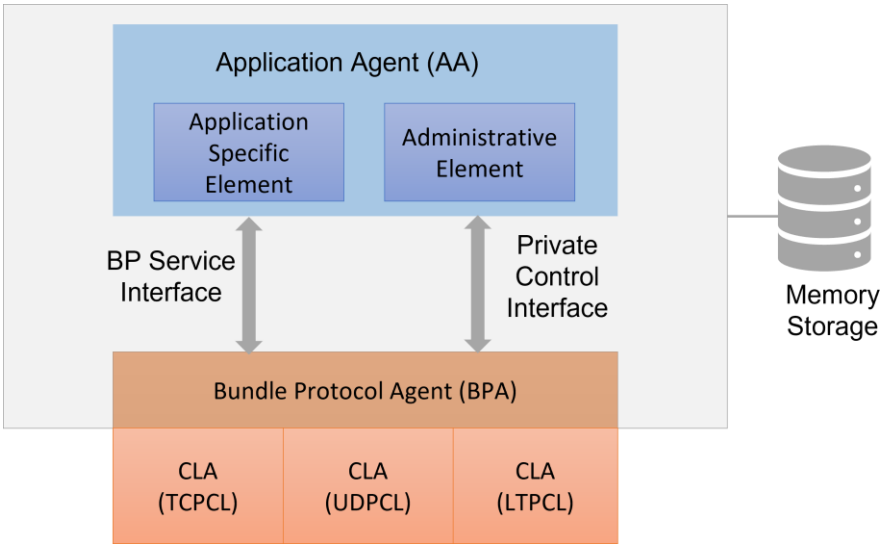

Fig. 1. Bundle Protocol processing components according to CCSDS BP Standard [6]

- A bundle node might be registered in multiple bundle endpoints and receive all data delivered at each of those endpoints.

A **singleton endpoint** is an endpoint that always contains only one member. Such a singleton endpoint can be used as a **node Id** identifying a bundle node.

1.3 Related work

Hop-by-hop reliability can be provided by the underlaying layers in a DTN deployment, but there could be cases when some links along a bundle path do not provide this. Since reliable transmission is often a requirement, it is rather useful to have an additional mechanism implementing this at the bundle layer.

Custody Transfer [7] is one of the first proposed solutions for the issue just described, which translates to a BP node taking responsibility for the further forwarding of a bundle with re-transmission of a bundle if no custody is assumed by another node within some configured time interval. Conditions under which a node may accept custody of a bundle are not formally defined but may include considerations related to available local storage, contact and routing information and the time to live of the bundle. Even so, this mechanism is regarded as a last resort solution and should rarely reissue bundles, since it is too general to work efficiently enough in every possible DTN environment. As an attempt to improve the performance of Custody Transfer, Aggregated Custody Transfer (ACS) [Annex D][6] is an extension of the mechanism that proposes aggregated reports, but still exhibits all the other shortcomings unrelated to that aspect of efficient reporting. Introduced in version 6 of BP [7], Custody Transfer has been removed from the core functionalities of BPv7 due to timer configuration difficulties and fragmentation related issues.

There are other variations of Custody Transfer, such as **BIBE Custody Transfer** [8]. For BIBE-like custody transfer, which operates at the CLA level, although the re-transmission is still based on timers, the claim is that they are easier to set since the specific BIBE destination is known beforehand. However, a BIBE CLA may accept custody and then it is possible that the BPA discards the bundle for various reasons like depleted storage or inability to forward the bundle to a suitable next hop. Also, for a node to be able to take custody of a certain bundle, it needs to be explicitly addressed with a singleton endpoint Id as BIBE destination.

Delay-Tolerant Payload Conditioning (DTPC) [Annex E][6] is a protocol that extends the DTN architecture placed in between the application and the bundle layer. The main scenario that led to the development of this concept is DTN applications transmitting many small bundles [9] and eventually requiring end-to-end bundle acknowledgement. Papastergiou et. al redefine the concept of data aggregation for DTN networks as the concatenation of multiple ADUs into larger data units to be later passed on to the bundle layer. The protocol was designed to provide support for the following end-to-end requirements that BP normally does not guarantee: delivering ADUs in transmission order and not network order, reception gap detection, re-transmission of missing data, data aggregation, and suppression of duplicated ADUs.

In our research, we will consider an aggregate reporting mechanism which can also be used as a basis for reliability mechanisms. The current BPv7 standard already specifies a non-aggregated reporting mechanism, **Bundle Status Reports**. The source of the bundle determines what type of reporting is required: reception, delivery, forwarding, or deletion. However, the generation of status reports should typically be disabled and not used for operational purposes since they only allow reporting per individual bundles, which can easily result in a traffic surge.

