TRUSTED® COMPUTING GROUP

SPECIFICATION

TCG Storage Opal Family Feature Set: Additional Datastore Tables

Version 1.01
Revision 1.17
July 25, 2023

Contact: admin@trustedcomputinggroup.org

Public Review

Work in Progress

This document is an intermediate draft for comment only and is subject to change without notice. Readers should not design products based on this document.

DISCLAIMERS, NOTICES, AND LICENSE TERMS

CHANGE HISTORY

VERSION	REVISION	DATE	DESCRIPTION
v1.00	1.00	2012.02.24	• Initial version
v1.01	1.17	2023.07.25	• Added ACE UID extension support

ACKNOWLEDGEMENT

The TCG wishes to thank all those who contributed to this specification. This document builds on work done in various working groups in the TCG and the industry at large.

Name	Company
Burt Wagner	CNEX Labs, Inc.
Chandra Nelogal	Dell Technologies, Inc.
Taku Kato	Kioxia Corporation
Alan Arnold	Lenovo Inc.
Alon Cohen	Microchip Technology
Bharath Madanayakanahalli Gururaj	Micron Technology, Inc.
Robert Strong	Micron Technology, Inc.
Walt Hubis	Micron Technology, Inc.
Artem Zankovich	Micron Technology, Inc.
Grigory Lyakhovitskiy	Microsoft
Eric Hibbard	Samsung Semiconductor Inc.
Santosh Kumar	Solidigm
Patrick Hery	Toshiba Corporation
Joseph Chen	ULINK Technology Inc.
Yoni Shternhell	Western Digital Technologies, Inc.

CONTENTS

DISCLAIMERS, NOTICES, AND LICENSE TERMS 1
CHANGE HISTORY 2
ACKNOWLEDGEMENT 3
CONTENTS 4
LIST OF TABLES 6
1 INTRODUCTION 7
1.1 Document Purpose 7
1.2 Scope and Intended Audience 7
1.3 Conventions 7
1.3.1 Key Words 7
1.3.2 Font Conventions 7
1.3.3 Statement Types 7
1.3.4 SP Table Cell Color Legend 8
1.3.5 List Conventions 9
1.3.5.1 Lists Overview 9
1.3.5.2 Unordered Lists 9
1.3.5.3 Ordered Lists 9
1.3.6 Numbering 10
1.3.7 Bit Conventions 10
1.3.8 Number Range Conventions 10
1.4 Document References 10
1.4.1 Document Precedence 10
1.4.2 Approved References 10
1.4.3 References Under Development 11
1.5 Dependencies on Other Feature Sets 11
1.6 Interactions with Other Feature Sets 11
1.7 Definition of Terms 11
2 ADDITIONAL DATASTORE TABLES OVERVIEW 12
3 SSC SPECIFIC FUNCTIONALITY 13
3.1 Methods 13
3.1.1 New Methods 13
3.1.2 Modified Methods 13
3.1.2.1 Activate 13
3.1.2.2 Reactivate 13
3.2 Tables 14
3.2.1 New Tables 14
3.2.1.1 DataStoreXXXX and DataStoreYYYY 14
3.2.2 Modified Tables 14
3.3 Types 14
3.3.1 New Types 14
3.3.2 Modified Types 14
4 FEATURE SET REQUIREMENTS 15

4.1 REQUIREMENTS OVERVIEW 15
4.2 LEVEL 0 DISCOVERY 15
4.2.1 Additional DataStore Tables Feature Descriptor (Feature Code = 0x0202) *15*
4.2.1.1 Feature Code 16
4.2.1.2 Feature Descriptor Version Number 16
4.2.1.3 Feature Set Minor Version Number 16
4.2.1.4 Length 16
4.2.1.5 Maximum number of DataStore tables 16
4.2.1.6 Maximum total size of DataStore tables 16
4.2.1.7 DataStore table size alignment 16
4.2.1.8 Level 0 requirements for the Additional DataStore Tables Feature Descriptor 16
4.3 ADMIN SP 17
4.3.1 Activate method *17*
4.4 LOCKING SP 17
4.4.1 Reactivate method *17*
4.4.2 Tables *18*
4.4.2.1 Table Table 18
4.4.2.2 AccessControl Table 19
4.4.2.3 ACE Table 24
4.5 ADDITIONAL SPS 25
4.6 OTHER FEATURE SET INTERACTIONS 25

