TCG TSS 2.0 Overview and Common Structures Specification

Version 1.0
Revision 10
September 23, 2021

PUBLISHED

Contact: admin@trustedcomputinggroup.org

TCG PUBLISHED

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 admin@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].

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
- Tadeusz Struk Intel
- William Roberts Intel
- Lee Wilson Security Innovation (OnBoard Security)

Table of Contents

1 Definitions and References

.

1.1 Acronyms

Term or Acronym	Definition
Application Binary Interface (ABI)	The ABI is the byte-wise layout of data types and function parameters in RAM as well as symbol definitions used to communicate between applications, shared objects and the kernel.
Application Programming Interface (API)	The API is the software interface defined by the functions and structures in a high-level programming language used to communicate between layers in the software stack.
Caller	The caller is the software that invokes a function call or that sends a TCTI command to the TAB/RM.
Connection	A “connection” to the TAB/RM corresponds to a TCTI context from the SAPI to the TAB/RM.
ESAPI	TSS 2.0 Enhanced System API. This layer is intended to sit on top of the System API providing enhanced context management and cryptography.
FAPI	TSS 2.0 Feature API. This layer sits above the ESAPI and provides a high-level interface including a policy definition language and key store.
Implementation	An implementation is the source code and binary code that embodies a specification or parts of a specification.
Marshal	To marshal data is to convert data from C structures to marshaled data.
Marshalled Data	Marshaled data is the representation of data used to communicate with the TPM. In order to optimize data communication to and from the TPM, the smallest amount of data possible is sent to the TPM. For instance, if a structure starts with a size field and that field is set to 0, none of the other fields in the structure are sent to the TPM. Another example: if an input structure consists of a union of data structures, the marshalled representation of the union data structure will be the size of just the data structure selected from the union (actually the marshalled version of that structure itself). Also, the marshalled data must be in big-endian format since this is what the TPM expects.
NV	Non-volatile means that data is not lost when the system is powered down.
PCR	Platform Configuration Register (see TPM 2.0 Library Specification)
RM	The “Resource Manager” is software executing on a system with a TPM that ensures that the resources necessary to execute TPM commands are present in the TPM.
SAPI	TSS 2.0 System API. This layer is intended to sit on top of the TCTI providing marshaling/unmarshalling for TPM commands and responses.
TAB	The TPM Access Broker is software executing on a system with a TPM managing concurrent access from multiple applications.
TPM Command Transmission Interface (TCTI)	The TCTI is an IPC abstraction layer used to send commands to and receive responses from the TPM or the TAB/RM.

TPM	Trusted Platform Module
TPM Resource	Data managed by a TPM that can be referenced by a TPM handle. For TPM 2.0, this includes TPM objects (keys and data objects), TPM NV indices, TPM PCRs and TPM reserved handles and hierarchies.
TSS	TPM Software Stack
Unmarshal	To unmarshal data is to convert data from marshalled format to C structures.

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

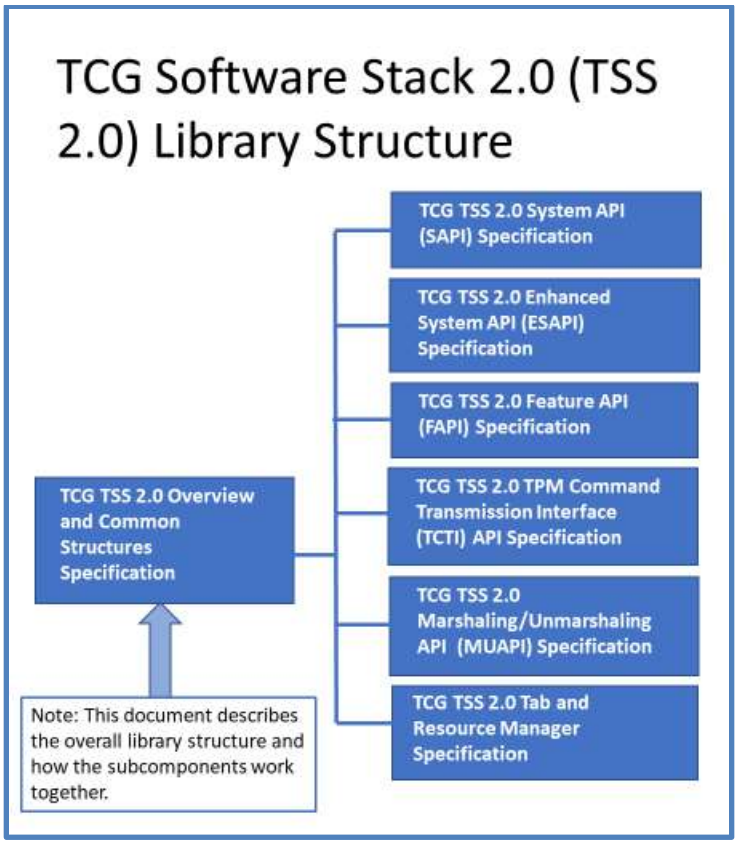

Figure 1: TSS 2.0 Specification Library

1.3 References

Documents change over time. The following rules apply to determining the edition of a reference needed for TSS 2.0. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

The following referenced documents are necessary or very useful for understanding the TPM 2.0 specification.

