Standards Change Request

File Checksums **SCR3-1034.v7**
Elizabeth D. Rye, Dick Simpson **May 16, 2006**

Provenance:

Date 2006-05-15, revision 6.0
Working Group: E. Rye
Title: File Checksums (SCR3-1034.v6)

Date: 2006-03-22, revision 5.0
Working Group: E. Rye (lead), T. King, M. McAuley
Title: File Checksums (SCR3-1034.v5)

Date: 2006-03-20, revision 4.0
Working Group: E. Rye (lead), T. King, M. McAuley
Title: File Checksums (SCR3-1034.v4)

Date: 2006-03-06, revision 3.0
Working Group: E. Rye (lead), T. King, M. McAuley
Title: File Checksums (SCR3-1034.v3)

Date: 2005-11-14, revision 2.0
Working Group: T. King (lead), M. McAuley
Title: MD5 Checksums (SCR3-1034.v2)

Date: 2004-11-22, revision 1.0
Working Group: J. Wilf (lead), T. King, M. McAuley
Title: MD5 Checksums for Files (SCR3-1034.v1)

Problem:

As an entity responsible for maintaining data, PDS must be able to ascertain the integrity of its archive. This includes (PDS requirements):

1. verifying the integrity of data stored on physical media (4.1.2),
2. detecting errors introduced during transfer of data to newer media (4.1.3) ,
3. detecting errors that occur during the transmission of data, such as from data providers to the PDS, between PDS nodes, from the PDS to the NSSDC, and from the PDS to end users (3.2.3).

At present, there are no generally accepted methods within PDS for achieving any of these objectives.

Proposed Solution:

A simple method for detecting errors is to create and maintain a list of checksum values for every file of interest. The checksum values can be confirmed periodically for static PDS holdings or at the completion of a transfer.

This SCR outlines an optional procedure whereby such checksum lists can be included in a prescribed manner within a volume of data. "Volume" is taken to be the structure defined in Chapter 19 of the *PDS Standards Reference* and may be either physical or logical, including ad hoc volumes created solely for the purpose of transferring data.

Inclusion of a checksum list within a volume has the following advantages (1-3) and disadvantages (4-6):

1. The checksum list is conveniently packaged with the data files of interest in a known location and in a known format;
2. No communication with an outside data source is needed in order to confirm the validity of the data files;
3. The checksum list can be created at the same time as the volume and accompanies it for the lifetime of the volume;
4. The checksum list itself cannot be verified;
5. The single list may not be convenient if the volume is augmented or subdivided later;
6. This solution is not helpful if "volume" is not the relevant structure,

The proposal outlined here, while permitting only the MD5 checksum type and recognizing the disadvantages above, may be amended in the future based on needs and experience.

The changes proposed in this SCR are to:

1. Establish an optional, reserved file, "CHECKSUM.TAB", which contains checksum values for all files in a volume, to be included in the INDEX directory.
2. Create a new keyword, CHECKSUM_TYPE, for use in the CHECKSUM.LBL file, to allow for the future use of different types of checksums, should they become permitted.
3. Update the element definition for the MD5_CHECKSUM keyword, changing the STATUS_TYPE to APPROVED.

Requested Changes:

Changes to the Standards Reference

The following changes to the PDS Standards Reference are required to support this SCR:

Add to section 10.2.2 Reserved File Names, "CHECKSUM.TAB".

Add to Chapter 19.3.2.3 INDEX Subdirectory, after INDEX.TAB:

CHECKSUM.LBL **Optional**

This is the PDS label for the CHECKSUM.TAB file. The object definition for the CHECKSUM column must contain the CHECKSUM_TYPE keyword.

CHECKSUM.TAB **Optional**

This file contains a checksum for every file on the volume except itself and its label. The file is a PDS ASCII TABLE object with two columns, one (named CHECKSUM) providing the checksum values and another (named FILE_SPECIFICATION_NAME) containing the path and name of each file in the archive relative to the root directory of the volume.

For examples of the CHECKSUM.TAB and CHECKSUM.LBL files, see Appendix D, section D.2.

Each figure in Chapter 19. Volume Organization and Naming, will need to be updated to include the (optional) “CHECKSUM.TAB” and "CHECKSUM.LBL" files in the INDEX directory.

Appendix D, section D.2 add the sample CHECKSUM.TAB and CHECKSUM.LBL files as shown in the attachment. (The following sections of Appendix D will need to be re-numbered.)