List of Tables

Table 1 - SP Table Legend 8
Table 2 - Level 0 Discovery – Additional DataStore Tables Feature Descriptor 15
Table 3 - Feature Set Minor Versions 16
Table 4 Locking SP – Additions to Table Table Preconfiguration 18
Table 5 Locking SP – Additions to AccessControl Table Preconfiguration 19
Table 6 Locking SP – Additions to ACE Table Preconfiguration 24

1 Introduction

1.1 Document Purpose

The Storage Workgroup specifications provide a comprehensive architecture for Storage Devices under policy control as determined by the trusted platform host, the capabilities of the Storage Device to conform to the policies of the trusted platform, and the lifecycle state of the Storage Device as a Trusted Peripheral.

1.2 Scope and Intended Audience

This specification defines the Additional DataStore Tables Feature Set. Any Storage Device that claims compliance to the Additional DataStore Tables Feature Set SHALL conform to this specification.

The intended audience for this specification is both trusted Storage Device manufacturers and developers that want to use these Storage Devices in their systems.

1.3 Conventions

1.3.1 Key Words

Key words are used to signify requirements.

The Key Words “SHALL”, “SHALL NOT”, “SHOULD,” and “MAY” are used in this document. These words are a subset of the RFC 2119 (see [1]) key words used by TCG. These key words are to be interpreted as described in [1].

In addition to the above key words, the following are also used in this document to describe the requirements of particular features, including tables, methods, and usages thereof:

- Mandatory (M): When a feature is Mandatory, the feature SHALL be implemented. A Compliance test SHALL validate that the feature is operational.
- Optional (O): When a feature is Optional, the feature MAY be implemented. If implemented, a Compliance test SHALL validate that the feature is operational.
- Excluded (X): When a feature is Excluded, the feature SHALL NOT be implemented. A Compliance test SHALL validate that the feature is not operational.
- Not Required (N): When a feature is Not Required, the feature MAY be implemented. No Compliance test is required.

1.3.2 Font Conventions

Names of methods and SP tables are in `Courier New font` (e.g., the `Set` method, the `Locking` table). This convention does not apply to method and table names appearing in headings or captions.

Hexadecimal numbers are in `Courier New font`.
All other text is in the Arial font.

1.3.3 Statement Types

Please note a very important distinction between different sections of text throughout this document. There are two distinctive kinds of text: informative comment and normative statements. Because most of the text in this specification will be of the kind normative statements, the authors have informally defined it as the default and, as such, have specifically called out text of the kind informative comment. They have done this by flagging the beginning and end of each informative comment and highlighting its text in gray. This means that unless text is specifically marked as of the kind informative comment, it can be considered a kind of normative statements.

EXAMPLE: Start of Informative Comment

This is the first paragraph of 1-n paragraphs containing text of the kind *informative comment* ...

This is the second paragraph of text of the kind *informative comment* ...

This is the nth paragraph of text of the kind *informative comment* ...

To understand the TCG specification the user must read the specification. (This use of MUST does not require any action).

End of Informative Comment

1.3.4 SP Table Cell Color Legend

The legend in Table 1 defines the SP table cell color coding, with the RGB values for the shading of each cell indicated in parentheses. This color coding is informative only. The table cell content is normative. If a table cell is empty or blank (regardless of shading), then the contents of that cell are not specified in this specification. The contents may be specified in another specification.

Table 1 - SP Table Legend

Table Cell Legend	R-W	Value	Access Control	Comment
Arial-Narrow (230, 230, 230)	Read-only	Additional DataStore Tables Feature Set specified	Fixed	• Cell content is Read-Only. • Access control is fixed. • Value is specified by the Additional DataStore Tables Feature Set.
Arial Narrow bold-under (230, 230, 230)	Read-only	VU	Fixed	• Cell content is Read-Only. • Access Control is fixed. • Values are Vendor Unique (VU). A minimum or maximum value may be specified.
Arial-Narrow (0, 0, 0)	Not Defined	(N)	Not Defined	• Cell content is (N). • Access control is not defined. • Any text in table cell is informative only. • A Get MAY omit this column from the method response.
Arial Narrow bold-under (179, 179, 179)	Write	Preconfigured, user personalizable	Preconfigured, user personalizable	• Cell content is writable. • Access control is personalizable • Get Access Control is not described by this color coding
Arial-Narrow (179, 179, 179)	Write	Preconfigured, user personalizable	Fixed	• Cell content is writable. • Access control is fixed. • Get Access Control is not described by this color coding

1.3.5 List Conventions

1.3.5.1 Lists Overview

Lists are associated with an introductory paragraph or phrase, and are numbered relative to that paragraph or phrase (i.e., all lists begin with an a) or 1) entry).