[1] The Trusted Platform Module Library Specification, Family "2.0"
NOTE: More information, the specification, and other documents can be found at https://trustedcomputinggroup.org/tpm-library-specification/ and http://www.trustedcomputinggroup.org/work-groups/trusted-platform-module/.

[i] Part 1: Architecture
[ii] Part 2: Structures
[iii] Part 3: Commands
[iv] Part 3: Commands – Code
[v] Part 4: Supporting Routines
[vi] Part 4: Supporting Routines – Code

[2] Errata for the Trusted Platform Module Library Specification, Family "2.0"

[i] Errata Version 1.1 for Trusted Platform Module Library Specification, Family "2.0", Revision 01.38
[ii] Errata Version 1.0 for Trusted Platform Module Library Specification, Family "2.0", Revision 01.38
[iii] Errata Version 1.1 for TCG Trusted Platform Module Library Family "2.0" Level 00 Revision 1.59

[3] IETF RFC 3447, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1

[4] NIST SP800-56A, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)*

[5] NIST SP800-108, *Recommendation for Key Derivation Using Pseudorandom Functions (revised)*

[6] FIPS PUB 186-3, *Digital Signature Standard (DSS)*

[7] ISO/IEC 9797-2, Information technology -- Security techniques -- Message Authentication Codes (MACs) -- Part 2: Mechanisms using a dedicated hash-function

[8] IEEE Std 1363™-2000, *Standard Specifications for Public Key Cryptography*

[9] IEEE Std 1363a™-2004 (Amendment to IEEE Std 1363™-2000), IEEE *Standard Specifications for Public Key Cryptography- Amendment 1: Additional Techniques*

[10] ISO/IEC 10116:2006, *Information technology — Security techniques — Modes of operation for an n-*bit *block cipher*

[11] GM/T 0003.1-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 1: General*

[12] GM/T 0003.2-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 2: Digital Signature Algorithm*

[13] GM/T 0003.3-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 3: Key Exchange Protocol*

[14] GM/T 0003.5-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 5: Parameter definition*

[15] GM/T 0004-2012: *SM3 Cryptographic Hash Algorithm*

[16] GM/T 0002-2012: *SM4 Block Cipher Algorithm*

[17] ISO/IEC 10118-3, Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash functions

[18] ISO/IEC 14888-3, Information technology -- Security techniques -- Digital signature with appendix -- Part 3: Discrete logarithm based mechanisms

[19] ISO/IEC 15946-1, Information technology — Security techniques — Cryptographic techniques based on elliptic curves — Part 1: General

[20] ISO/IEC 18033-3, Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers

[21] TCG Algorithm Registry, https://trustedcomputinggroup.org/tcg-algorithm-registry/

[22] TCG TSS 2.0 Overview and Common Structures Specification

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

[24] TCG TSS 2.0 Marshaling/Unmarshaling API Specification

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

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

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

[28] TCG TSS 2.0 TAB and Resource Manager Specification

1.4 TPM Specification Level Addressed by This Specification Version and Revision

This specification defines TSS 2.0 in alignment with Trusted Platform Module Library Specification, Family “2.0”, Level 00, Revision 01.59 – November 2019.

Future changes in the Trusted Platform Module Library Specification may require updates to this document and other documents in the TSS 2.0 library.

2 TSS Overview

The TSS is a software stack designed to isolate TPM application programmers from the low level details of interfacing to the TPM. The TSS consists of multiple layers allowing scalable TSS implementations to be tailored to fit well in high end systems and resource constrained low end systems.

