TCG TSS 2.0 Marshaling/Unmarshaling API Specification

Family "2.0"

Version 1.0 Revision 04

4 January 2018

Committee Draft

Copyright © TCG 2013-2018

- ***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

Contact: admin@trustedcomputinggroup.org

TCG Public Review

TCG

Disclaimers, Notices, and License Terms

Copyright Licenses:

- Trusted Computing Group (TCG) grants to the user of the source code in this specification (the “Source Code”) a worldwide, irrevocable, nonexclusive, royalty free, copyright license to reproduce, create derivative works, distribute, display and perform the Source Code and derivative works thereof, and to grant others the rights granted herein.

- The TCG grants to the user of the other parts of the specification (other than the Source Code) the rights to reproduce, distribute, display, and perform the specification solely for the purpose of developing products based on such documents.

Source Code Distribution Conditions:

- Redistributions of Source Code must retain the above copyright licenses, this list of conditions and the following disclaimers.

- Redistributions in binary form must reproduce the above copyright licenses, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution.

Disclaimers:

- THE COPYRIGHT LICENSES SET FORTH ABOVE DO NOT REPRESENT ANY FORM OF LICENSE OR WAIVER, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, WITH RESPECT TO PATENT RIGHTS HELD BY TCG MEMBERS (OR OTHER THIRD PARTIES) THAT MAY BE NECESSARY TO IMPLEMENT THIS SPECIFICATION OR OTHERWISE. Contact TCG Administration (admin@trustedcomputinggroup.org) for information on specification licensing rights available through TCG membership agreements.

- THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO EXPRESS OR IMPLIED WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ACCURACY, COMPLETENESS, OR NONINFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

- Without limitation, TCG and its members and licensors disclaim all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

Any marks and brands contained herein are the property of their respective owners.

Corrections and Comments

Please send comments and corrections to techquestionsadmin@trustedcomputinggroup.org.

Normative-Informative Language

"SHALL," "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this document are normative statements. They are to be interpreted as described in [RFC-2119].

Revision History

Revision	Date	Description
Version 0.9 rev 00	05/30/17	• Removed Marshalling/Unmarshalling from the SAPI specification and moved it into this specification.

Acknowledgements

TCG and the TSS Work Group would like to thank the following people for their work on this specification.

- Will Arthur, Raytheon
- Brenda Baggaley, Security Innovation (OnBoard Security)
- Dave Challener, Johns Hopkins University
- Mike Cox, Security Innovation (OnBoard Security)
- Andreas Fuchs, Fraunhofer SIT
- Ken Goldman, IBM
- Jürgen Repp, Fraunhofer SIT
- Philip Tricca, Intel
- Lee Wilson, Security Innovation (OnBoard Security)

Table of Contents

1 General Information on The TCG TSS 2.0 Specification Library

1.1 Acronyms

For definitions of the acronyms used in the TSS 2.0 specifications please see the TCG TSS 2.0 Overview and Common Structures Specification [22}.

1.2 TCG Software Stack 2.0 (TSS 2.0) Specification Library Structure

At the time of writing, the documents that are part of the specification of the TSS 2.0 are:

[1] TCG TSS 2.0 Overview and Common Structures Specification

[2] TCG TSS 2.0 TPM Command Transmission Interface (TCTI) API Specification

[3] TCG TSS 2.0 Marshaling/Unmarshaling API Specification

[4] TCG TSS 2.0 System API (SAPI) Specification

[5] TCG TSS 2.0 Enhanced System API (ESAPI) Specification

[6] TCG TSS 2.0 Feature API (FAPI) Specification

[7] TCG TSS 2.0 TAB and Resource Manager Specification

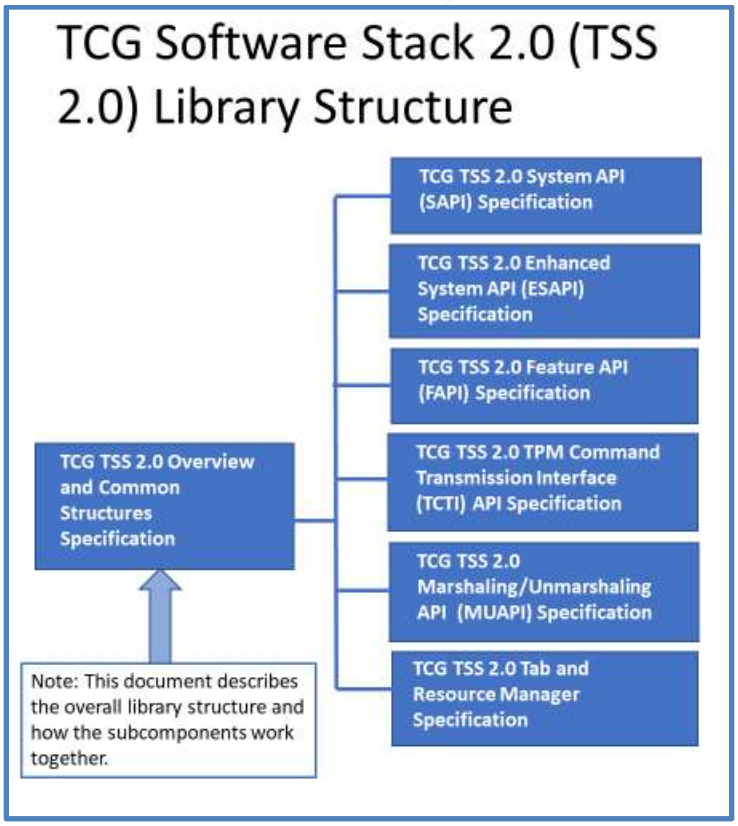

Figure 1: TSS 2.0 Specification Library

All references and acronyms for the TSS 2.0 library are in the TCG TSS 2.0 Overview and Common Structures Specification [22].

2 Marshaling/Unmarshaling (MUAPI) Overview

The MUAPI builds TPM command byte streams (marshalling) and decomposes TPM response byte streams (unmarshalling). It is useful for both the SAPI and ESAPI and is therefore kept in its own namespace with its own API.

The MUAPI is in a separate namespace that is shared by both the SAPI and ESAPI to do all their marshaling and unmarshaling functions. This makes it unnecessary to replicate this complex functionality in both the SAPI and ESAPI layers.

3 MUAPI module for Marshaling and Unmarshaling

The TPM receives commands and returns responses in formatted data streams as described in section 18 of the TPM2 Architecture Specification. Application programs are primarily concerned with using TPM functions, not the details of marshaling data between the language specific representation and the network byte ordered TPM data stream representation. For this purpose the TSS defines a standard set of functions for marshaling and unmarshaling the C representation of each TPM2 structure defined in the TPM2 Structures Specification. These functions are primarily intended for use by the upper layers of the TSS however they may be equally useful for those creating debugging or visualization tools.

Each marshaling and unmarshaling function is named using the general form: Tss2_MU_<TYPE_NAME>_Marshal and Tss2_MU_<TYPE_NAME>_Unmarshal. Each type defined in the TPM2 Structures Specification will have a corresponding Marshal / Unmarshal function. The function name is made unique by replacing '<TYPE_NAME>' with the name of the type from the TPM2 structures specification. The parameters to each Marshal / Unmarshal function are defined by a grouping. These groupings and the associated function prototypes are described in the following subsections.

3.1 Data Structures and Function Prototypes for the MUAPI Module

MUAPI prototypes are included in the header file, tss2_mu.h. The contents of tss2_mu.h are specified in section 4.

