

Network Working Group
Internet-Draft
Expires: February 15, 2008

W. Eddy
Verizon
L. Wood
Cisco Systems
August 14, 2007

The DTN Bundle Protocol Payload Checksum Block
draft-eddy-dtnrg-checksum-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on February 15, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Internet-Draft

Bundle Protocol Checksum

August 2007

Abstract

The Delay-Tolerant-Networking Bundle Protocol includes a custody transfer mechanism to provide acknowledgements of receipt for particular bundles. No checksum is included in the basic DTN Bundle Protocol, however, so it is not possible to verify that bundles have been either forwarded or passed through various convergence layers without errors having been introduced. This document partially rectifies the situation by defining a Bundle Protocol extension for carrying payload checksums and describes how its use can be integrated with both custody transfer and fragmentation.

Table of Contents

1.	Motivations	3
2.	Payload Checksum Block Format	5
2.1.	Checksum Algorithms	6
3.	Rules for Use	8
4.	Security Considerations	9
5.	IANA Considerations	10
6.	Acknowledgements	11
7.	References	12
7.1.	Normative References	12
7.2.	Informative References	12
	Authors' Addresses	13
	Intellectual Property and Copyright Statements	14

1. Motivations

Reliable transmission of information is a well-known problem for all protocol layers. Error detection and correction capabilities are found in lower layers, but are also present in many higher-layer protocols in order to detect residual bit errors and bugs. For example, IPv4 verifies a simple header checksum before processing inbound packets, even when running over a data link like Ethernet that already performs a stronger CRC, and TCP and UDP segments further includes a checksum covering their contents as well as some IP header fields. What may seem like paranoia is actually not unfounded, as errors in received data or packet corruption are known to creep into packets from causes other than channel noise [[SP00](#)]. Although coding of data on the channel can reduce the impact of channel noise, end-to-end checksums are understood to be necessary for applications requiring relative certainty that the data received is error-free [[SG98](#)].

The Delay/Disruption-Tolerant Networking (DTN) architecture [[RFC4838](#)] is founded on an overlay of Bundle Agents (BAs), that forward data units called bundles via a Bundle Protocol [[SB07](#)]. Bundles may be lost or errored both during transmission between BAs, or within a BA itself. Bundles belonging to applications that are not tolerant of lost data have a "custody transfer" flag that requests reliable transmission between bundle agents. Unfortunately, the notion of reliability used in the basic custody transfer mechanism does not take bit errors into account. It is assumed that the "convergence layer adapters" that connect BAs to each other will detect and correct errors before presenting bundle data to the BAs themselves. This may be adequate in many cases, but is not always sufficient, and the insufficiency is encapsulated in the well-known end-to-end principle [[RFC4838](#)]. It is possible (and even statistically likely) that either transmission errors will go unnoticed, or unchecked errors will be introduced within a BA's memory, storage, or forwarding systems.

For example, the UDP convergence-layer adapter that has been popularly implemented in DTN stacks uses the usual IP 16-bit one's-complement checksum to validate incoming packets. This checksum is computed by summing 16-bit values within a packet. If two strings within the packet of size greater than 16 bits are swapped in position, the checksum will pass even though the datagram is now different from the original, and clearly errored. The proposed TCP-based convergence layer relies on the same checksum algorithm. This checksum algorithm is considered weak, in that it does not detect a class of subtle errors, but at least an attempt to verify that the packet was as sent has been made.

Even stronger convergence-layer adapter error detection is not sufficient. Errors within a BA's memory, errors due to memory issues within the BA's host, e.g. radiation-induced soft errors, or errors introduced from file-system corruption cannot be detected by convergence layer adapters, as these errors occur in between successive phases of forwarding and convergence-layer processing. End-to-end computation and verification of checksums is required to ensure integrity of DTN bundles forwarded across a system composed of BAs and convergence layer adapters [[SRC84](#)].