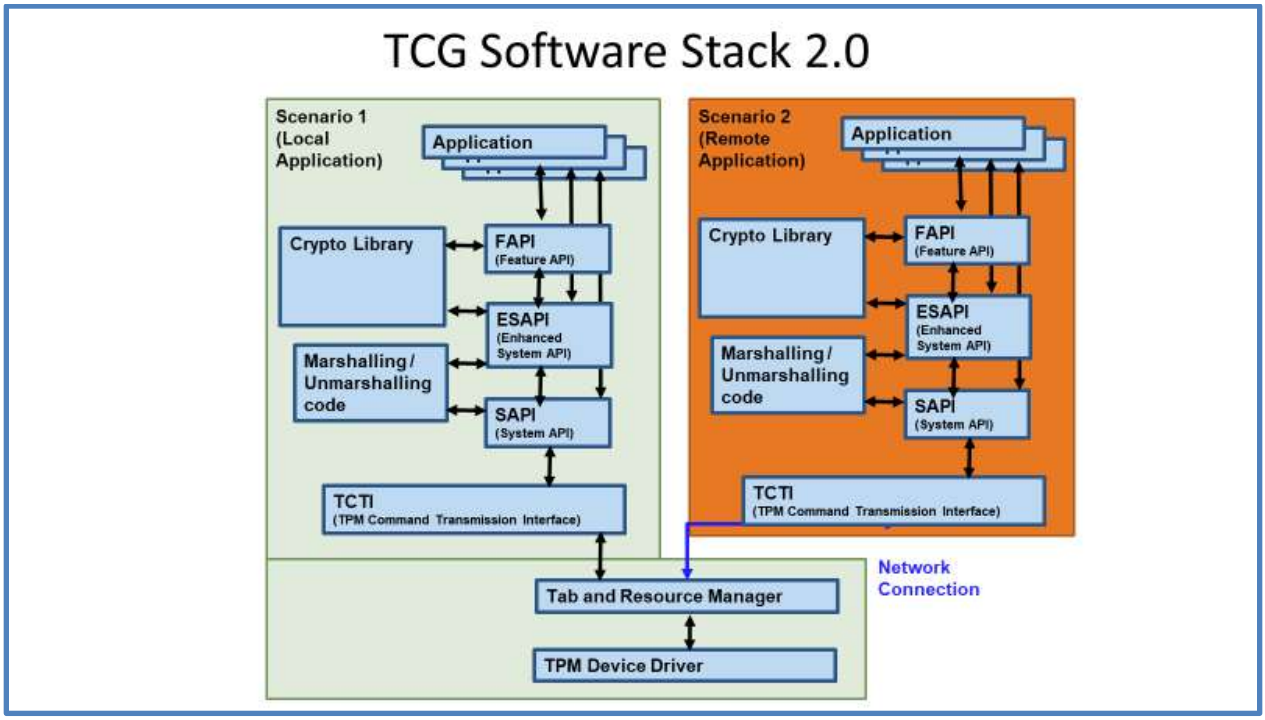

Figure 2: TCG Software Stack 2.0 (TSS 2.0)

The resource requirements for the various components are given in the following figure:

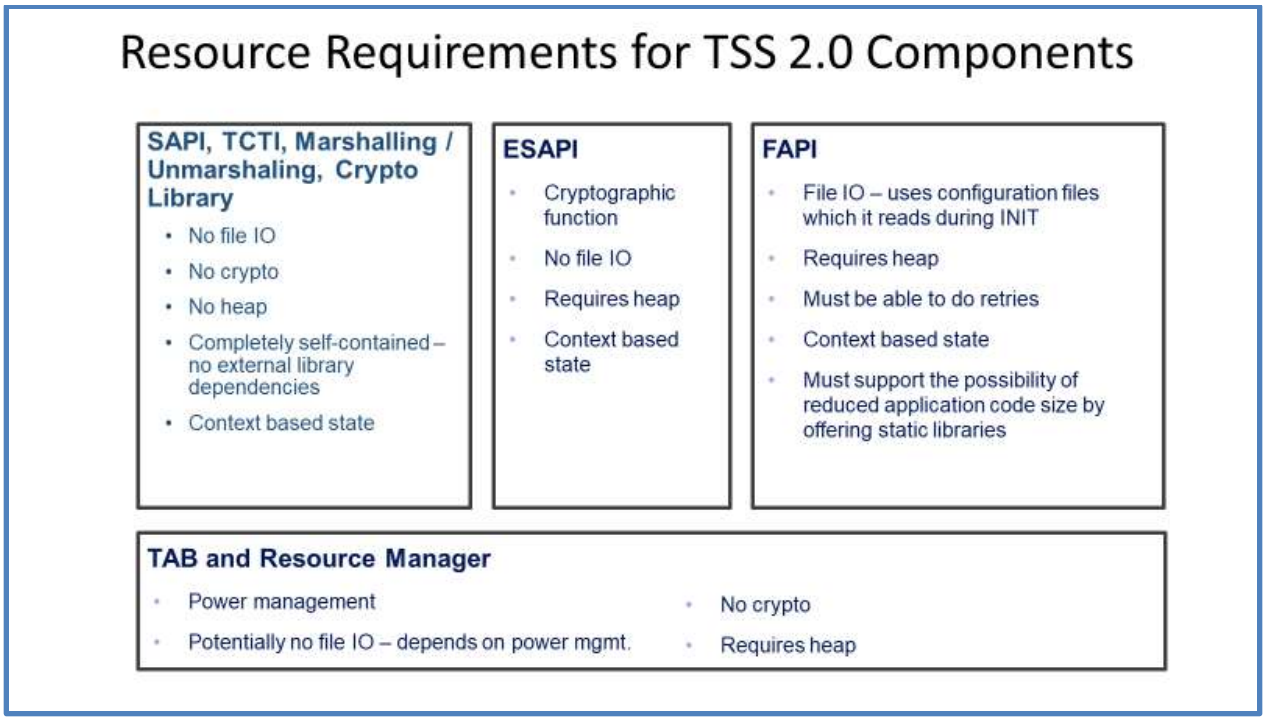

Figure 3: Resource Requirements for TSS 2.0 Components

2.1 TPM Device Driver

The TPM device driver is the OS-specific driver that handles all the handshaking with the TPM and reading and writing of data to the TPM.

2.2 TAB and Resource Manager

The resource manager manages the TPM context in a manner similar to a virtual memory manager. It swaps objects, sessions, and sequences in and out of the limited TPM onboard memory as needed. This layer is transparent to the upper layers of the TSS and is not mandatory. However, if not implemented, the upper layers will be responsible for TPM context management.

The TPM access broker (TAB) handles multi-process synchronization to the TPM. A process accessing the TPM can be guaranteed that it will be able to complete a TPM command without interference from other competing processes.

2.3 TPM Command Transmission Interface (TCTI)

The TPM command transmission interface (TCTI) handles all the communication to and from the lower layers of the stack. For instance, different interfaces are required for local HW TPMs, firmware TPMs, virtual TPMs, remote TPMs, and the TPM simulator. Also, there are two different interfaces to TPMs: the legacy TIS interface and the command/response buffer (CRB).