3.2 Marshalling TPMA and TPM base types

Marshhalling functions for this group of types take the form

```
TSS2_RC
Tss2_MU_<TYPE_NAME>_Marshal(
    <TYPE_NAME> src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset
    );
```

3.2.1 Parameters

- The `src` parameter is an instance of the type <TYPE_NAME> that shall be marshaled.
- The `buffer` parameter is a byte array where the marshaled form of the instance will be written.
- The `buffer_size` parameter is the size in bytes of the byte array represented by the 'buffer' parameter.
- The `offset` parameter specifies the starting byte in the destination buffer where the `src` data will be written / marshaled.

3.2.2 Behavior

This collection of functions marshals all TPM2 types that map directly to fixed size primitive types in the C programming language.

The 'offset' parameter will be updated by the function to the index of the next byte after the marshaled data in the 'buffer' byte array. For the purposes of this collection of functions the ‘offset’ parameter will be advanced by the size of the type being marshalled in bytes.

If the 'buffer' parameter is NULL the implementation shall not write any marshaled data but the 'offset' parameter shall be updated as though it had.

If an error is returned, 'offset' shall not be modified even though some data may be written to the output buffer.

The implementation shall not write data to the buffer array before the supplied 'offset' value and shall not write data beyond the end of the buffer.

Response Codes

- TSS2_MU_RC_INSUFFICIENT_BUFFER shall be returned when carrying out the marshaling operation would overrun the end of the provided buffer. This shall be the case even if the provided 'buffer' is null.
- TSS2_MU_RC_BAD_REFERENCE shall be returned when the 'buffer' and 'offset' parameters are both NULL.

3.3 Marshaling TPMS and TPM2B types

Marshaling functions for this first group of types take the form:

```
TSS2_RC
Tss2_MU_<TYPE_NAME>_Marshal(
    <TYPE_NAME> const *src,
    uint8_t                         buffer[],
    size_t                           buffer_size,
    size_t                          *offset
    );
```

3.3.1 Parameters

- The `src` parameter is a reference to an instance of the type <TYPE_NAME> that shall be marshaled.
- The `buffer` parameter is a byte array where the marshaled form of the instance will be written.
- The `buffer_size` parameter is the size in bytes of the byte array represented by the `buffer` parameter.
- The `offset` parameter specifies the starting byte in the destination buffer where the `src` data will be written / marshaled.

3.3.2 Behavior

The behavior of the marshaling function will depend upon the type being marshaled. Functions marshaling simple TPM2B_* types will consult the relevant size field to determine the size of the 'src' parameter. Functions marshaling complex TPM2B_* types (e.g. TPM2B_SENSITVE_CREATE) will ignore the supplied src->size field. Instead, the size is calculated from the type and the contents of the supplied data structure. This value is marshalled in place of the size field.

The `offset` parameter will be updated by the function to the index of the next byte after the marshaled data in the `buffer` byte array.

If the `buffer` parameter is NULL the implementation will not write any marshaled data but the `offset` parameter shall be updated as though it had.

If an error is returned, `offset` will not be modified even though some data may be written to the output buffer.

The implementation will not write data to the buffer array before the supplied `offset` value and shall not write data beyond the end of the buffer.

Response Codes:

- **TSS2_MU_RC_BAD_REFERENCE: shall be returned when the 'src' parameter is NULL**
- **TSS2_MU_RC_INSUFFICIENT_BUFFER: shall be returned when carrying out the marshaling operation would overrun the end of the provided buffer. This shall be the case even if the provided 'buffer' is null.**
- **TSS2_MU_RC_BAD_REFERENCE: shall be returned when the 'buffer' and 'offset' parameters are both NULL.**
- **TSS2_MU_RC_BAD_VALUE: shall be returned when an invalid enumeration is encountered while processing the selector field for a union type.**
- **TSS2_MU_RC_BAD_SIZE: shall be returned when the type of the 'src' parameter is a simple TPM2B and the 'size' field in the structure is larger than allowed for the type (e.g. size > sizeof (TPMU_HA) for the TPM2_DIGEST type).**

Additionally the marshaling functions shall propagate response codes that may be returned by any marshalling function invoked internally.

3.4 Unmarshaling TPMS_*, TPM2B_*, TPMA_* and TPM base types

Unmarshaling functions for this first group of types take the form:

```
TSS2_RC
Tss2_MU_<TYPE_NAME>_Unmarshal (
    uint8_t const      buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    <TYPE_NAME> *dest
    );
```

3.4.1 Parameters

- The 'buffer' parameter is a byte array where the marshaled form of the <TYPE_NAME> type resides.
- The 'buffer_size' parameter is the size in bytes of the byte array represented by the 'buffer' parameter.
- The 'offset' parameter specifies the starting byte in the source buffer where the marshaled data shall be read.
- The 'dest' parameter is a reference to an instance of the type <TYPE_NAME> that shall be populated with the unmarshaled data.

3.4.2 Behaviour

Unmarshaling function behaviour depends upon the type of the `dest` parameter. Functions unmarshaling simple TPM2B_* types shall consult the relevant size field in the marshaled data to determine the size of the data in the `buffer`. The size in the `dest` instance must be either the caller allocated size or 0 to denote the default allocation for this type. If the caller allocated size is less than the size in the marshaled data an error shall be returned. Functions unmarshaling complex TPM2B_* types (e.g. TPM2B_SENSITVE_CREATE) shall ensure that the caller provided `dest` size field has been initialized to 0.

The function shall calculate the size of the marshaled data and compare this to the size represented in the marshaled form. An error shall be returned if the two sizes do not match or if the size of the unmarshaled data would exceed that of the destination type.

The `offset` parameter shall be updated by the function to the index of the next byte after the unmarshaled data in the `buffer` byte array.

If the `dest` parameter is NULL the implementation shall update the `offset` as it would have if `dest` were non-NULL.

If `offset` is NULL, unmarshaling shall begin at offset 0.

If an error is returned, `offset` shall not be modified even though some data may have been written to the object referenced by the `dest` parameter.

Implementations shall not read data from the buffer array before the supplied `offset`.

Implementations shall not read data beyond the end of the buffer.

Response Codes:

- **TSS2_MU_RC_BAD_REFERENCE: shall be returned if the `buffer` parameter is NULL.**
- **TSS2_MU_RC_INSUFFICIENT_BUFFER: if the unmarshaling would overrun the end of the buffer array.**
- **TSS2_MU_RC_BAD_REFERENCE: shall be returned if the `offset` and `dest` are NULL**
- **TSS2_MU_RC_BAD_VALUE: if the parsing fails due to an invalid enumeration value. This check will be performed for all interface types (as defined in the TPM specification) encountered, regardless of whether the destination object is NULL or not. In cases where an interface type has conditional value(s), the conditional value(s) are always accepted as valid.**

- **TSS2_MU_RC_BAD_SIZE: if the size in the marshaled data does not match the calculated size.**

Additionally the unmarshaling functions shall propagate response codes that may be returned by any unmarshaling function invoked internally.

3.5 Marshaling TPMU_* types

The second group of marshaling functions is specific to TPMU (union) types. The associated marshaling functions take the form:

```
TSS2_RC
Tss2_MU_<TYPE_NAME>_Marshal (
    <TYPE_NAME> const *src,
    uint32_t                        selector_value,
    uint8_t                         buffer[],
    size_t                           buffer_size,
    size_t                          *offset
    );
```

3.5.1 Parameters

- The `src` parameter is a reference to an instance of the type being marshaled.
- The `selector_value` identifies which member of the union shall be marshalled.
- The `buffer` is a reference to a memory region where the marshalled representation shall be written.
- The `buffer_size` is the total size of the memory region referenced by the `buffer` parameter.
- The `offset` parameter is a reference to the index into the `buffer` array where the first byte of the marshalled data shall be written.