The structure of this paper is as following: Section 2 discusses in detail the solution we propose, Section 3 briefly covers the experiments we conducted to validate our approach, followed by Section 4 that analyses the results. We conduct a qualitative analysis of the previously existing mechanisms and our solution in Section 5. Finally, Section 6 concludes the paper and summarizes our results.

2. Compressed Bundle Reporting (CBR): A solution for reliable data transmission at BP layer

Firstly, this section will provide a general overview of the proposed mechanism, then introduce the key concepts it relies on. Secondly, it will specify the expected behaviour of the mechanism such as the process of report requesting and how it reflects in the implementation. Finally, a discussion on a previously existing mechanism for reliable bundle transmission, custody transfer, will be provided and how CBR is flexible enough to integrate the idea of custody transfer in its functionality.

CBR is a generic BP extension designed to achieve efficient bundle status reporting as it allows to report the status of a whole set of bundles in an efficient manner and serve as a reliable bundle transmission mechanism. The key concepts our solution relies on are Flow Id, Flow Sequence Number (FSN), Compressed Reporting Extension Block (CREB), and Compressed Reporting Signal (CRS).

One scenario that could benefit from deploying the CBR extension on top of BP is the case of Earth Observation missions that implement a high-rate downlink. Ideally, we would run BP with CBR on both the spacecraft and the ground station to allow for efficient reporting of the received data on ground to free up resources on-board the spacecraft.

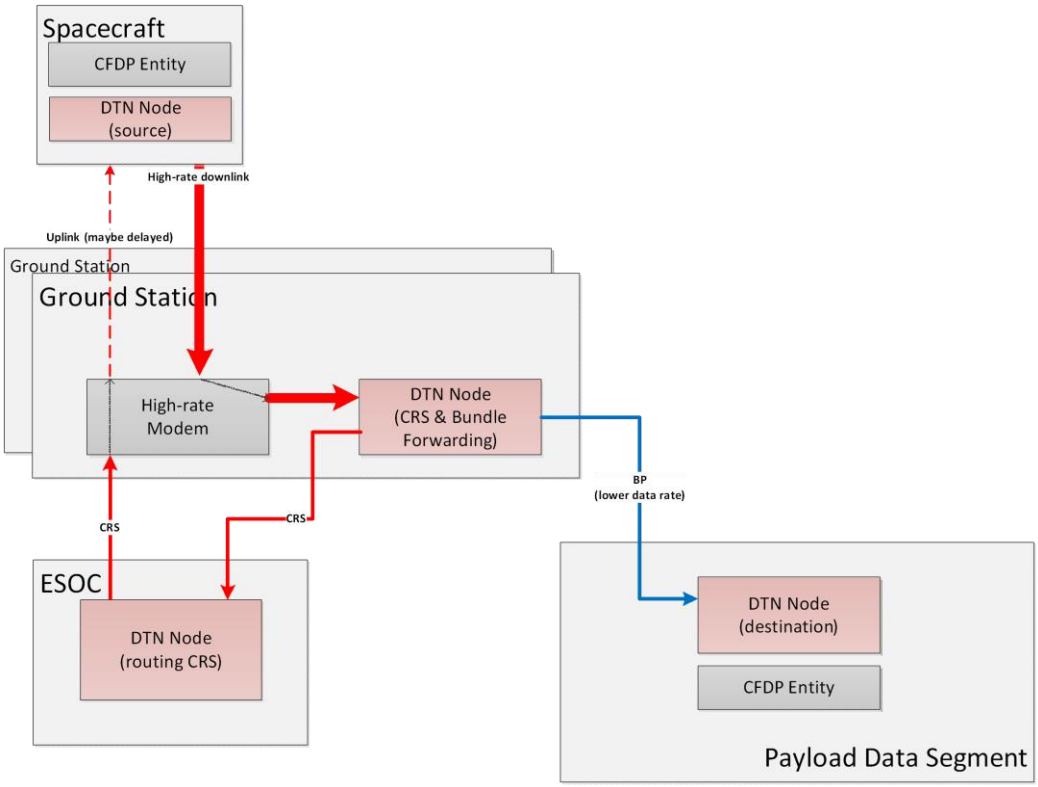

Fig. 2. Application scenario for Bundle Protocol and CBR [10]

2.1 Flow Id and Flow Sequence Number (FSN)

The need to uniquely identify bundles given a certain time frame arose from the desire to send aggregated reports. Moreover, we wanted these reports to take up as little bandwidth as possible, hence we opted for using a combination of incrementing numbers as part of the identifier instead of the source endpoint Id and a timestamp, which would ensure unicity, but fails to facilitate efficient reporting. A bundle is uniquely identified by the combination of a scope node Id, Flow Id and FSN. A Flow Id is a number and could be used to differentiate between different types of data, such as scientific measurements originating from different instruments. A FSN is a number associated with a Flow Id that increases in increments of 1.