Each item in a list is preceded by an identification with the style of the identification being determined by whether the list is intended to be an ordered list or an unordered list.

If the item in a list is not a complete sentence, the first word in the item is not capitalized. If the item in a list is a complete sentence, the first word in the item is capitalized.

Each item in a list ends with a semicolon, except the last item, which ends in a period. The next to the last entry in the list ends with a semicolon followed by an “and” or an “or” (i.e., “…; and”, or “…; or”). The “and” is used if all the items in the list are required. The “or” is used if only one or more items in the list are required.

1.3.5.2 Unordered Lists

An unordered list is one in which the order of the listed items is unimportant (i.e., it does not matter where in the list an item occurs as all items have equal importance). Each list item shall start with a lowercase letter followed by a close parenthesis. If it is necessary to subdivide a list item further with an additional unordered list (i.e., have a nested unordered list), then the nested unordered list shall be indented and each item in the nested unordered list shall start with an uppercase letter followed by a close parenthesis.

The following is an example of an unordered list with a nested unordered list:

EXAMPLE - The following are the items for the assembly:
a) a box containing:
    A) a bolt;
    B) a nut; and
    C) a washer;
b) a screwdriver; and
c) a wrench.

1.3.5.3 Ordered Lists

An ordered list is one in which the order of the listed items is important (i.e., item n is required before item n+1). Each listed item starts with a Western-Arabic numeral followed by a close parenthesis. If it is necessary to subdivide a list item further with an additional unordered list (i.e., have a nested unordered list), then the nested unordered list shall be indented and each item in the nested unordered list shall start with an uppercase letter followed by a close parenthesis.

The following is an example of an ordered list with a nested unordered list:

EXAMPLE - The following are the instructions for the assembly:
1) remove the contents from the box;
2) assemble the item;
    A) use a screwdriver to tighten the screws; and
    B) use a wrench to tighten the bolts;
    and
3) take a break.

1.3.6 Numbering

A binary number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 and 1 immediately followed by a lowercase b (e.g., 0101b). Underscores or spaces may be included between characters in binary number representations to increase readability or delineate field boundaries (e.g., 0 0101 1010b or 0_0101_1010b).

A hexadecimal number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 through 9 and/or the uppercase English letters A through F immediately preceded by "`0x`". Underscores or spaces may be included between characters in hexadecimal number representations to increase readability or delineate field boundaries (e.g., `0xFD8C FA23` or `0x0B_FD8C_FA23`). Hexadecimal numbers are in `Courier New font`.

A decimal number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 through 9 not immediately followed by a lowercase b or lowercase h (e.g., 25). This standard uses the following conventions for representing decimal numbers:

a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
b) the thousands separator (i.e., separating groups of three digits in a portion of the number) is a space; and
c) the thousands separator is used in both the integer portion and the fraction portion of a number.

A decimal number represented in this standard with an overline over one or more digits following the decimal point is a number where the overlined digits are infinitely repeating (e.g., $666.\overline{6}$ means 666.666 666… or 666 2/3, and $12.\overline{142857}$ means 12.142 857 142 857… or 12 1/7).

1.3.7 Bit Conventions

Name (n:m), where n is greater than m, denotes a set of bits (e.g., Feature (7:0)).

1.3.8 Number Range Conventions

p..q, where p is less than q, represents a range of numbers (e.g., words 100..103 represents words 100, 101, 102, and 103).

1.4 Document References

1.4.1 Document Precedence

If there is a conflict between this specification and any other reference, then the precedence is (where a lower number indicates higher precedence):

1. this specification;
2. references under development (see section 1.4.3); and
3. approved references (see section 1.4.2).

Each reference under development and each approved reference may specify its own document precedence.