3.5.2 Behaviour

The behaviour of the marshaling function will depend upon the selector value and the type of the union member selected.

The `offset` parameter shall be updated by the function to the index of the next byte after the marshaled data in the `buffer` byte array.

If the `buffer` parameter is NULL the implementation shall not write any marshaled data but the `offset` parameter shall be updated as though it had.

If an error is returned, `offset` shall not be modified even though some data may be written to the output buffer.

The implementation shall not write data to the buffer array before the supplied `offset` value and shall not write data beyond the end of the buffer.

Response Codes

- **TSS2_MU_RC_BAD_REFERENCE: shall be returned when the 'src' parameter is NULL.**

- **TSS2_MU_RC_INSUFFICIENT_BUFFER: shall be returned if the marshaling would overrun the end of buffer. This condition applies to length queries when buffer is NULL.**
- **TSS2_MU_RC_BAD_VALUE: shall be returned if the encoding fails because it encountered an invalid enumeration value for a union selector field.**

Additionally the TPMU_* marshaling functions shall propagate response codes that may be returned by any marshalling function invoked as part of the TPMU_* marshaling.

3.6 Unmarshaling TPMU_* types

Unmarshaling functions for types in this second group take the form:

```
TSS2_RC
Tss2_MU_<TYPE_NAME>_Unmarshal (
    uint8_t const      buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    uint32_t               selector_value,
    <TYPE_NAME> *dest
    );
```

3.6.1 Parameters

- The `buffer` is a reference to a memory region where the marshalled representation resides.
- The `buffer_size` parameter is the total size of the memory region referenced by the `buffer` parameter.
- The `selector_value` identifies which member of the union shall be marshalled.
- The `offset` parameter is a reference to the index into the `buffer` array where the first byte of the marshalled data shall be written.
- The `dest` parameter is a reference to an instance of the type being unmarshaled. The unmarshaled data shall be written to this instance.

3.6.2 Behaviour

The marshaled representation of <TYPE_NAME> shall be read from `buffer` at offset `offset` and written to the instance of <TYPE_NAME> at `dest`. The underlying behaviour of the marshaling function will depend upon the selector value and the type of the union member selected. If `offset` is NULL the marshaling shall begin at offset 0. If `dest` is NULL the call will update the value referenced by `offset` to the size of the data to be unmarshaled.

NOTE: In some cases there is no selector given in the tables of TPM library specification Part 2. In those cases the size is used as selector.

If a response code indicating an error is returned, `offset` shall not be modified irrespective of whether `dest` is modified.

Response Codes

- **TSS2_MU_RC_BAD_REFERENCE: if src is NULL**
- **TSS2_MU_RC_INSUFFICIENT_BUFFER; if the marshaling would overrun the end of buffer (indicated by buffer_size). This condition applies even to length queries when buffer is NULL**
- **TSS2_MU_RC_BAD_VALUE: if the encoding fails due to an invalid enumeration value in a union selector field**
- **Additionally the TPMU_* unmarshaling functions shall propagate response codes that may be returned by any unmarshaling function invoked as part of the TPMU_* unmarshaling.**

4 Type Marshaling and Unmarshaling Header File

tss2_mu.h

4.1 tss2_mu.h Prelude

```
#ifndef TSS2_MU_H
#define TSS2_MU_H

#include <stdlib.h>
#include "tss2_common.h"
#include "tss2_tpm2_types.h"

#ifndef TSS2_API_VERSION_1_2_1_108
#error Version mismatch among TSS2 header files.
#endif

#ifdef __cplusplus
extern "C" {
#endif
```