2.2 Compressed Reporting Extension Block (CREB)

When an application requests CBR, a new type of "extension block" will be created and inserted among the bundle blocks which will convey the necessary information to uniquely identify a bundle and specify the requested type of reporting. Figure 3 depicts a potential bundle structure with the blocks it could contain. The structure of the CREB is a definite-length Concise Binary Object Representation (CBOR) [11] array that consists of the following fields:

- *Scope node Id*: the node that inserted this block and thus requested the reporting (usually, but not necessarily the source of the bundle),
- Flow Id,
- FSN,
- *Status Report Requests Flags*: used to specify what type of report it is expected (reception, forwarding, delivery, deletion, custody),
- *Report-to node Id*: optional, if different form the Scope node Id.

Fig. 3. Bundle Blocks with CREB

2.3 Compressed Reporting Signal (CRS)

A BP node receiving a CREB may generate a CRS in the form of an Administrative Record which as requested in the CREB. A CRS consists of five definite-length CBOR arrays, each one corresponding to the five types of possible reports than can be requested: reception, delivery, forwarding, deletion, and custody. Each individual report is composed of ranges of FSNs corresponding to a Flow Id. Subsequently, each range contains two elements: the former represents the first FSN for that specific flow, while the latter represents the length of the sequence.

A CRS would be generated based on two node-specific parameters: the number of bundles to report on and a timer. If the number of bundles contained in the report exceeds a certain threshold the report will be sent out, otherwise this event will be triggered by a timer expiration.

An example of a CRS as logged by our prototype implementation is displayed below. The following output reports that bundles with FSNs 0, 1, 2, and 4 have been successfully received, while bundle 3, as the gap in FSN ranges suggests, has been lost.

```
[03:02:39 PM] INFO: Received a CRS from ipn:2.0
CompressedReportingSignal [
      Reception Reports = [  ]
      Delivery Reports = [ [ Flow Id 0:[ [0, 3] [4, 1] ] ] ]
      Forwarding Reports = [  ]
      Deletion Reports = [  ] ]
```

2.4 Behaviour

Bundle nodes may support CBR as bundle source node and implement Report Requesting (2.4.1). Bundle nodes supporting CBR as intermediate or destination node shall implement Compressed Reporting Signal (CRS) Generation (2.4.2). Optionally, sending or intermediate nodes may support re-transmission mechanisms for unreported bundles as described in Bundle Re-transmission (2.4.4).

2.4.1 Report requesting

A CBR request is initiated by the application. Therefore, the application must specify the type of reporting it requires and the flow Id the Application Data Unit (ADU) belongs to. How the application will decide the flow Id of an ADU is an implementation matter. Once a flow Id has been assigned, the following steps need to be integrated in the bundle dispatch procedure specified in the protocol standard [4]. Once a flow Id has been assigned, the following steps need to be integrated in the bundle dispatch procedure specified in the protocol standard:

1. Generate a FSN for that specific bundle,
2. Generate a CREB extension block which will contain the necessary information to uniquely identify the bundle as defined in subsection 2.1,
3. Add a new type of retention constraint to the bundle, CBR_PENDING, so the bundle does not get deleted once forwarded,
4. Save the bundle key store associated with the FSN in a map.

2.4.2 Compressed Reporting Signal (CRS) generation

Since a CRS is a new type of Administrative Record, the BPA will delegate the responsibility of generating and managing CRSs to the Administrative Element of the bundle node once the CREB extension block has been processed.

If the BP node is CBR-enabled, the Administrative Element will construct the report and request its transmission as described by the following procedure:

1. The bundle either triggers the creation of a new CRS or the corresponding FSN is added to a pending CRS,
2. A CRS will transition from a pending to a “ready to be transmitted” state according to two configuration parameters: a configurable number of bundles to report on has been reached or a timer expired.

A bundle node processing a bundle containing a CREB block can take different actions depending on a configurable local policy. It is expected that for specific mission scenarios such policies are defined according to an overall network policy to support the specific use case. For instance, a node could be configured to handle CBR by following one of the policies:

- Ignore the block,
- Send an aggregated report, then delete the block,
- Send an aggregated report, and keep the block,
- Send an aggregated report, then replace some fields of the block.

Regarding the first possible policy, the block processing control flags of the CREB are set so that a node not implementing CBR will simply ignore the block.

2.4.3 Compressed Reporting Signal (CRS) processing

The handling of CRS is an implementation matter. Implementations may just record CRS for monitoring reasons or may implement one of the reliability mechanisms proposed below.

2.4.4 Bundle re-transmission

Unlike Custody Transfer, which provides only timer-based re-transmission, CBR can support additional re-transmission options:

- **Sequence-based re-transmission** *(sender-initiated, reactive)*: the sender can immediately re-transmit the bundles identified as missing once a CRS has been processed. This strategy assumes in-sequence delivery

of bundles, which typically can only be assumed in some tightly controlled mission scenarios. However, there could also be variations of this strategies which allow some re-ordering of bundles in small timer intervals.