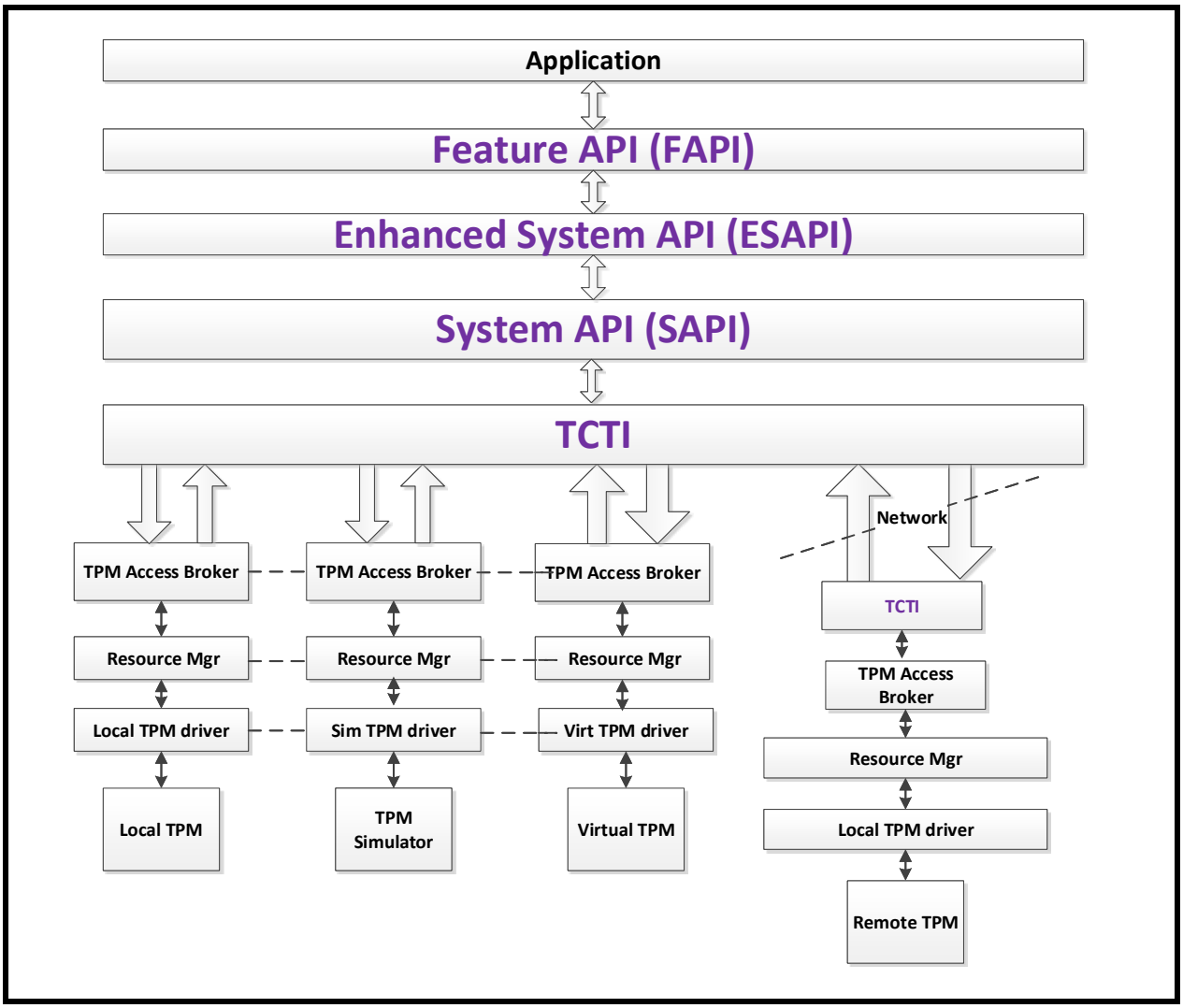

Figure 4: TCTI Allows Connection to Various Target TPMs

2.4 Marshaling/Unmarshaling (MUAPI)

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

2.5 System API (SAPI)

The System API is a layer of the overall TSS architecture that provides access to all the functionality of a TPM 2.0 implementation. It is designed to be used wherever low level calls to the TPM functions are made: firmware, BIOS, applications, OS, etc. The System API, as a low level interface, is targeted towards expert applications.

2.6 Enhanced System API (ESAPI)

The Enhanced System API (ESAPI) is an interface that is intended to sit directly above the System API. The primary purpose of the ESAPI is to reduce the complexity required of applications that desire to send individual “system level” TPM calls to the TPM, but that also require cryptographic operations on the data

being passed to and from the TPM. In particular, applications that wish to utilize secure sessions to perform Hash-based Message Authentication Code (HMAC) operations, parameter encryption, parameter decryption, TPM command audit and TPM policy operations could benefit from using the ESAPI. Additionally, context and object management are provided by the ESAPI.

While the ESAPI is significantly less complex to use than the System API for cryptographically protected communications with the TPM, it still requires in-depth understanding about the interface to a TPM 2.0. It is therefore envisioned that only expert applications will utilize the ESAPI and that typical applications would utilize a higher-level interface such as the Feature API. It is, however, expected that Feature API implementations would utilize the ESAPI as appropriate.

2.7 Feature API (FAPI)

The feature/environment API provides a higher level software abstraction to application developers. For instance, an application may want to create a key without any knowledge of the low level details. This level of abstraction will be provided by the feature and application APIs.

3 Common Header File

tss2_common.h

3.1 tss2_common.h Prelude

```
#ifndef TSS2_COMMON_H
#define TSS2_COMMON_H

#define TSS2_API_VERSION_1_2_1_108

#include <stdint.h>
```

3.2 tss2_common.h DLL Export Macros

While the TCG TSS specifications do not mandate distribution of a TSS2 implementation as either a static or shared library, the provided header files contain macros to facilitate implementation as a shared library on Microsoft Windows™. These macros are not provided for other platforms such as Linux because they are not needed to build shared libraries on unix-like platforms.

When built as a DLL on Microsoft Windows™ the library creator should define the macro `TSS2_DLL_EXPORTS` (with any value) to enable the export of symbols from the DLL. The application developer using the library does not have to define any macros in order to enable the import of symbols from the DLL for use in the application.

When built as a static library on Microsoft Windows™ both the library creator and the application developer must define the macro `TSS2_STATIC`.

On unix-like Oses these macros are not needed and have no effect.

```
/*
 * Macros to export function symbols on Microsoft Windows (TM)
 */
#if !defined(TSS2_DLL_EXPORT)
  #if !defined(WIN32) || defined (TSS2_STATIC)
    #define TSS2_DLL_EXPORT
  #elif defined (TSS2_DLL_EXPORTS)
    #define TSS2_DLL_EXPORT __declspec(dllexport)
  #else
    #define TSS2_DLL_EXPORT __declspec(dllimport)
  #endif
#endif /* end TSS2_DLL_EXPORT */
```

3.3 tss2_common.h Application Binary Interface (ABI) Negotiation

```
/*
 * ABI runtime negotiation definitions
 */
typedef struct TSS2_ABI_VERSION TSS2_ABI_VERSION;
struct TSS2_ABI_VERSION {
    uint32_t tssCreator;
    uint32_t tssFamily;
    uint32_t tssLevel;
    uint32_t tssVersion;
};

#define TSS2_ABI_VERSION_CURRENT {1, 2, 1, 108}
```