4.2 tss2_mu.h Marshaling Functions

```
/*
 * Marshaling Functions
 */
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT8_Marshal(
    INT8    src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT8_Marshal(
    UINT8   src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT16_Marshal(
    INT16   src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT16_Marshal(
    UINT16  src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT32_Marshal(
    INT32   src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT32_Marshal(
    UINT32  src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT64_Marshal(
    INT64   src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT64_Marshal(
    UINT64  src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_BYTE_Marshal(
    BYTE    src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ALGORITHM_ID_Marshal(
    TPM2_ALGORITHM_ID src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_MODIFIER_INDICATOR_Marshal(
    TPM2_MODIFIER_INDICATOR src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_AUTHORIZATION_SIZE_Marshal(
    TPM2_AUTHORIZATION_SIZE src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PARAMETER_SIZE_Marshal(
    TPM2_PARAMETER_SIZE src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_KEY_SIZE_Marshal(
    TPM2_KEY_SIZE src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_KEY_BITS_Marshal(
```

```
    TPM2_KEY_BITS src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SPEC_Marshal(
    TPM2_SPEC src,
    uint8_t   buffer[],
    size_t    buffer_size,
    size_t   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_GENERATED_Marshal(
    TPM2_GENERATED src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ALG_ID_Marshal(
    TPM2_ALG_ID src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ECC_CURVE_Marshal(
    TPM2_ECC_CURVE src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CC_Marshal(
    TPM2_CC src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_RC_Marshal(
    TPM2_RC src,
```

```
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CLOCK_ADJUST_Marshal(
    TPM2_CLOCK_ADJUST src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_EO_Marshal(
    TPM2_EO src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ST_Marshal(
    TPM2_ST src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SU_Marshal(
    TPM2_SU src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SE_Marshal(
    TPM2_SE src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CAP_Marshal(
    TPM2_CAP src,
    uint8_t  buffer[],
```

```
    size_t   buffer_size,
    size_t  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PT_Marshal(
    TPM2_PT src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PT_PCR_Marshal(
    TPM2_PT_PCR src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PS_Marshal(
    TPM2_PS src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HANDLE_Marshal(
    TPM2_HANDLE src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HT_Marshal(
    TPM2_HT src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_RH_Marshal(
    TPM2_RH src,
    uint8_t buffer[],
    size_t  buffer_size,
```

```
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HC_Marshal(
    TPM2_HC src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_ALGORITHM_Marshal(
    TPMA_ALGORITHM src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_OBJECT_Marshal(
    TPMA_OBJECT src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_SESSION_Marshal(
    TPMA_SESSION src,
    uint8_t      buffer[],
    size_t       buffer_size,
    size_t      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_LOCALITY_Marshal(
    TPMA_LOCALITY src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_PERMANENT_Marshal(
    TPMA_PERMANENT src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_STARTUP_CLEAR_Marshal(
    TPMA_STARTUP_CLEAR src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_MEMORY_Marshal(
    TPMA_MEMORY src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_CC_Marshal(
    TPMA_CC src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_MODES_Marshal(
    TPMA_MODES src,
    uint8_t    buffer[],
    size_t     buffer_size,
    size_t    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_YES_NO_Marshal(
    TPMI_YES_NO src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_OBJECT_Marshal(
    TPMI_DH_OBJECT src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PARENT_Marshal(
    TPMI_DH_PARENT src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PERSISTENT_Marshal(
    TPMI_DH_PERSISTENT src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_ENTITY_Marshal(
    TPMI_DH_ENTITY src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PCR_Marshal(
    TPMI_DH_PCR src,
    uint8_t     buffer[],
    size_t      buffer_size,
    size_t     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_AUTH_SESSION_Marshal(
    TPMI_SH_AUTH_SESSION src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_HMAC_Marshal(
    TPMI_SH_HMAC src,
    uint8_t      buffer[],
    size_t       buffer_size,
    size_t      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_POLICY_Marshal(
```

```
    TPMI_SH_POLICY src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_CONTEXT_Marshal(
    TPMI_DH_CONTEXT src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_HIERARCHY_Marshal(
    TPMI_RH_HIERARCHY src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_ENABLES_Marshal(
    TPMI_RH_ENABLES src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_HIERARCHY_AUTH_Marshal(
    TPMI_RH_HIERARCHY_AUTH src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_PLATFORM_Marshal(
    TPMI_RH_PLATFORM src,
    uint8_t          buffer[],
    size_t           buffer_size,
    size_t          *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_OWNER_Marshal(
    TPMI_RH_OWNER src,
```

```
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_ENDORSEMENT_Marshal(
    TPMI_RH_ENDORSEMENT src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_PROVISION_Marshal(
    TPMI_RH_PROVISION src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_CLEAR_Marshal(
    TPMI_RH_CLEAR src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_NV_AUTH_Marshal(
    TPMI_RH_NV_AUTH src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_LOCKOUT_Marshal(
    TPMI_RH_LOCKOUT src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_NV_INDEX_Marshal(
    TPMI_RH_NV_INDEX src,
    uint8_t          buffer[],
```

```
    size_t           buffer_size,
    size_t          *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_HASH_Marshal(
    TPMI_ALG_HASH src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ASYM_Marshal(
    TPMI_ALG_ASYM src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_Marshal(
    TPMI_ALG_SYM src,
    uint8_t      buffer[],
    size_t       buffer_size,
    size_t      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_OBJECT_Marshal(
    TPMI_ALG_SYM_OBJECT src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_MODE_Marshal(
    TPMI_ALG_SYM_MODE src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_KDF_Marshal(
    TPMI_ALG_KDF src,
    uint8_t      buffer[],
    size_t       buffer_size,
```

```
    size_t      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SIG_SCHEME_Marshal(
    TPMI_ALG_SIG_SCHEME src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ECC_KEY_EXCHANGE_Marshal(
    TPMI_ECC_KEY_EXCHANGE src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ST_COMMAND_TAG_Marshal(
    TPMI_ST_COMMAND_TAG src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_EMPTY_Marshal(
    TPMS_EMPTY const *src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALGORITHM_DESCRIPTION_Marshal(
    TPMS_ALGORITHM_DESCRIPTION const *src,
    uint8_t                           buffer[],
    size_t                            buffer_size,
    size_t                           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_HA_Marshal(
    TPMU_HA const *src,
    uint32_t       selector,
    uint8_t        buffer[],
    size_t         buffer_size,
```

```
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_HA_Marshal(
    TPMT_HA const *src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DIGEST_Marshal(
    TPM2B_DIGEST const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DATA_Marshal(
    TPM2B_DATA const *src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NONCE_Marshal(
    TPM2B_NONCE const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_AUTH_Marshal(
    TPM2B_AUTH const *src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_OPERAND_Marshal(
    TPM2B_OPERAND const *src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_EVENT_Marshal(
    TPM2B_EVENT const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_MAX_BUFFER_Marshal(
    TPM2B_MAX_BUFFER const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_MAX_NV_BUFFER_Marshal(
    TPM2B_MAX_NV_BUFFER const *src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_TIMEOUT_Marshal(
    TPM2B_TIMEOUT const *src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_IV_Marshal(
    TPM2B_IV const *src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_NAME_Marshal(
    TPMU_NAME const *src,
    uint32_t         selector,
    uint8_t          buffer[],