- **Timer-based** *(sender-initiated, proactive*): the sender keeps timers for (a set of) bundles and re-transmits a bundle if the timer associated with it expired and no CRS has been received,
- **Command-based re-transmission** *(receiver-initiated, reactive)*: an explicit command is sent to the BP node that will trigger the re-transmission of all the bundles for which reporting was requested but have been missing from the CRSs processed so far at the sender. We envision this strategy to be useful scenarios, such as a spacecraft being signalled to re-transmit everything by mission control before issuing any other CRS. Furthermore, this command could be extended in terms of complexity to send for instance all non-confirmed bundles before a certain time or FSN.

As a result, we have the possibility use different re-transmission mechanism based on the specific mission scenario. Some of them do not require accurate estimations of RTTs or to know which node will issue a response to our custody transfer request as BIBE does. However, the chose of a suitable reliability mechanism needs to be based on network knowledge and needs to be supported by policy. This may only be possible in limited, controlled scenarios and might not work in a large and open network unless either specific policies are enforced or network management and suitable routing and forwarding solutions are available and widely deployed.

2.4.5 Custody Transfer Discussion

The modularity of the CBR design allows us to easily extend it to support a custody transfer-like mechanism on top of it. The CREB allows a sending node to request custody by setting the “custody request” flag. A BP node deciding to accept custody for a bundle will take a set of actions like in BPv6 custody acceptance. First, if the node accepts custody, it will store the bundle. Then it will construct an (aggregated) administrative record to report custody acceptance to the previous custodian node specified in the CREB. Afterwards it will either replace the custodian field in the CREB with its own identifier or remove the CREB and replace it with one newly generated containing requesting a custody signal to itself. Furthermore, one crucial requirement for a node that is accepting custody is the implementation of one of the re-transmission strategies described in subsection 2.4.4. However, regardless of the possibility of using a different re-transmission strategy that could alleviate the timer configuration related issues of custody transfer, we still face the difficulties related to the custody acceptance decision that is based on the storage space of a node plus routing or contact information to ensure a high probability that the accepted bundle will reach its destination.

3 Experiments

For validation of CBR with a re-transmission strategy, we have implemented it within the ESA BP implementation. To evaluate the benefits and performance of our design, we ran four experiments, with three runs for each experiment to ensure the consistency of our results. The re-transmission strategy that was selected for the experiments is sequence-based re-transmission. Firstly, we chose two types of link conditions, a lossy and a more reliable link (5% and 1% bundle loss, respectively), then we configured the mechanism to send reports at two different frequencies: first every 5 seconds, and then every 10 seconds. The motivation for choosing these values for packet loss is that for higher values, we would consider increasing the reliability at the lower layers in the networking stack. To simulate the packet loss in bursts, we used a simple two state Markov model, with a loss and a no-loss state, that has been integrated as well with ESA’s BP implementation.

Figure 4 illustrates the network topology we used to validate our CBR implementation.

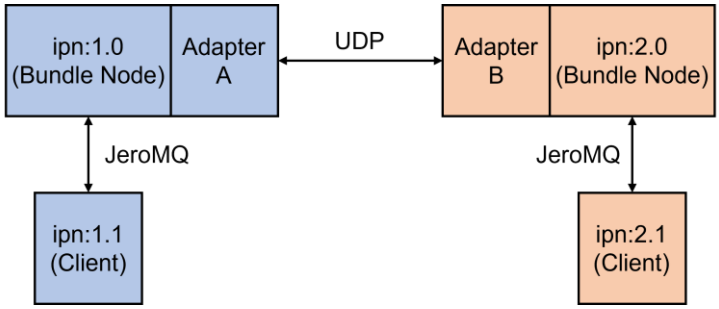

Fig. 4. Experiments network topology

Since the client ipn:1.1 is requesting the reporting, the node will check if there are any CRSs to be processed every 20 seconds. As BN ipn:2.0 is the node that will report back, we configured it with two sets of values. The first configuration involves considering a CRS to be ready to be sent once it either reports either on 50 bundles or 5 seconds have elapsed since the CRS has been created. The second configuration uses 200 as the limit of reported bundles the CRS can contain and 10 seconds as a timeout. As the CRSs ready to be sent accumulate in a queue, we release them in batches every 5 seconds.

4 Results

Figures 5 validates the behaviour of the mechanism during a communication session over a lossy link (5% overall drop rate).