The proposed Bundle Security mechanisms [[SFWP07](#)] are capable of providing an end-to-end checksum, but are intended for the very different problem of security. The current design of Bundle Security is not practical for simple integrity checking outside of a more paranoid security context. For example, the Bundle Security mechanisms include "mandatory ciphersuites" that implementations must support. No simple checksum algorithm is among the mandated algorithms. The mandated ciphersuites do include some more complex keyed-hash constructions, but these rely on key management, which is not appropriate for general integrity checking between multiple parties simply relaying bundles. While it would be technically possible to graft a non-security integrity-checking mechanism onto the available security mechanisms by specifying some assumed key, it would be inappropriate for the problem at hand, and overkill. Instead, it would be much simpler and less error-prone to implement a separate checksum block for optional inclusion on bundles. [Section 2](#) of this document defines such a block and [Section 3](#) gives some simple rules for its use.

Block Type	Block Processing Ctrl Flags (SDNV)
Block Length (SDNV)	
Checksum Algorithm (SDNV)	

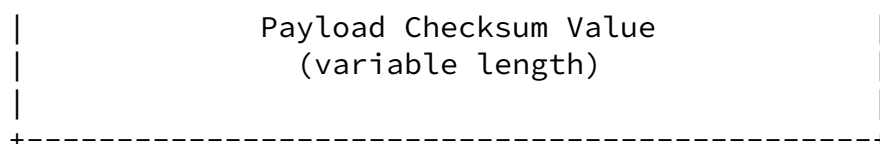


Figure 1: Payload Checksum Block Format

The fields shown in Figure 1 above are defined as:

- o Block Type - Implementations are currently using a value defined as experimental in the Bundle Protocol Specification, but can be expected to transition to an assigned value.
- o Block Processing Ctrl Flags - The bits in this SDNV are defined in the Bundle Protocol Specification. For Payload Checksum Blocks, none of these bits need be set, except perhaps bit 3, to indicate the "last block", when the block is sent as the final block in a bundle.
- o Block Length - This SDNV encodes the length of the remainder of the Payload Checksum Block. When the Checksum Algorithm SDNV is parsed, its length can be subtracted from the Block Length value to determine the level of truncation for the Payload Checksum Value, as explained below.

- o Checksum Algorithm - This identifies the algorithm used to compute the Payload Checksum Value field following it. Defined algorithms are listed below.
- o Payload Checksum Value - This is a raw string of octets whose length is implied by the Block Length. This string contains the checksum result computed over the Payload Block of the bundle, and may only contain the high-order bits of this result, if truncation is used to shorten the length of the checksum, as described below.

2.1. Checksum Algorithms

Any implementation of this specification MUST support the MD5 checksum algorithm [[RFC1321](#)]. MD5 has been chosen as the baseline checksum algorithm in this mechanism because it represents a

reasonable tradeoff between robust error detection and efficiency of implementation. For widespread use in DTNs, both resource-efficient implementation and decent error-detection capabilities are needed. MD5 algorithms are known to achieve processing several Mbps, even on rather limited hardware [[RFC1810](#)], yet MD5 provides a much more robust checksum than the Internet's 16-bit one's complement checksum. Although MD5 has cryptographic weaknesses and is discouraged for use in security protocols, concerns with resistance to pre-image generation are irrelevant here as we are not using MD5 values in a security context.

We also have a defined value to indicate use of SHA-1 checksums [[RFC3174](#)]. However, support for SHA-1 checksums is not required. SHA-1 is significantly less efficient than MD5 to compute in our experience, for seemingly little added error-detection capability when truncated to the same length. Implementations SHOULD at least support receiving and verifying SHA-1 checksums.

An Adler-32 checksum option is also specified, but should be used only in cases where bundle payloads are relatively small and efficiency of computation is highly important. Implementations SHOULD support at least receiving and verifying Adler-32 checksums.

The complete list of currently defined Checksum Algorithm values is:

+---+-----+-----+-----+			
#	Algorithm	Un-Truncated Length	
+---+-----+-----+-----+			
0	MD5	128 bits	
1	SHA-1	160 bits	

2	Adler-32	32 bits	
+---+-----+-----+-----+			

Other checksum algorithms may be assigned values in future documents.

On small payloads, the relatively long output of the MD5 or SHA-1 algorithms might be viewed as a detriment to end-to-end application performance by increasing the header overhead of the Bundle Protocol, and reducing the capacity available to higher layers. For this reason, senders of the Payload Checksum Block are permitted to truncate the transmitted Payload Checksum Values if the full checksum algorithm's output is deemed to be overly long in comparison to the size of the payload. This should be done carefully at the sending application's discretion, and never by default. When generating or verifying a truncated checksum, it is understood that only the high-order octets are included.