    size_t           buffer_size,
    size_t          *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NAME_Marshal(
    TPM2B_NAME const *src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_PCR_SELECT_Marshal(
    TPMS_PCR_SELECT const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_PCR_SELECTION_Marshal(
    TPMS_PCR_SELECTION const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_CREATION_Marshal(
    TPMT_TK_CREATION const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_VERIFIED_Marshal(
    TPMT_TK_VERIFIED const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_AUTH_Marshal(
    TPMT_TK_AUTH const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_HASHCHECK_Marshal(
    TPMT_TK_HASHCHECK const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALG_PROPERTY_Marshal(
    TPMS_ALG_PROPERTY const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_PROPERTY_Marshal(
    TPMS_TAGGED_PROPERTY const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_PCR_SELECT_Marshal(
    TPMS_TAGGED_PCR_SELECT const *src,
    uint8_t                       buffer[],
    size_t                        buffer_size,
    size_t                       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_POLICY_Marshal(
    TPMS_TAGGED_POLICY const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_CC_Marshal(
    TPML_CC const *src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_CCA_Marshal(
```

```
    TPML_CCA const *src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ALG_Marshal(
    TPML_ALG const *src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_HANDLE_Marshal(
    TPML_HANDLE const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_DIGEST_Marshal(
    TPML_DIGEST const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_DIGEST_VALUES_Marshal(
    TPML_DIGEST_VALUES const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_PCR_SELECTION_Marshal(
    TPML_PCR_SELECTION const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ALG_PROPERTY_Marshal(
    TPML_ALG_PROPERTY const *src,
```

```
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_TPM_PROPERTY_Marshal(
    TPML_TAGGED_TPM_PROPERTY const *src,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_PCR_PROPERTY_Marshal(
    TPML_TAGGED_PCR_PROPERTY const *src,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ECC_CURVE_Marshal(
    TPML_ECC_CURVE const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_POLICY_Marshal(
    TPML_TAGGED_POLICY const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_CAPABILITIES_Marshal(
    TPMU_CAPABILITIES const *src,
    uint32_t                 selector,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CAPABILITY_DATA_Marshal(
    TPMS_CAPABILITY_DATA const *src,
```

```
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CLOCK_INFO_Marshal(
    TPMS_CLOCK_INFO const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TIME_INFO_Marshal(
    TPMS_TIME_INFO const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TIME_ATTEST_INFO_Marshal(
    TPMS_TIME_ATTEST_INFO const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CERTIFY_INFO_Marshal(
    TPMS_CERTIFY_INFO const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_QUOTE_INFO_Marshal(
    TPMS_QUOTE_INFO const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_COMMAND_AUDIT_INFO_Marshal(
    TPMS_COMMAND_AUDIT_INFO const *src,
    uint8_t                       buffer[],
```

```
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SESSION_AUDIT_INFO_Marshal(
    TPMS_SESSION_AUDIT_INFO const *src,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CREATION_INFO_Marshal(
    TPMS_CREATION_INFO const *src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_CERTIFY_INFO_Marshal(
    TPMS_NV_CERTIFY_INFO const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ST_ATTEST_Marshal(
    TPMI_ST_ATTEST src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ATTEST_Marshal(
    TPMU_ATTEST const *src,
    uint32_t           selector,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ATTEST_Marshal(
    TPMS_ATTEST const *src,
    uint8_t            buffer[],
```

```
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ATTEST_Marshal(
    TPM2B_ATTEST const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_AUTH_COMMAND_Marshal(
    TPMS_AUTH_COMMAND const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_AUTH_RESPONSE_Marshal(
    TPMS_AUTH_RESPONSE const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_AES_KEY_BITS_Marshal(
    TPMI_AES_KEY_BITS src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SM4_KEY_BITS_Marshal(
    TPMI_SM4_KEY_BITS src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_CAMELLIA_KEY_BITS_Marshal(
    TPMI_CAMELLIA_KEY_BITS src,
    uint8_t                buffer[],
    size_t                 buffer_size,
```

```
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SYM_KEY_BITS_Marshal(
    TPMU_SYM_KEY_BITS const *src,
    uint32_t                 selector,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SYM_MODE_Marshal(
    TPMU_SYM_MODE const *src,
    uint32_t             selector,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SYM_DEF_Marshal(
    TPMT_SYM_DEF const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SYM_DEF_OBJECT_Marshal(
    TPMT_SYM_DEF_OBJECT const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SYM_KEY_Marshal(
    TPM2B_SYM_KEY const *src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SYMCIPHER_PARMS_Marshal(
    TPMS_SYMCIPHER_PARMS const *src,
    uint8_t                    buffer[],
```

```
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_LABEL_Marshal(
    TPM2B_LABEL const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_DERIVE_Marshal(
    TPMS_DERIVE const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DERIVE_Marshal(
    TPM2B_DERIVE const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SENSITIVE_CREATE_Marshal(
    TPMU_SENSITIVE_CREATE const *src,
    uint32_t                     selector,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_DATA_Marshal(
    TPM2B_SENSITIVE_DATA const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SENSITIVE_CREATE_Marshal(
    TPMS_SENSITIVE_CREATE const *src,
    uint8_t                      buffer[],
```

```
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_CREATE_Marshal(
    TPM2B_SENSITIVE_CREATE const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_HASH_Marshal(
    TPMS_SCHEME_HASH const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_ECDAA_Marshal(
    TPMS_SCHEME_ECDAA const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_KEYEDHASH_SCHEME_Marshal(
    TPMI_ALG_KEYEDHASH_SCHEME src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_HMAC_Marshal(
    TPMS_SCHEME_HMAC const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_XOR_Marshal(
    TPMS_SCHEME_XOR const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
```

```
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SCHEME_KEYEDHASH_Marshal(
    TPMU_SCHEME_KEYEDHASH const *src,
    uint32_t                     selector,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_KEYEDHASH_SCHEME_Marshal(
    TPMT_KEYEDHASH_SCHEME const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_RSASSA_Marshal(
    TPMS_SIG_SCHEME_RSASSA const *src,
    uint8_t                       buffer[],
    size_t                        buffer_size,
    size_t                       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_RSAPSS_Marshal(
    TPMS_SIG_SCHEME_RSAPSS const *src,
    uint8_t                       buffer[],
    size_t                        buffer_size,
    size_t                       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECDSA_Marshal(
    TPMS_SIG_SCHEME_ECDSA const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_SM2_Marshal(
    TPMS_SIG_SCHEME_SM2 const *src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
```

```
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECSCHNORR_Marshal(
    TPMS_SIG_SCHEME_ECSCHNORR const *src,
    uint8_t                          buffer[],
    size_t                           buffer_size,
    size_t                          *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECDAA_Marshal(
    TPMS_SIG_SCHEME_ECDAA const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SIG_SCHEME_Marshal(
    TPMU_SIG_SCHEME const *src,
    uint32_t               selector,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SIG_SCHEME_Marshal(
    TPMT_SIG_SCHEME const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ENC_SCHEME_OAEP_Marshal(
    TPMS_ENC_SCHEME_OAEP const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ENC_SCHEME_RSAES_Marshal(
    TPMS_ENC_SCHEME_RSAES const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
```

```
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEY_SCHEME_ECDH_Marshal(
    TPMS_KEY_SCHEME_ECDH const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEY_SCHEME_ECMQV_Marshal(
    TPMS_KEY_SCHEME_ECMQV const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_MGF1_Marshal(
    TPMS_SCHEME_MGF1 const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF1_SP800_56A_Marshal(
    TPMS_SCHEME_KDF1_SP800_56A const *src,
    uint8_t                          buffer[],