First, we can observe the distribution of drop events while sending the data and simultaneously the re-transmission events resulted from the processing of a batch of CRSs. We chose a sample rate of 20 seconds to illustrate the results since it matches the rate at which the BN ipn:1.1 is processing CRSs queuing up during the experiments. We display only the lost and re-transmitted bundles since plotting the number of sent bundles at the same time would offset the scale of the graph. The two categories of data are stacked, with the retransmitted bundles having as starting point the peak of the lost bundles.

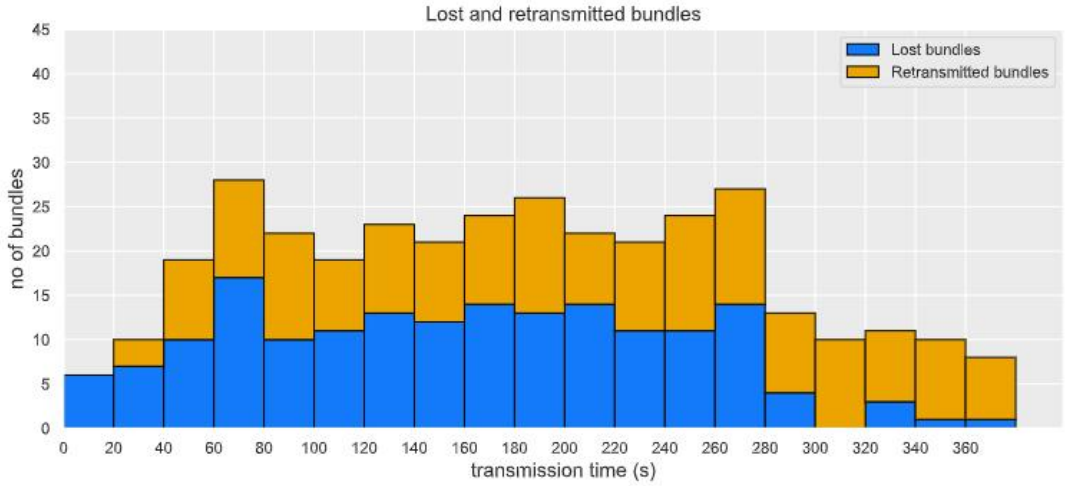

Fig. 5. Experiment 1: 5% overall drop rate, CRS Timeout = 5s, CRS No of Reported Bundles Threshold = 50

Figure 6 illustrates how the mechanism runs over a reliable link with very few lost bundles. One can observe the CRSs arriving at regular intervals and 3 re-transmission events. Since the loss events are so rare, one could send CRSs less frequent.

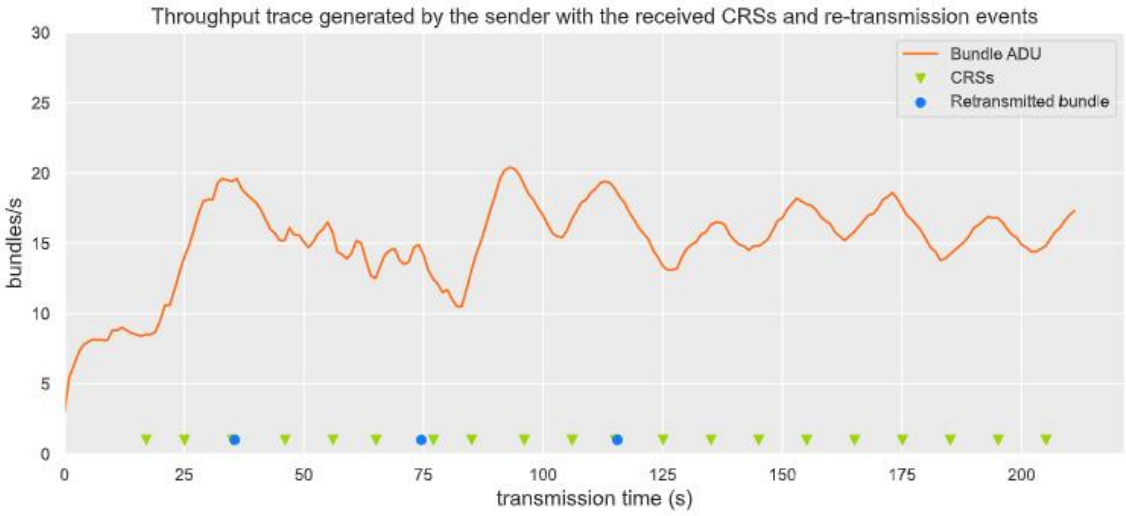

Fig. 6. Experiment 4: 1% overall drop rate, CRS Timeout = 10s, CRS No of Reported Bundles Threshold = 200

To illustrate the performance enhancements that result from aggregated reporting we computed the number of bytes used by a reported bundle in a CRS by dividing the size of the ADU containing the CRS by the number of bundles it is reporting on. Naturally, for both types of links (lossy and reliable), when we allow for more bundles to accumulate in a report (200 as the number of bundles the report needs to contain before it is ready to be sent), there would be less bytes required per bundle. Figure 7 displays how the number of bytes required per bundle varies during the first two experiments with the lossy link. The circle marker in the figure represents the reception of a CRS. Despite having the same bundle drop distribution, the report allowing for the aggregation of more bundles compresses better, never going over 1 bytes/bundle. As for the first experiment, the values oscillate between 0.5 and 1 bytes/bundle for most of the reports, with occasional bursts of around 2 bytes/bundle and reaching a peek at 7 bytes/bundle at the end of the experiment since it is the end of the communication, and the report contains considerably less bundles.