The title of Appendix D must be changed to "Appendix D. Examples of Required and Selected Optional Files".

Changes to the Data Dictionary

Modify the description of the MD5_CHECKSUM keyword as shown in the attached element definition template.

Add the new keyword, CHECKSUM_TYPE, as shown in the attached element definition template.

Changes to the PDS Tool Suite

A number of utilities are already widely available which can be used to produce and read the CHECKSUM.TAB file. But any tool which validates PDS volumes (*e.g.*, the volume verifier) will need to be modified to determine whether CHECKSUM.TAB is present and to carry out the checksum verifications.

Impact Assessment:

There is no impact beyond the above described changes that is mandated by this SCR. If nodes should choose to implement checksums as a method for validating the integrity of their archives, they will need, at some point, to go back and generate checksum values for all of their data holdings that were created without checksums. A rough estimate of the time to accomplish this task is about 8 hours of labor per node. Further effort (perhaps 4 hours) would be required to generate a script to validate the integrity of existing archive holdings periodically using the checksum values previously generated.

At some point in the future (again, not mandated by this SCR), a mechanism associated with the product servers may be generated on a best efforts basis to provide checksum values to users who download individual PDS data product files.

```
PDS_VERSION_ID                    = PDS3
LABEL_REVISION_NOTE               = "2004-04-06, CN: BAM;
2004-10-14, PPI: S. Joy; 2006-05-15, EN: EDR"

OBJECT                            = ELEMENT_DEFINITION
  ELEMENT_NAME                    = "md5_checksum"
  BL_NAME                         = "md5checksum"
  DESCRIPTION                     = "

   The MD5 algorithm takes as input a file (message) of arbitrary length
   and produces as output a 128-bit 'fingerprint' or 'message digest' of
   the input. It is conjectured that it is computationally infeasible to
   produce two messages having the same message digest, or to produce
   any message having a given prespecified target message digest.

   Most standard MD5 checksum calculators return a 32 character
   hexadecimal value containing lower case letters.  In order to accommodate
   this existing standard, the PDS requires that the value assigned to the
   MD5_CHECKSUM keyword be a value composed of lowercase letters (a-f) and
   numbers (0-9).  In order to comply with other standards relating to the
   use of lowercase letters in strings, the value must be quoted using double
   quotes.

   Example:  MD5_CHECKSUM = "0ff0a5dd0f3ea4e104b0eae98c87f36c"

   The MD5 algorithm was described by its inventor, Ron Rivest of
   RSA Data Security, Inc., in an Internet Request For Comments
   document, RFC1321 (document available from the PDS).

   References
   ==========
   Rivest, R., The MD5 Message-Digest Algorithm, RFC 1321, MIT Laboratory for
   Computer Science and RSA Data Security, Inc., April 1992."

  GENERAL_DATA_TYPE               = "CHARACTER"
  MAXIMUM                         = ""
  MINIMUM                         = ""
  MAXIMUM_LENGTH                  = "32"
  MINIMUM_LENGTH                  = "32"
  STANDARD_VALUE_TYPE             = "DEFINITION"
  STANDARD_VALUE_SET_DESC         = "N/A"
  KEYWORD_DEFAULT_VALUE           = "N/A"
  UNIT_ID                         = "NONE"
  SOURCE_NAME                     = "PDS CN/B. SWORD"
  FORMATION_RULE_DESC             = "N/A"
  SYSTEM_CLASSIFICATION_ID        = "COMMON"
  GENERAL_CLASSIFICATION_TYPE     = "N/A"
  CHANGE_DATE                     = "2006-03-20"
  STATUS_TYPE                     = "APPROVED"
  STANDARD_VALUE_OUTPUT_FLAG      = "N"
  TEXT_FLAG                       = "N"
  TERSE_NAME                      = "md5checksum"
  SQL_FORMAT                      = "CHAR(32)"
  BL_SQL_FORMAT                   = "char(32)"
  DISPLAY_FORMAT                  = "JUSTLEFT"
  AVAILABLE_VALUE_TYPE            = "N/A"
END_OBJECT                        = ELEMENT_DEFINITION
END
```

```
PDS_VERSION_ID                  = PDS3
LABEL_REVISION_NOTE             = "2006-05-22, EN: EDR"

OBJECT                          = ELEMENT_DEFINITION
  ELEMENT_NAME                  = "checksum_type"
  BL_NAME                       = "checksumtype"
  DESCRIPTION                   = "
   The CHECKSUM_TYPE keyword is used to specify the type of checksum
   algorithm used to calculate a checksum for a file or data object."


  GENERAL_DATA_TYPE             = "IDENTIFIER"
  MAXIMUM                       = "N/A"
  MINIMUM                       = "N/A"
  MAXIMUM_LENGTH                = "12"
  MINIMUM_LENGTH                = "1"
  STANDARD_VALUE_TYPE           = "DYNAMIC"
  STANDARD_VALUE_SET            = {"MD5"}
  STANDARD_VALUE_SET_DESC       = "N/A"
  KEYWORD_DEFAULT_VALUE         = "N/A"
  UNIT_ID                       = "N/A"
  SOURCE_NAME                   = "PDS EN/E. RYE"
  FORMATION_RULE_DESC           = "N/A"
  SYSTEM_CLASSIFICATION_ID      = "COMMON"
  GENERAL_CLASSIFICATION_TYPE   = "N/A"
  CHANGE_DATE                   = "2006-03-20"
  STATUS_TYPE                   = "APPROVED"
  STANDARD_VALUE_OUTPUT_FLAG    = "Y"
  TEXT_FLAG                     = "N"
  TERSE_NAME                    = "checksumtype"
  SQL_FORMAT                    = "CHAR(12)"
  BL_SQL_FORMAT                 = "char(12)"
  DISPLAY_FORMAT                = "JUSTLEFT"
  AVAILABLE_VALUE_TYPE          = "N/A"
END_OBJECT                      = ELEMENT_DEFINITION
END
```