1.4.2 Approved References

[1] IETF RFC 2119, 1997, "Key words for use in RFCs to Indicate Requirement Levels"

[2] Trusted Computing Group (TCG), "TCG Storage Architecture Core Specification", Version 2.01

[3] Trusted Computing Group (TCG), "Storage Interface Interactions Specification", Version 1.10

[4] Trusted Computing Group (TCG), "TCG Storage Security Subsystem Class: Opal", Version 2.02

[5] Trusted Computing Group (TCG), "TCG Storage Security Subsystem Class: Opalite", Version 1.00

[6] Trusted Computing Group (TCG), "TCG Storage Security Subsystem Class: Ruby", Version 1.00

[7] Trusted Computing Group (TCG), “TCG Storage Security Subsystem Class: Pyrite”, Version 1.00

1.4.3 References Under Development

None

1.5 Dependencies on Other Feature Sets

This feature set does not depend upon any other feature sets.

1.6 Interactions with Other Feature Sets

This feature set has no interactions with other feature sets.

1.7 Definition of Terms

Term	Definition
Locking SP	A security provider that incorporates the Locking Template as described in the Core Specification (see [2])
Locking SP is owned	A condition in which specific modifications of an SP have been made (see [3])
Opal family	Any of: Opal SSC [4], Opalite SSC [5], Ruby SSC [6], or Pyrite SSC [7]
SD	Storage Device
SP	Security Provider
SSC	Security Subsystem Class. SSC specifications describe profiled sets of TCG functionality
TPer	The Trusted Peripheral is the subset of a SD for which TCG manages security

2 Additional DataStore Tables Overview

Start of Informative Comment

The Additional DataStore Tables Feature Set defined in this specification is an extension of the existing mechanism provided via the DataStore table defined in [4], [5], [6] and [7]. This specification extends the DataStore table concept by providing a mechanism allowing the host to partition the space that the TPer allocates for that table into a number of independently managed DataStore tables.

The Additional DataStore Tables Feature Set provides the ability to store application-specific metadata in the TPer in such a way that:

- It is possible to create multiple independent DataStore tables
- It is possible to configure access control in such a way that multiple entities owning DataStore tables are prevented from accessing each other's data
- The storage area of the DataStore tables does not overlap with the User Area in order to avoid potential compatibility issues with existing host software

The DataStore tables can, in some implementations, be used to store key material without which C_PIN credentials cannot be obtained. Because of that, corruption of DataStore tables may have the same consequences as corruption of C_PIN or K_AES tables. TPer implementors should keep this in mind while designing redundancy mechanisms for DataStore tables.

End of Informative Comment

3 SSC Specific Functionality

This section specifies additional SSC-specific functionality (not contained in [2], [3], or [4], [5], [6], [7]) required to support the Additional DataStore Tables Feature Set.

3.1 Methods

This section defines new methods and modifications to existing methods required for this feature set.

3.1.1 New Methods

There are no new methods added to this feature set.

3.1.2 Modified Methods

This feature set modifies the following methods:

a) `Activate`

b) `Reactivate`

3.1.2.1 Activate

The `Activate` method of the Admin SP is modified to include an optional parameter DataStoreTableSizes as follows:

```
SPObjectUID.Activate[ DataStoreTableSizes = list [uinteger ...]]
=>
[ ]
```

If provided, the DataStoreTableSizes parameter (parameter number `0x060002`) specifies the number of DataStore tables that the TPer is requested to create and their sizes in bytes. The first element in the DataStoreTableSizes list corresponds to the `DataStore` table defined in [4], [5], [6] and [7]. The subsequent elements specify the desired sizes of the additional DataStore tables introduced by this feature set (see section 3.2.1.1).

The number and the sizes of the DataStore tables created via the `Activate` method are subject to certain constraints specific to the TPer. The constraints can be discovered via the Level 0 feature descriptor defined in section 4.2.1. If the TPer cannot satisfy a request for additional DataStore tables due to constraints related to the total number or size of the DataStore tables, the `Activate` method SHALL fail with the INSUFFICIENT_SPACE code. If the TPer cannot satisfy a request for additional DataStore tables due to constraints related to DataStore table size alignment, the `Activate` method SHALL fail with the INVALID_PARAMETER code.

If the DataStoreTableSizes parameter is not provided, the `Activate` method shall be processed as if the DataStoreTableSizes parameter was provided and consisted of a single element equal to the “Maximum total size of DataStore tables” field of the Additional DataStore Tables Feature Descriptor defined in section 4.2.1.

Refer to section 4.3.1 for additional requirements defined for the `Activate` method by this feature set.