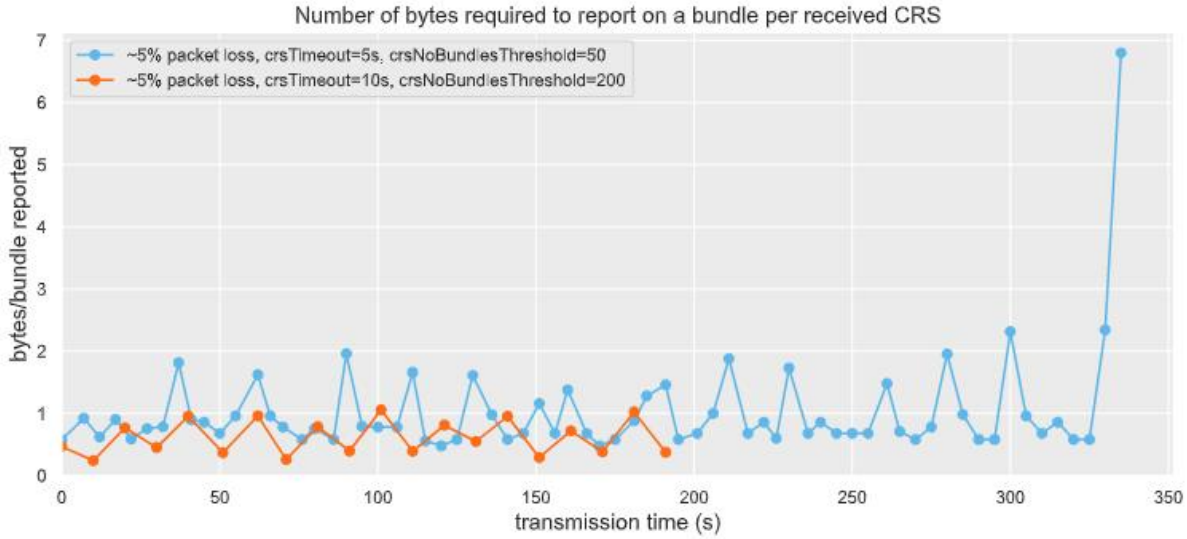

Fig. 7. Experiments 1 and 2

The experiments run over the more reliable link (see Figure 8), exhibiting a drop rate of around 1%, achieved impressive results, with the number of bytes per reported bundle fluctuating between around 0.45 and 0.7 and then stabilizing at 0.5 bytes/bundle during most of experiment 3. This trend is due to the very few losses occurring at this

drop rate. As was anticipated, the best results were recorded during experiment 4, with values shifting between a minimum of approximately 0.1 bytes/bundle and 0.2 usually, reaching a maximum of approximately 0.3 at the beginning of the experiment attributable to not having enough bundles to report on to maximise compression.