3.4 tss2_common.h Common Return Codes

```
/*
 * Return Codes
 */

/* The return type for all TSS2 functions */
typedef uint32_t TSS2_RC;

/* For return values other than SUCCESS, the second most significant
 * byte of the return value is a layer code indicating the software
 * layer that generated the error.
 */
#define TSS2_RC_LAYER_SHIFT      (16)
#define TSS2_RC_LAYER(layer)     ((TSS2_RC)((layer) <<
TSS2_RC_LAYER_SHIFT))
#define TSS2_RC_LAYER_MASK       TSS2_RC_LAYER(0xff)

/* These layer codes are reserved for software layers defined in the
TCG
 * specifications.
 */
#define TSS2_TPM_RC_LAYER            TSS2_RC_LAYER(0) /* base is a
TPM2_RC_* */
#define TSS2_FEATURE_RC_LAYER        TSS2_RC_LAYER(6) /* base is a
TSS2_BASE_RC_* */
#define TSS2_ESYS_RC_LAYER           TSS2_RC_LAYER(7) /* base is a
TSS2_BASE_RC_* */
#define TSS2_SYS_RC_LAYER            TSS2_RC_LAYER(8) /* base is a
TSS2_BASE_RC_* */
#define TSS2_MU_RC_LAYER             TSS2_RC_LAYER(9) /* base is a
TSS2_BASE_RC_* */
#define TSS2_TCTI_RC_LAYER           TSS2_RC_LAYER(10) /* base is a
TSS2_BASE_RC_* */
#define TSS2_RESMGR_RC_LAYER         TSS2_RC_LAYER(11) /* base is a
TSS2_BASE_RC_* */
#define TSS2_RESMGR_TPM_RC_LAYER     TSS2_RC_LAYER(12) /* base is a
TPM_RC_* */
```

3.5 tss2_common.h Base Return Codes

```
/* Base return codes.
 * These base codes indicate the error that occurred. They are
 * logical-ORed with a layer code to produce the TSS2 return value.
*/
#define TSS2_BASE_RC_GENERAL_FAILURE       1U /* Catch all for all
errors not otherwise specifed */
#define TSS2_BASE_RC_NOT_IMPLEMENTED       2U /* If called
functionality isn't implemented */
#define TSS2_BASE_RC_BAD_CONTEXT           3U /* A context structure
is bad */
#define TSS2_BASE_RC_ABI_MISMATCH          4U /* Passed in ABI version
doesn't match called module's ABI version */
#define TSS2_BASE_RC_BAD_REFERENCE         5U /* A pointer is NULL
that isn't allowed to be NULL. */
```

```
#define TSS2_BASE_RC_INSUFFICIENT_BUFFER    6U /* A buffer isn't large
enough */
#define TSS2_BASE_RC_BAD_SEQUENCE           7U /* Function called in
the wrong order */
#define TSS2_BASE_RC_NO_CONNECTION          8U /* Fails to connect to
next lower layer */
#define TSS2_BASE_RC_TRY_AGAIN              9U /* Operation timed out;
function must be called again to be completed */
#define TSS2_BASE_RC_IO_ERROR              10U /* IO failure */
#define TSS2_BASE_RC_BAD_VALUE             11U /* A parameter has a bad
value */
#define TSS2_BASE_RC_NOT_PERMITTED         12U /* Operation not
permitted. */
#define TSS2_BASE_RC_INVALID_SESSIONS      13U /* The TPM command
doesn't use the number of sessions provided by the caller */
#define TSS2_BASE_RC_NO_DECRYPT_PARAM      14U /* A session with its
TPMA_SESSION_DECRYPT bit set was passed to a TPM command that doesn't
support encryption of the first command parameter. */
#define TSS2_BASE_RC_NO_ENCRYPT_PARAM      15U /* A session with its
TPMA_SESSION_ENCRYPT bit set was passed to a TPM command that doesn't
support encryption of the first response parameter. */
#define TSS2_BASE_RC_BAD_SIZE              16U /* If size of a
parameter is incorrect */
#define TSS2_BASE_RC_MALFORMED_RESPONSE    17U /* Response is malformed
*/
#define TSS2_BASE_RC_INSUFFICIENT_CONTEXT  18U /* Context not large
enough */
#define TSS2_BASE_RC_INSUFFICIENT_RESPONSE 19U /* Response is not long
enough */
#define TSS2_BASE_RC_INCOMPATIBLE_TCTI     20U /* Unknown or unusable
TCTI version */
#define TSS2_BASE_RC_NOT_SUPPORTED         21U /* Functionality not
supported. */
#define TSS2_BASE_RC_BAD_TCTI_STRUCTURE    22U /* TCTI context is bad.
*/
#define TSS2_BASE_RC_MEMORY                23U /* memory allocation
failed */
#define TSS2_BASE_RC_BAD_TR                24U /* invalid ESYS_TR
handle */
#define TSS2_BASE_RC_MULTIPLE_DECRYPT_SESSIONS 25U /* More than one
session with TPMA_SESSION_DECRYPT bit set */
#define TSS2_BASE_RC_MULTIPLE_ENCRYPT_SESSIONS 26U /* More than one
session with TPMA_SESSION_ENCRYPT bit set */
#define TSS2_BASE_RC_RSP_AUTH_FAILED           27U
#define TSS2_BASE_RC_NO_CONFIG                 28U
#define TSS2_BASE_RC_BAD_PATH                  29U
#define TSS2_BASE_RC_NOT_DELETABLE             30U
#define TSS2_BASE_RC_PATH_ALREADY_EXISTS       31U
#define TSS2_BASE_RC_KEY_NOT_FOUND             32U
#define TSS2_BASE_RC_SIGNATURE_VERIFICATION_FAILED 33U
#define TSS2_BASE_RC_HASH_MISMATCH             34U
#define TSS2_BASE_RC_KEY_NOT_DUPLICABLE        35U
#define TSS2_BASE_RC_PATH_NOT_FOUND            36U
#define TSS2_BASE_RC_NO_CERT                   37U
#define TSS2_BASE_RC_NO_PCR                    38U
#define TSS2_BASE_RC_PCR_NOT_RESETTABLE        39U
#define TSS2_BASE_RC_BAD_TEMPLATE              40U
```

```
#define TSS2_BASE_RC_AUTHORIZATION_FAILED      41U
#define TSS2_BASE_RC_AUTHORIZATION_UNKNOWN     42U
#define TSS2_BASE_RC_NV_NOT_READABLE           43U
#define TSS2_BASE_RC_NV_TOO_SMALL              44U
#define TSS2_BASE_RC_NV_NOT_WRITEABLE          45U
#define TSS2_BASE_RC_POLICY_UNKNOWN            46U
#define TSS2_BASE_RC_NV_WRONG_TYPE             47U
#define TSS2_BASE_RC_NAME_ALREADY_EXISTS       48U
#define TSS2_BASE_RC_NO_TPM                    49U
#define TSS2_BASE_RC_BAD_KEY                   50U
#define TSS2_BASE_RC_NO_HANDLE                 51U

/* Base return codes in the range 0xf800 - 0xffff are reserved for
 * implementation-specific purposes.
 */
#define TSS2_LAYER_IMPLEMENTATION_SPECIFIC_OFFSET 0xf800

/* Success is the same for all software layers */
#define TSS2_RC_SUCCESS                         ((TSS2_RC) 0)
```