3.1.2.2 Reactivate

If the Locking SP supports the `Reactivate` method, the method is modified to include an optional parameter DataStoreTableSizes as follows:

```
ThisSP.Reactivate[ DataStoreTableSizes = list [uinteger ...]]
=>
[ ]
```

If provided, the DataStoreTableSizes parameter (parameter number `0x060003`) specifies the number of DataStore tables that the TPer is requested to create upon Locking SP reactivation and their sizes in bytes. The first element in

the DataStoreTableSizes list corresponds to the DataStore table defined in [4], [5], [6] and [7]. The subsequent elements specify the desired sizes of the additional DataStore tables introduced by this feature set.

The number and the sizes of DataStore tables created via the Reactivate method are subject to certain constraints specific to the TPer. The constraints can be discovered via the Level 0 feature descriptor defined in section 4.2.1. If the TPer cannot satisfy a request for additional DataStore tables due to constraints related to the total number or size of the DataStore tables, the Reactivate method SHALL fail with the INSUFFICIENT_SPACE code. If the TPer cannot satisfy a request for additional DataStore table due to constraints related to DataStore table size alignment, the Reactivate method SHALL fail with the INVALID_PARAMETER code.

If the DataStoreTableSizes parameter is not provided, the Reactivate method shall be processed as if the DataStoreTableSizes parameter was provided and consisted of a single element equal to the “Maximum total size of DataStore tables” field of the Additional DataStore Tables Feature Descriptor defined in section 4.2.1.

Refer to section 4.4.1 for additional requirements defined for the Reactivate method by this feature set.

3.2 Tables

This section defines new tables and modifications to existing tables required for this feature set.

3.2.1 New Tables

3.2.1.1 DataStoreXXXX and DataStoreYYYY

A number of new DataStoreXXXX and DataStoreYYYY byte tables are added to the Locking SP. The new DataStoreXXXX and DataStoreYYYY tables are functionally equivalent to the DataStore table defined in [4], [5], [6] and [7] and can be used by the host to persistently store implementation-specific information using the Set and Get methods on the DataStoreXXXX and DataStoreYYYY objects.

3.2.2 Modified Tables

This feature set modifies the following tables of Locking SP:

a) Table Table
b) AccessControl Table
c) ACE Table

A number of new entries are added to the above tables based on the Maximum number of DataStore Tables defined in section 4.2.1.5. Refer to section 4.4.2 for additional details.

3.3 Types

This section defines new types and modifications to existing types required for this feature set.

3.3.1 New Types

There are no new types defined by this feature set.

3.3.2 Modified Types

There are no types modified by this feature set.

4 Feature Set Requirements

This section defines the Mandatory (M) and Optional (O) requirements for this feature set.

4.1 Requirements Overview

The Additional DataStore Tables Feature Set consists of DataStore Table Feature capabilities that MAY be implemented in a TPer. A Host discovers the DataStore Table Feature specific capabilities and properties of a TPer by examining the Additional DataStore Table Feature Descriptor.

4.2 Level 0 Discovery

An SD implementing this feature set SHALL:

a) return the Additional DataStore Tables Feature Descriptor as described in section 4.2.1; and

b) support the Level 0 Discovery response requirements defined in the applicable Opal Family SSC.

4.2.1 Additional DataStore Tables Feature Descriptor (Feature Code = 0x0202)

The Additional DataStore Tables Feature Descriptor SHALL be returned when the SD supports the Additional DataStore Tables Feature Set. The contents of the feature descriptor are defined in Table 2 - Level 0 Discovery – Additional DataStore Tables Feature Descriptor.

Table 2 - Level 0 Discovery – Additional DataStore Tables Feature Descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			Feature Code				
1								(LSB)
2	Feature Descriptor Version Number				Feature Set Minor Version Number			
3	Length							
4	Reserved							
5								
6	(MSB)			Maximum number of DataStore tables				
7								(LSB)
8	(MSB)			Maximum total size of DataStore tables				
9								
10								
11								(LSB)
12	(MSB)			DataStore table size alignment				
13								
14								
15								(LSB)

4.2.1.1 Feature Code
The Feature Code field value SHALL be set to 0x0202.

4.2.1.2 Feature Descriptor Version Number
The Feature Descriptor Version Number field SHALL be set to 0x2.

4.2.1.3 Feature Set Minor Version Number
The Feature Set Minor Version Number represents the minor version of the Additional DataStore Tables Feature Set supported by the SD.