The checksum conveyed in the Payload Checksum Block only covers the Payload Block of a bundle, and does not include any pseudoheaders with EIDs, timestamps, etc., or any other portion of a bundle or its other extension blocks. Ensuring robustness of bundle header information and metadata is a separate problem not addressed here; ideally each header would be self-checking to guarantee a degree of robustness on receipt.

The checksum within the Payload Checksum Block has differing semantics if it occurs before or after the Payload Block. If placed before the Payload Block, then the Checksum Value should be understood to cover the entire (unfragmented / reassembled) payload, whereas if it follows the Payload Block within a Bundle Fragment, then the Checksum Value only applies to the included fragment of the payload.

Intermediate Bundle Agents, which may not be affiliated with either the source nor the destination of a bundle, are permitted to verify the Payload Checksum Block and attempt local recovery. If local recovery is not possible, then the bundle MAY be deleted.

This document suggests amending the Bundle Protocol specification with regard to Custody Transfer. Without any checksum verification, claiming custody of a bundle is a potentially troublesome operation. We suggest that the Bundle Protocol specification require the use of a Payload Checksum Block when Custody Transfer is requested by an application in order to close this gap.

[4.](#) Security Considerations

The mechanism described in this document does not introduce any new security concerns beyond those present in the basic Bundle Protocol. It only attempts to ensure the reliability of the Bundle Protocol payload. Ensuring Bundle Protocol header reliability remains an open problem.

[5.](#) IANA Considerations

This document has no considerations for IANA.

[6.](#) Acknowledgements

Some of the work on this document was performed at NASA's Glenn Research Center under funding from the Earth Science Technology Office (ESTO) and the Space Communications Architecture Working Group (SCAWG).

[7.](#) References

[7.1.](#) Normative References

- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", [RFC 1321](#), April 1992.
- [RFC1810] Touch, J., "Report on MD5 Performance", [RFC 1810](#), June 1995.
- [RFC3174] Eastlake, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", [RFC 3174](#), September 2001.
- [SB07] Scott, K. and S. Burleigh, "Bundle Protocol Specification", [draft-irtf-dtnrg-bundle-spec-09](#) (work in progress), April 2007.

[7.2.](#) Informative References

- [RFC4838] Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant Networking Architecture", [RFC 4838](#), April 2007.
- [SFWP07] Symington, S., Farrell, S., Weiss, H., and P. Lovell, "Bundle Security Protocol Specification",

[draft-irtf-dtnrg-bundle-security-03](#) (work in progress),
April 2007.

- [SG98] Stone, J., Greenwald, M., Hughes, J., and C. Partridge,
"Performance of checksums and CRCs over real data", IEEE
Transactions on Networks vol. 6 issue 5, pp. 529-543.
- [SP00] Stone, J. and C. Partridge, "When the CRC and TCP Checksum
Disagree", Proceedings of ACM SIGCOMM 2000,
September 2000.
- [SRC84] Saltzer, J., Reed, D., and D. Clark, "End-to-end Arguments
in System Design", ACM Transactions on Computer Systems 2
(4), November 1984.

Eddy & Wood

Expires February 15, 2008

[Page 12]

Internet-Draft

Bundle Protocol Checksum

August 2007

Authors' Addresses

Wesley M. Eddy
Verizon Federal Network Systems
NASA Glenn Research Center
21000 Brookpark Rd, MS 54-5
Cleveland, OH 44135
United States of America

Phone: +1-216-433-6682
Email: weddy@grc.nasa.gov

Lloyd Wood
Cisco Systems
11 New Square Park, Bedfont Lakes
Feltham, Middlesex TW14 8HA
United Kingdom

Phone: +44-20-8824-4236
Email: lwood@cisco.com

Eddy & Wood	Expires February 15, 2008	[Page 13]
-------------	---------------------------	-----------

Internet-Draft	Bundle Protocol Checksum	August 2007
----------------	--------------------------	-------------

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND

THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).