    size_t                           buffer_size,
    size_t                          *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF2_Marshal(
    TPMS_SCHEME_KDF2 const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF1_SP800_108_Marshal(
    TPMS_SCHEME_KDF1_SP800_108 const *src,
    uint8_t                          buffer[],
    size_t                           buffer_size,
    size_t                          *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_KDF_SCHEME_Marshal(
    TPMU_KDF_SCHEME const *src,
    uint32_t               selector,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_KDF_SCHEME_Marshal(
    TPMT_KDF_SCHEME const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ASYM_SCHEME_Marshal(
    TPMI_ALG_ASYM_SCHEME src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ASYM_SCHEME_Marshal(
    TPMU_ASYM_SCHEME const *src,
    uint32_t                selector,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_ASYM_SCHEME_Marshal(
    TPMT_ASYM_SCHEME const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_RSA_SCHEME_Marshal(
    TPMI_ALG_RSA_SCHEME src,
    uint8_t             buffer[],
    size_t              buffer_size,
```

```
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_RSA_SCHEME_Marshal(
    TPMT_RSA_SCHEME const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_RSA_DECRYPT_Marshal(
    TPMI_ALG_RSA_DECRYPT src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_RSA_DECRYPT_Marshal(
    TPMT_RSA_DECRYPT const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PUBLIC_KEY_RSA_Marshal(
    TPM2B_PUBLIC_KEY_RSA const *src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RSA_KEY_BITS_Marshal(
    TPMI_RSA_KEY_BITS src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_KEY_RSA_Marshal(
    TPM2B_PRIVATE_KEY_RSA const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ECC_PARAMETER_Marshal(
    TPM2B_ECC_PARAMETER const *src,
    uint8_t                    buffer[],
    size_t                     buffer_size,
    size_t                    *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ECC_POINT_Marshal(
    TPMS_ECC_POINT const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ECC_POINT_Marshal(
    TPM2B_ECC_POINT const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ECC_SCHEME_Marshal(
    TPMI_ALG_ECC_SCHEME src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ECC_CURVE_Marshal(
    TPMI_ECC_CURVE src,
    uint8_t        buffer[],
    size_t         buffer_size,
    size_t        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_ECC_SCHEME_Marshal(
    TPMT_ECC_SCHEME const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALGORITHM_DETAIL_ECC_Marshal(
    TPMS_ALGORITHM_DETAIL_ECC const *src,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSA_Marshal(
    TPMS_SIGNATURE_RSA const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSASSA_Marshal(
    TPMS_SIGNATURE_RSASSA const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSAPSS_Marshal(
    TPMS_SIGNATURE_RSAPSS const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECC_Marshal(
    TPMS_SIGNATURE_ECC const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECDSA_Marshal(
    TPMS_SIGNATURE_ECDSA const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECDAA_Marshal(
```

```
    TPMS_SIGNATURE_ECDAA const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_SM2_Marshal(
    TPMS_SIGNATURE_SM2 const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECSCHNORR_Marshal(
    TPMS_SIGNATURE_ECSCHNORR const *src,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SIGNATURE_Marshal(
    TPMU_SIGNATURE const *src,
    uint32_t              selector,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SIGNATURE_Marshal(
    TPMT_SIGNATURE const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ENCRYPTED_SECRET_Marshal(
    TPMU_ENCRYPTED_SECRET const *src,
    uint32_t                     selector,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ENCRYPTED_SECRET_Marshal(
    TPM2B_ENCRYPTED_SECRET const *src,
    uint8_t                      buffer[],
    size_t                       buffer_size,
    size_t                      *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_PUBLIC_Marshal(
    TPMI_ALG_PUBLIC src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_PUBLIC_ID_Marshal(
    TPMU_PUBLIC_ID const *src,
    uint32_t             selector,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEYEDHASH_PARMS_Marshal(
    TPMS_KEYEDHASH_PARMS const *src,
    uint8_t                     buffer[],
    size_t                      buffer_size,
    size_t                     *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ASYM_PARMS_Marshal(
    TPMS_ASYM_PARMS const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_RSA_PARMS_Marshal(
    TPMS_RSA_PARMS const *src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ECC_PARMS_Marshal(
    TPMS_ECC_PARMS const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_PUBLIC_PARMS_Marshal(
    TPMU_PUBLIC_PARMS const *src,
    uint32_t                selector,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_PUBLIC_PARMS_Marshal(
    TPMT_PUBLIC_PARMS const *src,
    uint8_t                 buffer[],
    size_t                  buffer_size,
    size_t                 *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_PUBLIC_Marshal(
    TPMT_PUBLIC const *src,
    uint8_t           buffer[],
    size_t            buffer_size,
    size_t           *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PUBLIC_Marshal(
    TPM2B_PUBLIC const *src,
    uint8_t            buffer[],
    size_t             buffer_size,
    size_t            *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_TEMPLATE_Marshal(
    TPM2B_TEMPLATE const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_VENDOR_SPECIFIC_Marshal(
    TPM2B_PRIVATE_VENDOR_SPECIFIC const *src,
    uint8_t                              buffer[],
    size_t                               buffer_size,
    size_t                              *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SENSITIVE_COMPOSITE_Marshal(
    TPMU_SENSITIVE_COMPOSITE const *src,
    uint32_t                        selector,
    uint8_t                         buffer[],
    size_t                          buffer_size,
    size_t                         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SENSITIVE_Marshal(
    TPMT_SENSITIVE const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_Marshal(
    TPM2B_SENSITIVE const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU__PRIVATE_Marshal(
    _PRIVATE const *src,
    uint8_t         buffer[],
    size_t          buffer_size,
    size_t         *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_Marshal(
    TPM2B_PRIVATE const *src,
    uint8_t              buffer[],
    size_t               buffer_size,
    size_t              *offset);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ID_OBJECT_Marshal(
    TPMS_ID_OBJECT const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ID_OBJECT_Marshal(
    TPM2B_ID_OBJECT const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_NV_INDEX_Marshal(
    TPM2_NV_INDEX src,
    uint8_t       buffer[],
    size_t        buffer_size,
    size_t       *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_NT_Marshal(
    TPM2_NT src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_PIN_COUNTER_PARAMETERS_Marshal(
    TPMS_NV_PIN_COUNTER_PARAMETERS const *src,
    uint8_t                               buffer[],
    size_t                                buffer_size,
    size_t                               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_NV_Marshal(
    TPMA_NV src,
    uint8_t buffer[],
    size_t  buffer_size,
    size_t *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_PUBLIC_Marshal(
```

```
    TPMS_NV_PUBLIC const *src,
    uint8_t               buffer[],
    size_t                buffer_size,
    size_t               *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NV_PUBLIC_Marshal(
    TPM2B_NV_PUBLIC const *src,
    uint8_t                buffer[],
    size_t                 buffer_size,
    size_t                *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CONTEXT_SENSITIVE_Marshal(
    TPM2B_CONTEXT_SENSITIVE const *src,
    uint8_t                        buffer[],
    size_t                         buffer_size,
    size_t                        *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CONTEXT_DATA_Marshal(
    TPMS_CONTEXT_DATA const *src,
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CONTEXT_DATA_Marshal(
    TPM2B_CONTEXT_DATA const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CONTEXT_Marshal(
    TPMS_CONTEXT const *src,
    uint8_t             buffer[],
    size_t              buffer_size,
    size_t             *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CREATION_DATA_Marshal(
    TPMS_CREATION_DATA const *src,
```

```
    uint8_t                  buffer[],
    size_t                   buffer_size,
    size_t                  *offset);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CREATION_DATA_Marshal(
    TPM2B_CREATION_DATA const *src,
    uint8_t                   buffer[],
    size_t                    buffer_size,
    size_t                   *offset);
```