The Feature Set Minor Version Number field SHALL be set to a value as specified in Table 3 - Feature Set Minor Versions.

Table 3 - Feature Set Minor Versions

Feature Set Minor Version Number	Specification Referenced
0x0	Additional DataStore Tables Feature Set v1.00
0x1	Additional DataStore Tables Feature Set v1.01
All others	Reserved

4.2.1.4 Length
The Length field indicates the number of bytes in the feature descriptor following byte 3. The value of the Length field SHALL be set to 0x0C.

4.2.1.5 Maximum number of DataStore tables
This field specifies the maximum number of DataStore tables that the TPer supports, including the DataStore table defined in [4], [5], [6] and [7].

4.2.1.6 Maximum total size of DataStore tables
This field specifies the maximum total size in bytes of all the DataStore tables that the TPer supports, including the DataStore table defined in [4], [5], [6] and [7].

4.2.1.7 DataStore table size alignment
This field specifies the size alignment in bytes for the DataStore tables other than the DataStore table defined in [4], [5], [6] and [7].

4.2.1.8 Level 0 requirements for the Additional DataStore Tables Feature Descriptor
- Maximum number of DataStore tables: 1 or more
- Maximum total size of DataStore tables:
  - An SD implementing Opal SSC [4] SHALL report 0x00A00000 or above
  - An SD implementing Opalite SSC [5] or Ruby SSC [6] or Pyrite SSC [7] SHALL report 0x00020000 or above
- DataStore table size alignment: 1 or above

4.3 Admin SP

An SD that supports this feature set SHALL contain the additions to the Admin SP specified in this section, in addition to the Admin SP requirements defined in [4], [5], [6] and [7] .

4.3.1 Activate method

An SD that supports this feature set supports the modifications to the `Activate` method defined in section 3.1.2.1.

The `Activate` method supports any DataStoreTableSizes parameter value that meets all of the following criteria:

- The number of elements in DataStoreTableSizes parameter is less than or equal to the value of the "Maximum number of DataStore tables" field of the Additional DataStore Tables Feature Descriptor.
- The total sum of all elements in the DataStoreTableSizes parameter is less than or equal to the value of the "Maximum total size of DataStore tables" field of the Additional DataStore Tables Feature Descriptor.
- The value of each element in the DataStoreTableSizes parameter is a multiple of the value of the "DataStore table size alignment" field of the Additional DataStore Tables Feature descriptor.

If the `Activate` method succeeds, the TPer creates `DataStore` tables in the Locking SP as described in section 3.2.1.1 and `ACE/AccessControl` elements for those tables as described in sections 4.4.2.2 and 4.4.2.3. The sizes of the `DataStore` tables can be retrieved from the Rows column of the `Table` Table after Locking SP activation.

4.4 Locking SP

An SD that supports this feature set SHALL contain the additions to the Locking SP specified in this section, in addition to the Locking SP requirements defined in [4], [5], [6] and [7] .

4.4.1 Reactivate method

An SD that supports this feature set and implements the `Reactivate` method supports the modifications to the `Reactivate` method defined in section 3.1.2.2.

The `Reactivate` method supports any DataStoreTableSizes parameter value that meets all of the following criteria:

- The number of elements in the DataStoreTableSizes parameter is less than or equal to the value of the "Maximum number of DataStore tables" field of the Additional DataStore Tables Feature Descriptor.
- The total sum of all elements in the DataStoreTableSizes parameter is less than or equal to the value of the "Maximum total size of DataStore tables" field of the Additional DataStore Tables Feature Descriptor.
- The value of each element in the DataStoreTableSizes parameter is a multiple of the value of the "DataStore table size alignment" field of the Additional DataStore Tables Feature Descriptor.

If the `Reactivate` method succeeds, the TPer replaces existing `DataStore` tables in the Locking SP with new `DataStore` tables as described in section 3.2.1.1. The content of the `DataStore` tables is not guaranteed to be preserved across a `Reactivate` method invocation, even if the DataStoreTableSizes list provided by the `Reactivate` method is the same as in the previous `Activate` or `Reactivate` method invocation.

The existing `ACE/AccessContol` elements for the DataStore tables are replaced with new `ACE/AccessControl` elements as described in sections 4.4.2.2 and 4.4.2.3. This results in the BooleanExpr column of the ACEs related to the DataStore tables being reset to "Admins". The sizes of the DataStore tables can be retrieved from the Rows column of the `Table` Table after Locking SP activation.