D.2 CHECKSUM.TAB and CHECKSUM.LBL

Each PDS volume may include a "CHECKSUM.TAB" file in the INDEX subdirectory. This file, when present, must be accompanied by a detached PDS label. The CHECKSUM.TAB file contains a checksum for every file contained within the volume with the exception of the checksum file itself and its label.

A CHECKSUM.TAB file is a PDS ASCII TABLE object comprising two required COLUMN objects. One COLUMN object has the name CHECKSUM; the other has the name FILE_SPECIFICATION_NAME. The definition of the CHECKSUM column must include the keyword-value pair "CHECKSUM_TYPE = MD5".

D.2.1 Example of CHECKSUM.TAB

```
1e8d45f622e09b9e2998af1a6d67a296 AAREADME.TXT
7dcfa51691ddd149a5a091ebe87b9bb1 ERRATA.TXT
f8dd7758cb5231c9e7817c4710d00b6e BROWSE/MARS/C1246XXX/I862934L.IMG
8ed31a70bc95aa104edbf4bc30a8c199 BROWSE/MARS/C1246XXX/I862934L.LBL
d8b83365f5e117b9665181944889da3d BROWSE/MARS/C1246XXX/I862934R.IMG
.
.
.
```

D.2.2 Example of CHECKSUM.LBL

```
PDS_VERSION_ID               = PDS3

RECORD_TYPE                  = FIXED_LENGTH
RECORD_BYTES                 = 71
FILE_RECORDS                 = 3623

DESCRIPTION                  = "CHECKSUM.TAB provides a checksum for all
                                files included on this archive volume, with
                                the exception of the checksum file itself
                                and its label."

^CHECKSUM_TABLE              = "CHECKSUM.TAB"

OBJECT                       = CHECKSUM_TABLE
  INTERCHANGE_FORMAT         = ASCII
  ROW_BYTES                  = 71
  ROWS                       = 3623
  COLUMNS                    = 2

  OBJECT                     = COLUMN
    NAME                     = CHECKSUM
    DESCRIPTION              = "The checksum of the indicated file."
    CHECKSUM_TYPE            = MD5
    DATA_TYPE                = CHARACTER
```

```
    START_BYTE              = 1
    BYTES                   = 32
  END_OBJECT                = COLUMN

  OBJECT                    = COLUMN
    NAME                    = FILE_SPECIFICATION_NAME
    DESCRIPTION             = "Identifies the file for which the checksum
                               was calculated."
    DATA_TYPE               = CHARACTER
    START_BYTE              = 34
    BYTES                   = 36
  END_OBJECT                = COLUMN

END_OBJECT                  = CHECKSUM_TABLE
END
```