4.3 tss2_mu.h Unmarshaling Functions

```
/*
 * Unmarshaling Functions
 */
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT8_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    INT8         *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT8_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    UINT8        *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT16_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    INT16        *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT16_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
```

```
    size_t       *offset,
    UINT16       *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT32_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    INT32        *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT32_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    UINT32       *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_INT64_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    INT64        *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_UINT64_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    UINT64       *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_BYTE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    BYTE         *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ALGORITHM_ID_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
```

```
    TPM2_ALGORITHM_ID *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_MODIFIER_INDICATOR_Unmarshal(
    uint8_t            const buffer[],
    size_t                   buffer_size,
    size_t                  *offset,
    TPM2_MODIFIER_INDICATOR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_AUTHORIZATION_SIZE_Unmarshal(
    uint8_t            const buffer[],
    size_t                   buffer_size,
    size_t                  *offset,
    TPM2_AUTHORIZATION_SIZE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PARAMETER_SIZE_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPM2_PARAMETER_SIZE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_KEY_SIZE_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2_KEY_SIZE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_KEY_BITS_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SPEC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_SPEC    *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_GENERATED_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPM2_GENERATED *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ALG_ID_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_ALG_ID  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ECC_CURVE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPM2_ECC_CURVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_CC      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_RC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_RC      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CLOCK_ADJUST_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPM2_CLOCK_ADJUST *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_EO_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_EO      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_ST_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_ST      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SU_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_SU      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_SE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_SE      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_CAP_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_CAP     *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_PT      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PT_PCR_Unmarshal(
```

```
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_PT_PCR  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_PS_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_PS      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HANDLE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_HANDLE  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_HT      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_RH_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_RH      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_HC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_HC      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_ALGORITHM_Unmarshal(
    uint8_t   const buffer[],
```

```
    size_t          buffer_size,
    size_t         *offset,
    TPMA_ALGORITHM *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_OBJECT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMA_OBJECT  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_SESSION_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMA_SESSION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_LOCALITY_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPMA_LOCALITY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_PERMANENT_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMA_PERMANENT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_STARTUP_CLEAR_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMA_STARTUP_CLEAR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_MEMORY_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
```

```
    size_t       *offset,
    TPMA_MEMORY  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_CC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMA_CC      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_MODES_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMA_MODES   *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_YES_NO_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMI_YES_NO  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_OBJECT_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_DH_OBJECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PARENT_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_DH_PARENT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PERSISTENT_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
```

```
    TPMI_DH_PERSISTENT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_ENTITY_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_DH_ENTITY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_PCR_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMI_DH_PCR  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_AUTH_SESSION_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMI_SH_AUTH_SESSION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_HMAC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMI_SH_HMAC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SH_POLICY_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_SH_POLICY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_DH_CONTEXT_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMI_DH_CONTEXT *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_HIERARCHY_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMI_RH_HIERARCHY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_ENABLES_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMI_RH_ENABLES *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_HIERARCHY_AUTH_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPMI_RH_HIERARCHY_AUTH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_PLATFORM_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMI_RH_PLATFORM *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_OWNER_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPMI_RH_OWNER *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_ENDORSEMENT_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMI_RH_ENDORSEMENT *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_PROVISION_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMI_RH_PROVISION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_CLEAR_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPMI_RH_CLEAR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_NV_AUTH_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMI_RH_NV_AUTH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_LOCKOUT_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMI_RH_LOCKOUT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RH_NV_INDEX_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMI_RH_NV_INDEX *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_HASH_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPMI_ALG_HASH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ASYM_Unmarshal(
```

```
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPMI_ALG_ASYM *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMI_ALG_SYM *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_OBJECT_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMI_ALG_SYM_OBJECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SYM_MODE_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMI_ALG_SYM_MODE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_KDF_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMI_ALG_KDF *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_SIG_SCHEME_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMI_ALG_SIG_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ECC_KEY_EXCHANGE_Unmarshal(
    uint8_t          const buffer[],
```

```
    size_t                 buffer_size,
    size_t                *offset,
    TPMI_ECC_KEY_EXCHANGE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ST_COMMAND_TAG_Unmarshal(
    uint8_t        const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMI_ST_COMMAND_TAG *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_EMPTY_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMS_EMPTY   *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALGORITHM_DESCRIPTION_Unmarshal(
    uint8_t               const buffer[],
    size_t                      buffer_size,
    size_t                     *offset,
    TPMS_ALGORITHM_DESCRIPTION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_HA_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    uint32_t      selector,
    TPMU_HA      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_HA_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMT_HA      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DIGEST_Unmarshal(
    uint8_t const buffer[],
```

```
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_DIGEST *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DATA_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_DATA   *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NONCE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_NONCE  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_AUTH_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_AUTH   *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_OPERAND_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2B_OPERAND *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_EVENT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_EVENT  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_MAX_BUFFER_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
```

```
    size_t           *offset,
    TPM2B_MAX_BUFFER *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_MAX_NV_BUFFER_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPM2B_MAX_NV_BUFFER *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_TIMEOUT_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2B_TIMEOUT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_IV_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_IV     *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_NAME_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    uint32_t      selector,
    TPMU_NAME    *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NAME_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_NAME   *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_PCR_SELECT_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
```

```
    size_t          *offset,
    TPMS_PCR_SELECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_PCR_SELECTION_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_PCR_SELECTION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_CREATION_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMT_TK_CREATION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_VERIFIED_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMT_TK_VERIFIED *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_AUTH_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMT_TK_AUTH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_TK_HASHCHECK_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMT_TK_HASHCHECK *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALG_PROPERTY_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
```

```
    TPMS_ALG_PROPERTY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_PROPERTY_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_TAGGED_PROPERTY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_PCR_SELECT_Unmarshal(
    uint8_t           const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_TAGGED_PCR_SELECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TAGGED_POLICY_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_TAGGED_POLICY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_CC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPML_CC      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_CCA_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPML_CCA     *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ALG_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPML_ALG     *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_HANDLE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPML_HANDLE  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_DIGEST_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPML_DIGEST  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_DIGEST_VALUES_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPML_DIGEST_VALUES *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_PCR_SELECTION_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPML_PCR_SELECTION *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ALG_PROPERTY_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPML_ALG_PROPERTY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_TPM_PROPERTY_Unmarshal(
    uint8_t             const buffer[],
    size_t                    buffer_size,
    size_t                   *offset,
    TPML_TAGGED_TPM_PROPERTY *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_PCR_PROPERTY_Unmarshal(
    uint8_t             const buffer[],
    size_t                    buffer_size,
    size_t                   *offset,
    TPML_TAGGED_PCR_PROPERTY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_ECC_CURVE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPML_ECC_CURVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPML_TAGGED_POLICY_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPML_TAGGED_POLICY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_CAPABILITIES_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    uint32_t           selector,
    TPMU_CAPABILITIES *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CAPABILITY_DATA_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_CAPABILITY_DATA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CLOCK_INFO_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMS_CLOCK_INFO *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TIME_INFO_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMS_TIME_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_TIME_ATTEST_INFO_Unmarshal(
    uint8_t          const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_TIME_ATTEST_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CERTIFY_INFO_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMS_CERTIFY_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_QUOTE_INFO_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMS_QUOTE_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_COMMAND_AUDIT_INFO_Unmarshal(
    uint8_t            const buffer[],
    size_t                   buffer_size,
    size_t                  *offset,
    TPMS_COMMAND_AUDIT_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SESSION_AUDIT_INFO_Unmarshal(
    uint8_t            const buffer[],
    size_t                   buffer_size,
    size_t                  *offset,
    TPMS_SESSION_AUDIT_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CREATION_INFO_Unmarshal(
```

```
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_CREATION_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_CERTIFY_INFO_Unmarshal(
    uint8_t         const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMS_NV_CERTIFY_INFO *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ST_ATTEST_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_ST_ATTEST *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ATTEST_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    uint32_t      selector,