4.4.2 Tables

4.4.2.1 Table Table

A number of new entries are added to the `Table` Table for the additional DataStore tables, resulting in the following changes in the `Table` Table preconfiguration of the Locking SP.

*TT1 = The number of rows in the `DataStore` tables is specified via the `Activate`/ `Reactivate` method and subject to the constraints defined in section 4.2.1

*TT2 = The number of the `DataStore` tables is specified via the `Activate`/ `Reactivate` method and subject to the constraints defined in section 4.2.1

*TT3 = The column is added if it is defined in the applicable Opal Family SSC

DataStoreXXXX objects are added to the `Table` Table where XXXX can range from 2 to 128. If the number of DataStore tables as defined in section 4.2.1.5 exceeds 128 then extra DataStoreYYYY objects are added to the `Table` Table where YYYY can range from 129 to 2048.

Table 4 Locking SP – Additions to Table Table Preconfiguration

UID	Name	CommonName	TemplateID	Kind	Column	NumColumn	Rows	RowsFree	RowBytes	LastID	MinSize	MaxSize	MandatoryWriteGranularity *TT3	RecommendedAccess Granularity *TT3
00 00 00 01 00 00 10 02	“DataStore2”			Byte			*TT1						**VU**	**VU**
00 00 00 01 00 00 10 00 (+XXXX) *TT2	“DataStoreXXXX”			Byte			*TT1						**VU**	**VU**
00 00 00 01 00 00 10 00 (+YYYY) *TT2	“DataStoreYYYY”			Byte			*TT1						**VU**	**VU**

4.4.2.2 AccessControl Table

A number of new entries are added to the `AccessControl` Table for the additional DataStore tables, resulting in the following changes to `AccessControl` Table preconfiguration of the Locking SP.

*ACT = The number of `DataStore` related AccessControl entries depends on the number of `DataStore` tables, which is specified via the `Activate`/ `Reactivate` method and subject to constraints defined in section 4.2.1

Table 5 Locking SP – Additions to AccessControl Table Preconfiguration

Table Association - informative only	UID	InvokingID	InvokingID Name - informative only	MethodID	CommonName	ACL	Log	AddACEACL	RemoveACEACL	GetACLACL	DeleteMethodACL	AddACELog	RemoveACELog	GetACLLog	DeleteMethodLog	LogTo
ACE																
		00 00 00 08 00 03 FC 02 *ACT	ACE_DataStore2_Get_All	Set		ACE_ACE_Set_BooleanExpression				ACE_Anybody						
		00 00 00 08 00 03 FC 03 *ACT	ACE_DataStore2_Set_All	Set		ACE_ACE_Set_BooleanExpression				ACE_Anybody						

			Table Association - informative only
			UID
00 00 00 08 00 04 51 00 *ACT	00 00 00 08 00 03 FC 01 (+(XXXX – 1) * 2) *ACT	00 00 00 08 00 03 FC 00 (+(XXXX – 1) * 2) *ACT	**InvokingID**
ACE_DataStore129_Get_All	ACE_DataStoreXXXX_Set_All	ACE_DataStoreXXXX_Get_All	InvokingID Name - informative only
Set	Set	Set	**MethodID**
			CommonName
ACE_ACE_Set_BooleanExpression	ACE_ACE_Set_BooleanExpression	ACE_ACE_Set_BooleanExpression	**ACL**
			Log
			AddACEACL
			RemoveACEACL
ACE_Anybody	ACE_Anybody	ACE_Anybody	**GetACLACL**
			DeleteMethodACL
			AddACELog
			RemoveACELog
			GetACLLog
			DeleteMethodLog
			LogTo

			Table Association - informative only
			UID
00 00 00 08 00 04 50 01 (+(YYYY – 1) * 2) *ACT	00 00 00 08 00 04 50 00 (+(YYYY – 1) * 2) *ACT	00 00 00 08 00 04 51 01 *ACT	**InvokingID**
ACE_DataStoreYYYY_Set_All	ACE_DataStoreYYYY_Get_All	ACE_DataStore129_Set_All	InvokingID Name - informative only
Set	Set	Set	**MethodID**
			CommonName
ACE_ACE_Set_BooleanExpression	ACE_ACE_Set_BooleanExpression	ACE_ACE_Set_BooleanExpression	**ACL**
			Log
			AddACEACL
			RemoveACEACL
ACE_Anybody	ACE_Anybody	ACE_Anybody	**GetACLACL**
			DeleteMethodACL
			AddACELog
			RemoveACELog
			GetACLLog
			DeleteMethodLog
			LogTo