3.6 tss2_common.h TCTI Response Codes

```
/* TCTI response codes */
#define TSS2_TCTI_RC_GENERAL_FAILURE \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_GENERAL_FAILURE))
#define TSS2_TCTI_RC_NOT_IMPLEMENTED
\
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_NOT_IMPLEMENTED))
#define TSS2_TCTI_RC_BAD_CONTEXT
\
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER | TSS2_BASE_RC_BAD_CONTEXT))
#define TSS2_TCTI_RC_ABI_MISMATCH \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_ABI_MISMATCH))
#define TSS2_TCTI_RC_BAD_REFERENCE \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_BAD_REFERENCE))
#define TSS2_TCTI_RC_INSUFFICIENT_BUFFER \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_BUFFER))
#define TSS2_TCTI_RC_BAD_SEQUENCE \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_BAD_SEQUENCE))
#define TSS2_TCTI_RC_NO_CONNECTION \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_NO_CONNECTION))
#define TSS2_TCTI_RC_TRY_AGAIN \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER | TSS2_BASE_RC_TRY_AGAIN))
#define TSS2_TCTI_RC_IO_ERROR \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER | TSS2_BASE_RC_IO_ERROR))
#define TSS2_TCTI_RC_BAD_VALUE \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER | TSS2_BASE_RC_BAD_VALUE))
#define TSS2_TCTI_RC_NOT_PERMITTED \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_NOT_PERMITTED))
```

```
#define TSS2_TCTI_RC_MALFORMED_RESPONSE \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_MALFORMED_RESPONSE))
#define TSS2_TCTI_RC_NOT_SUPPORTED \
            ((TSS2_RC) (TSS2_TCTI_RC_LAYER |
TSS2_BASE_RC_NOT_SUPPORTED))
```