    TPMU_ATTEST  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ATTEST_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMS_ATTEST  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ATTEST_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_ATTEST *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_AUTH_COMMAND_Unmarshal(
```

```
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMS_AUTH_COMMAND *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_AUTH_RESPONSE_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_AUTH_RESPONSE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_AES_KEY_BITS_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMI_AES_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_SM4_KEY_BITS_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMI_SM4_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_CAMELLIA_KEY_BITS_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPMI_CAMELLIA_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SYM_KEY_BITS_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    uint32_t           selector,
    TPMU_SYM_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SYM_MODE_Unmarshal(
```

```
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    uint32_t       selector,
    TPMU_SYM_MODE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SYM_DEF_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMT_SYM_DEF *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SYM_DEF_OBJECT_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMT_SYM_DEF_OBJECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SYM_KEY_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2B_SYM_KEY *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SYMCIPHER_PARMS_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_SYMCIPHER_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_LABEL_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_LABEL  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_DERIVE_Unmarshal(
```

```
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMS_DERIVE  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_DERIVE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_DERIVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SENSITIVE_CREATE_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    uint32_t               selector,
    TPMU_SENSITIVE_CREATE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_DATA_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPM2B_SENSITIVE_DATA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SENSITIVE_CREATE_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_SENSITIVE_CREATE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_CREATE_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPM2B_SENSITIVE_CREATE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_HASH_Unmarshal(
```

```
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMS_SCHEME_HASH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_ECDAA_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMS_SCHEME_ECDAA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_KEYEDHASH_SCHEME_Unmarshal(
    uint8_t              const buffer[],
    size_t                     buffer_size,
    size_t                    *offset,
    TPMI_ALG_KEYEDHASH_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_HMAC_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMS_SCHEME_HMAC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_XOR_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMS_SCHEME_XOR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SCHEME_KEYEDHASH_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    uint32_t               selector,
    TPMU_SCHEME_KEYEDHASH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_KEYEDHASH_SCHEME_Unmarshal(
```

```
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPMT_KEYEDHASH_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_RSASSA_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPMS_SIG_SCHEME_RSASSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_RSAPSS_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPMS_SIG_SCHEME_RSAPSS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECDSA_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_SIG_SCHEME_ECDSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_SM2_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMS_SIG_SCHEME_SM2 *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECSCHNORR_Unmarshal(
    uint8_t              const buffer[],
    size_t                     buffer_size,
    size_t                    *offset,
    TPMS_SIG_SCHEME_ECSCHNORR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIG_SCHEME_ECDAA_Unmarshal(
    uint8_t          const buffer[],
```

```
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_SIG_SCHEME_ECDAA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SIG_SCHEME_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    uint32_t         selector,
    TPMU_SIG_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SIG_SCHEME_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMT_SIG_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ENC_SCHEME_OAEP_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_ENC_SCHEME_OAEP *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ENC_SCHEME_RSAES_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_ENC_SCHEME_RSAES *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEY_SCHEME_ECDH_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_KEY_SCHEME_ECDH *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEY_SCHEME_ECMQV_Unmarshal(
    uint8_t          const buffer[],
```

```
    size_t                 buffer_size,
    size_t                *offset,
    TPMS_KEY_SCHEME_ECMQV *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_MGF1_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMS_SCHEME_MGF1 *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF1_SP800_56A_Unmarshal(
    uint8_t               const buffer[],
    size_t                      buffer_size,
    size_t                     *offset,
    TPMS_SCHEME_KDF1_SP800_56A *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF2_Unmarshal(
    uint8_t     const buffer[],
    size_t            buffer_size,
    size_t           *offset,
    TPMS_SCHEME_KDF2 *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SCHEME_KDF1_SP800_108_Unmarshal(
    uint8_t               const buffer[],
    size_t                      buffer_size,
    size_t                     *offset,
    TPMS_SCHEME_KDF1_SP800_108 *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_KDF_SCHEME_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    uint32_t         selector,
    TPMU_KDF_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_KDF_SCHEME_Unmarshal(
    uint8_t    const buffer[],
```

```
    size_t           buffer_size,
    size_t          *offset,
    TPMT_KDF_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ASYM_SCHEME_Unmarshal(
    uint8_t         const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMI_ALG_ASYM_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ASYM_SCHEME_Unmarshal(
    uint8_t     const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    uint32_t         selector,
    TPMU_ASYM_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_ASYM_SCHEME_Unmarshal(
    uint8_t     const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMT_ASYM_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_RSA_SCHEME_Unmarshal(
    uint8_t        const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMI_ALG_RSA_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_RSA_SCHEME_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMT_RSA_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_RSA_DECRYPT_Unmarshal(
    uint8_t         const buffer[],
```

```
    size_t                buffer_size,
    size_t               *offset,
    TPMI_ALG_RSA_DECRYPT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_RSA_DECRYPT_Unmarshal(
    uint8_t     const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMT_RSA_DECRYPT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PUBLIC_KEY_RSA_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPM2B_PUBLIC_KEY_RSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_RSA_KEY_BITS_Unmarshal(
    uint8_t      const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMI_RSA_KEY_BITS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_KEY_RSA_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    TPM2B_PRIVATE_KEY_RSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ECC_PARAMETER_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPM2B_ECC_PARAMETER *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ECC_POINT_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
```

```
    size_t         *offset,
    TPMS_ECC_POINT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ECC_POINT_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPM2B_ECC_POINT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_ECC_SCHEME_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPMI_ALG_ECC_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ECC_CURVE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMI_ECC_CURVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_ECC_SCHEME_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMT_ECC_SCHEME *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ALGORITHM_DETAIL_ECC_Unmarshal(
    uint8_t              const buffer[],
    size_t                     buffer_size,
    size_t                    *offset,
    TPMS_ALGORITHM_DETAIL_ECC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSA_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
```

```
    TPMS_SIGNATURE_RSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSASSA_Unmarshal(
    uint8_t          const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_SIGNATURE_RSASSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_RSAPSS_Unmarshal(
    uint8_t          const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_SIGNATURE_RSAPSS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECC_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_SIGNATURE_ECC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECDSA_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_SIGNATURE_ECDSA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECDAA_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_SIGNATURE_ECDAA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_SM2_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_SIGNATURE_SM2 *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_SIGNATURE_ECSCHNORR_Unmarshal(
    uint8_t             const buffer[],
    size_t                    buffer_size,
    size_t                   *offset,
    TPMS_SIGNATURE_ECSCHNORR *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SIGNATURE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    uint32_t        selector,
    TPMU_SIGNATURE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SIGNATURE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMT_SIGNATURE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_ENCRYPTED_SECRET_Unmarshal(
    uint8_t          const buffer[],
    size_t                 buffer_size,
    size_t                *offset,
    uint32_t               selector,
    TPMU_ENCRYPTED_SECRET *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ENCRYPTED_SECRET_Unmarshal(
    uint8_t           const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPM2B_ENCRYPTED_SECRET *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMI_ALG_PUBLIC_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
```

```
    TPMI_ALG_PUBLIC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_PUBLIC_ID_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    uint32_t        selector,
    TPMU_PUBLIC_ID *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_KEYEDHASH_PARMS_Unmarshal(
    uint8_t         const buffer[],
    size_t                buffer_size,
    size_t               *offset,
    TPMS_KEYEDHASH_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ASYM_PARMS_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPMS_ASYM_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_RSA_PARMS_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMS_RSA_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ECC_PARMS_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMS_ECC_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_PUBLIC_PARMS_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
```

```
    uint32_t           selector,
    TPMU_PUBLIC_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_PUBLIC_PARMS_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMT_PUBLIC_PARMS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_PUBLIC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMT_PUBLIC  *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PUBLIC_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2B_PUBLIC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_TEMPLATE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPM2B_TEMPLATE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_VENDOR_SPECIFIC_Unmarshal(
    uint8_t                  const buffer[],
    size_t                        buffer_size,
    size_t                       *offset,
    TPM2B_PRIVATE_VENDOR_SPECIFIC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMU_SENSITIVE_COMPOSITE_Unmarshal(
    uint8_t             const buffer[],
    size_t                    buffer_size,
    size_t                   *offset,
```

```
    uint32_t                 selector,
    TPMU_SENSITIVE_COMPOSITE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMT_SENSITIVE_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMT_SENSITIVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_SENSITIVE_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPM2B_SENSITIVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU__PRIVATE_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    _PRIVATE     *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_PRIVATE_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2B_PRIVATE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_ID_OBJECT_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMS_ID_OBJECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_ID_OBJECT_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
```

```
    TPM2B_ID_OBJECT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_NV_INDEX_Unmarshal(
    uint8_t  const buffer[],
    size_t         buffer_size,
    size_t        *offset,
    TPM2_NV_INDEX *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2_NT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPM2_NT      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_PIN_COUNTER_PARAMETERS_Unmarshal(
    uint8_t                   const buffer[],
    size_t                         buffer_size,
    size_t                        *offset,
    TPMS_NV_PIN_COUNTER_PARAMETERS *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMA_NV_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMA_NV      *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_NV_PUBLIC_Unmarshal(
    uint8_t   const buffer[],
    size_t          buffer_size,
    size_t         *offset,
    TPMS_NV_PUBLIC *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_NV_PUBLIC_Unmarshal(
    uint8_t    const buffer[],
    size_t           buffer_size,
    size_t          *offset,
    TPM2B_NV_PUBLIC *dest);
```

```
TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CONTEXT_SENSITIVE_Unmarshal(
    uint8_t            const buffer[],
    size_t                  buffer_size,
    size_t                 *offset,
    TPM2B_CONTEXT_SENSITIVE *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CONTEXT_DATA_Unmarshal(
    uint8_t      const buffer[],
    size_t             buffer_size,
    size_t            *offset,
    TPMS_CONTEXT_DATA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CONTEXT_DATA_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPM2B_CONTEXT_DATA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CONTEXT_Unmarshal(
    uint8_t const buffer[],
    size_t        buffer_size,
    size_t       *offset,
    TPMS_CONTEXT *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPMS_CREATION_DATA_Unmarshal(
    uint8_t       const buffer[],
    size_t              buffer_size,
    size_t             *offset,
    TPMS_CREATION_DATA *dest);

TSS2_DLL_EXPORT TSS2_RC Tss2_MU_TPM2B_CREATION_DATA_Unmarshal(
    uint8_t        const buffer[],
    size_t               buffer_size,
    size_t              *offset,
    TPM2B_CREATION_DATA *dest);
```

4.4 tss2_mu.h Postlude

```
#ifdef __cplusplus
} /* end extern "C" */
#endif

#endif /* TSS2_MU_H */
```