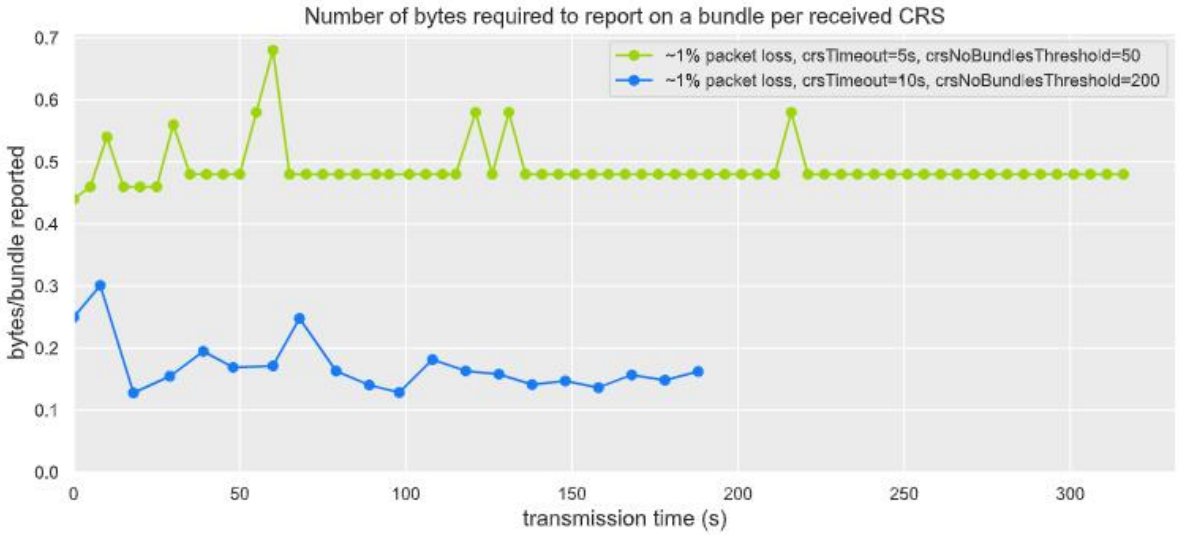

Fig. 8. Experiments 3 and 4

To illustrate the benefits of sending aggregated reports, we are comparing the size of a CRS to the size of a bundle status report that serves as our baseline reporting mechanism. The size of the ADU of a bundle status report is around 27 bytes. Comparing our experiment results to this value, we managed to reduce the number of bytes required to report on a bundle by a factor of 10 in the case of the first experiment (5% drop rate and considering a CRS to be ready to be sent once it either reports on 50 bundles or 5 seconds have elapsed since the CRS has been created) and by a factor of 100 in the best case scenario, which is experiment 4 (1% drop rate and considering a CRS to be ready to be sent once it either reports on 200 bundles or 10 seconds have elapsed since the CRS has been created).

5 Discussion

This section will first introduce the criteria based on which we conducted the qualitative analysis of our solution and previously existing mechanisms discussed briefly in subsection 1.3 and then all the mechanisms will be compared against each other considering each criteria individually.

The qualitative study of reliable data transmission mechanisms for BP is conducted considering five dimensions:

1. **Aggregated bundle reporting:** Reports are sent back containing information on the status of multiple bundles to reduce the header overhead since we have a smaller number of Administrative Records transmitted, thus improving network performance,
2. **Hop-based vs. E2E re-transmission:** Although E2E re-transmission is the de facto standard of operation in the Internet today, given the volatile nature of DTN links due to their temporal dimension, a hop-based approach is the obvious choice for space communication in most of the cases,
3. **Support for communication flow:** Separating the data into flows would allow a node to better manage its resources and prioritize traffic,
   Support for sending data without the need to address a specific next hop: Needing to address a specific next hop is a BIBE feature that limits the mechanism and makes it more rigid, preventing opportunistic communication. By removing this constraint introduced by BIBE, we enable the possibility of every node along the path to report.

Table 3 gives an overview of what features comes with each mechanism and summarizes our discussion.

Table 3. Features of the mechanisms

Mechanism	Aggregated reporting	Flow	Opportunistic reporting	Re-transmission	
				Hop-based	E2E
Bundle Status Reporting	no	no	yes	no	no
Custody Transfer	no	no	yes	yes	no
ACS	yes	no	yes	yes	no
BIBE	yes	no	no	yes	no
DTPC	yes	yes	no	no	yes
CBR	yes	yes	yes	yes	yes

Aggregated reporting is provided by four out of six of the analysed mechanisms, three previously existing proposals, ACS and BIBE (that incorporated ACS in its functionality), DTPC, plus our solution, CBR. However, both ACS, and consequently BIBE, come with some disadvantages. First, they are built on top of custody transfer, so they cannot provide just reporting. As for DTPC, it allows only E2E reporting, since it was designed to provide E2E services.

The only other mechanism that proposes communication flows aside from CBR is DTPC, which is an E2E mechanism. We find this feature highly relevant because we can separate between different streams of data, and eventually implement a prioritization policy on top of it. Certainly, support for flows could be introduced by other means then CBR. Still, the combination with a reporting mechanisms could be an attractive solution.

Opportunistic reporting is a key mechanism feature if we envision for scalable and autonomous networks in space. All the mechanisms analysed implement this behaviour with one exception, BIBE. BIBE requires knowing the specific endpoint Id of the next hop, which prevents any unplanned communication. This is one of the major drawbacks of BIBE when proposed as a reliable data transmission solution, even though it implements aggregated reports. DTCP comes with the limitation previously mentioned: only the bundle destination can produce a report.

Aside from Bundle Status Reports, the other mechanisms offer the possibility of re-transmitting lost bundles. We can analyse this dimension from two points of view: *how is the re-transmission triggered* and *the hop-by-hop versus the E2E approach*. All the mechanisms offer timer-based re-transmission, which roughly means that in the absence of a positive acknowledgement, the bundle will be resent when a timer expires. However, setting accurate values for the timers is the most challenging aspect of this approach. To mitigate the issues associated with timer configurations (redundant or too late re-transmissions), CBR is introducing enabling other re-transmission strategies such as discussed in subsection 2.4.4: sequence-based and command-based re-transmission. Concerning the hop-by-hop versus E2E approach, only DTPC and CBR (when delivery reports are requested) are capable of purely E2E re-transmission.