				DataStore	Table Association - informative only
					UID
00 00 10 00 00 00 00 00 (+XXXX * 2^{32}) *ACT	00 00 10 00 00 00 00 00 (+XXXX * 2^{32}) *ACT	00 00 10 02 00 00 00 00 *ACT	00 00 10 02 00 00 00 00 *ACT		**InvokingID**
DataStoreXXXX	DataStoreXXXX	DataStore 2	DataStore2		InvokingID Name - informative only
Set	Get	Set	Get		**MethodID**
					CommonName
ACE_DataStoreXXXX_Set_All	ACE_DataStoreXXXX_Get_All	ACE_DataStore2_Set_All	ACE_DataStore2_Get_All		**ACL**
					Log
					AddACEACL
					RemoveACEACL
ACE_Anybody	ACE_Anybody	ACE_Anybody	ACE_Anybody		**GetACLACL**
					DeleteMethodACL
					AddACELog
					RemoveACELog
					GetACLLog
					DeleteMethodLog
					LogTo

				Table Association - informative only
				UID
00 00 10 00 00 00 00 00 (+YYYY * 2^{32}) *ACT	00 00 10 00 00 00 00 00 (+YYYY * 2^{32}) *ACT	00 00 10 81 00 00 00 00	00 00 10 81 00 00 00 00	**InvokingID**
DataStoreYYYY	DataStoreYYYY	DataStore129	DataStore129	InvokingID Name - informative only
Set	Get	Set	Get	**MethodID**
				CommonName
ACE_DataStoreYYYY_Set_All	ACE_DataStoreYYYY_Get_All	ACE_DataStore129_Set_All	ACE_DataStore129_Get_All	**ACL**
				Log
				AddACEACL
				RemoveACEACL
ACE_Anybody	ACE_Anybody	ACE_Anybody	ACE_Anybody	**GetACLACL**
				DeleteMethodACL
				AddACELog
				RemoveACELog
				GetACLLog
				DeleteMethodLog
				LogTo

4.4.2.3 ACE Table

A number of new entries are added to the ACE Table for the additional DataStore tables, resulting in the following changes in ACE table preconfiguration of the Locking SP.

*ACET = The number of DataStore related ACEs depends on the number of DataStore tables, which is specified via the Activate / Reactivate method and subject to constraints defined in section 4.2.1

ACE_DataStoreXXXX objects are added to the ACE Table where XXXX can range from 2 to 128. If the number of DataStore tables as defined in section 4.2.1.5 exceeds 128 then extra ACE_DataStoreYYYY objects are added to the ACE Table where YYYY can range from 129 to 2048.

Table 6 Locking SP – Additions to ACE Table Preconfiguration

Table Association –Informative Column	UID	Name	CommonName	BooleanExpr	Columns
DataStore					
	00 00 00 08 00 03 FC 02 *ACET	"ACE_DataStore2_Get_All"		Admins	All
	00 00 00 08 00 03 FC 03 *ACET	"ACE_DataStore2_Set_All"		Admins	All
	00 00 00 08 00 03 FC 00 (+(XXXX – 1) * 2) *ACET	"ACE_DataStoreXXXX_Get_All"		Admins	All
	00 00 00 08 00 03 FC 01 (+(XXXX – 1) * 2) *ACET	"ACE_DataStoreXXXX_Set_All"		Admins	All
	00 00 00 08 00 04 51 00 *ACET	"ACE_DataStore129_Get_All"		Admins	All
	00 00 00 08 00 04 51 01 *ACET	"ACE_DataStore129_Set_All"		Admins	All

Table Association -Informative Column	UID	Name	CommonName	BooleanExpr	Columns
	00 00 00 08 00 04 50 00 (+(YYYY – 1) * 2) *ACET	“ACE_DataStoreYYYY_Get_All”		Admins	All
	00 00 00 08 00 04 50 01 (+(YYYY – 1) * 2) *ACET	“ACE_DataStoreYYYY_Set_All”		Admins	All

4.5 Additional SPs

This feature set requires no additional SPs.

4.6 Other Feature Set Interactions

This feature set has no interactions with other feature sets.