In terms of the overhead of communication generated, we can establish the following ranking. The mechanisms which report on individual bundles will intuitively perform the worst in terms of bandwidth utilization. Consequently, the latency will increase in case of lost bundles since their re-transmission will take longer. On the other hand, aggregated reporting will allow for the timely acknowledgment of bundles, less communication overhead due to the way reports are constructed, and a more efficient utilization of the bandwidth. As for CBR versus ACS and implicitly BIBE, it would be rather hard to make a comparison since the former provides the option to select multiple types of reporting, while the latter supports only custody transfer reports.

6 Conclusions and future work

Compressed Bundle Reporting (CBR) as defined in this paper provides not only efficient bundle reporting, but also enables re-transmission mechanisms at the bundle layer. In the end, we developed a highly modular mechanism proposal that covers all the gaps we identified in previous implementations by providing support for aggregated reports, communication flows, opportunistic reporting, and different re-transmission strategies.

Through the four experiment scenarios we ran, we validated the behaviour of CBR and proved the performance of aggregated reports by studying the number of bytes required by one bundle reported in a CRS.

Finally, we have analysed CBR in contrast to previously existing solutions across four dimensions and explained how CBR tackles the past challenges.

As future work is concerned, we would like to:

- Implement and test the other two re-transmission strategies: timer-based and command-based re-transmission,
- Study the mechanism in a more complex network topology, such as a lunar network,
- Implement custody transfer on top of CBR.

- Release CBR as an Experimental Specification under CCSDS

In conclusion, we regard CBR as a viable basis for reliable data transmission in space communication scenarios due to its simple and flexible design that could easily be extended with new functionalities in the future, should the necessity arise.

Acknowledgements

I want to thank my mentor at ESA, Felix Flentge, for his guidance in the field of DTN and my thesis coordinator, Stefan Schmid and his team, Max Frank and Arash Pourdamghani, for their support and extensive personal and professional guidance. This work would not have been possible without them.

References

[1] "Solar System Internetwork (SSI) Architecture," *Sol. Syst.*, 2014.

[2] L. Torgerson *et al.*, "Delay-Tolerant Networking Architecture," Internet Engineering Task Force, Request for Comments RFC 4838, Apr. 2007. doi: 10.17487/RFC4838.

[3] J. J. P. C. Rodrigues and V. N. G. J. Soares, "1 - An introduction to delay and disruption-tolerant networks (DTNs)," in *Advances in Delay-Tolerant Networks (DTNs)*, J. J. P. C. Rodrigues, Ed. Oxford: Woodhead Publishing, 2015, pp. 1–21. doi: 10.1533/9780857098467.1.

[4] S. Burleigh, K. Fall, and E. J. Birrane, "Bundle Protocol Version 7," Internet Engineering Task Force, Request for Comments RFC 9171, Jan. 2022. doi: 10.17487/RFC9171.

[5] R. Wang, S. C. Burleigh, P. Parikh, C.-J. Lin, and B. Sun, "Licklider Transmission Protocol (LTP)-Based DTN for Cislunar Communications," *IEEEACM Trans. Netw.*, vol. 19, no. 2, pp. 359–368, Apr. 2011, doi: 10.1109/TNET.2010.2060733.

[6] "CCSDS Bundle Protocol Specification," vol. 734.2, no. B1, p. 94, 2015.

[7] K. Scott and S. C. Burleigh, "Bundle Protocol Specification," Internet Engineering Task Force, Request for Comments RFC 5050, Nov. 2007. doi: 10.17487/RFC5050.

[8] S. Burleigh, "Bundle In Bundle Encapsulation (BIBE)," p. 21, 2022.

[9] G. Papastergiou, I. Alexiadis, S. Burleigh, and V. Tsaoussidis, "Delay Tolerant Payload Conditioning protocol," *Comput. Netw.*, vol. 59, pp. 244–263, Feb. 2014, doi: 10.1016/j.bjp.2013.11.003.

[10] D. F. Flentge, "Towards Implementation of Delay-Tolerant Networking in the ESA Ground Segment," *16th Int. Conf. Space Oper.*, May 2021, [Online]. Available: https://spaceops.iafastro.directory/a/proceedings/SpaceOps-2021/SpaceOps-2021/8/manuscripts/SpaceOps-2021,8,x1300.pdf

[11] C. Bormann and P. E. Hoffman, "Concise Binary Object Representation (CBOR)," Internet Engineering Task Force, Request for Comments RFC 8949, Dec. 2020. doi: 10.17487/RFC8949.