3.7 tss2_common.h SAPI (SYS) Error Codes

```
/* SAPI response codes */
#define TSS2_SYS_RC_GENERAL_FAILURE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_GENERAL_FAILURE))
#define TSS2_SYS_RC_ABI_MISMATCH \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_ABI_MISMATCH))
#define TSS2_SYS_RC_BAD_REFERENCE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_BAD_REFERENCE))
#define TSS2_SYS_RC_INSUFFICIENT_BUFFER \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_BUFFER))
#define TSS2_SYS_RC_BAD_SEQUENCE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_BAD_SEQUENCE))
#define TSS2_SYS_RC_BAD_VALUE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |  TSS2_BASE_RC_BAD_VALUE))
#define TSS2_SYS_RC_INVALID_SESSIONS \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_INVALID_SESSIONS))
#define TSS2_SYS_RC_NO_DECRYPT_PARAM \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_NO_DECRYPT_PARAM))
#define TSS2_SYS_RC_NO_ENCRYPT_PARAM \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_NO_ENCRYPT_PARAM))
#define TSS2_SYS_RC_BAD_SIZE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |  TSS2_BASE_RC_BAD_SIZE))
#define TSS2_SYS_RC_MALFORMED_RESPONSE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_MALFORMED_RESPONSE))
#define TSS2_SYS_RC_INSUFFICIENT_CONTEXT \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_CONTEXT))
#define TSS2_SYS_RC_INSUFFICIENT_RESPONSE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_RESPONSE))
#define TSS2_SYS_RC_INCOMPATIBLE_TCTI \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_INCOMPATIBLE_TCTI))
#define TSS2_SYS_RC_BAD_TCTI_STRUCTURE \
            ((TSS2_RC) (TSS2_SYS_RC_LAYER |
TSS2_BASE_RC_BAD_TCTI_STRUCTURE))
```

3.8 tss2_common.h MUAPI Error Codes

```
/* MUAPI response codes */
```

```
#define TSS2_MU_RC_GENERAL_FAILURE \
            ((TSS2_RC) (TSS2_MU_RC_LAYER |
TSS2_BASE_RC_GENERAL_FAILURE))
#define TSS2_MU_RC_BAD_REFERENCE \
            ((TSS2_RC) (TSS2_MU_RC_LAYER |
TSS2_BASE_RC_BAD_REFERENCE))
#define TSS2_MU_RC_BAD_SIZE \
            ((TSS2_RC) (TSS2_MU_RC_LAYER |  TSS2_BASE_RC_BAD_SIZE))
#define TSS2_MU_RC_BAD_VALUE \
            ((TSS2_RC) (TSS2_MU_RC_LAYER |  TSS2_BASE_RC_BAD_VALUE))
#define TSS2_MU_RC_INSUFFICIENT_BUFFER \
            ((TSS2_RC) (TSS2_MU_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_BUFFER))
```

3.9 tss2_common.h ESAPI Error Codes

```
/* ESAPI response codes */
#define TSS2_ESYS_RC_GENERAL_FAILURE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_GENERAL_FAILURE))
#define TSS2_ESYS_RC_NOT_IMPLEMENTED \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_NOT_IMPLEMENTED))
#define TSS2_ESYS_RC_BAD_CONTEXT \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER | TSS2_BASE_RC_BAD_CONTEXT))
#define TSS2_ESYS_RC_ABI_MISMATCH \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_ABI_MISMATCH))
#define TSS2_ESYS_RC_BAD_REFERENCE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_BAD_REFERENCE))
#define TSS2_ESYS_RC_BAD_SEQUENCE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_BAD_SEQUENCE))
#define TSS2_ESYS_RC_TRY_AGAIN \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER | TSS2_BASE_RC_TRY_AGAIN))
#define TSS2_ESYS_RC_BAD_VALUE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |  TSS2_BASE_RC_BAD_VALUE))
#define TSS2_ESYS_RC_NO_DECRYPT_PARAM \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_NO_DECRYPT_PARAM))
#define TSS2_ESYS_RC_NO_ENCRYPT_PARAM \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_NO_ENCRYPT_PARAM))
#define TSS2_ESYS_RC_MALFORMED_RESPONSE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_MALFORMED_RESPONSE))
#define TSS2_ESYS_RC_INSUFFICIENT_RESPONSE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_INSUFFICIENT_RESPONSE))
#define TSS2_ESYS_RC_INCOMPATIBLE_TCTI \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_INCOMPATIBLE_TCTI))
#define TSS2_ESYS_RC_BAD_TCTI_STRUCTURE \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_BAD_TCTI_STRUCTURE))
#define TSS2_ESYS_RC_MEMORY \
```

```
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |  TSS2_BASE_RC_MEMORY))
#define TSS2_ESYS_RC_BAD_TR \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |  TSS2_BASE_RC_BAD_TR))
#define TSS2_ESYS_RC_MULTIPLE_DECRYPT_SESSIONS \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_MULTIPLE_DECRYPT_SESSIONS))
#define TSS2_ESYS_RC_MULTIPLE_ENCRYPT_SESSIONS \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_MULTIPLE_ENCRYPT_SESSIONS))
#define TSS2_ESYS_RC_NOT_SUPPORTED \
            ((TSS2_RC) (TSS2_ESYS_RC_LAYER |
TSS2_BASE_RC_NOT_SUPPORTED))
```

3.10 tss2_common.h FAPI Error Codes

```
#define TSS2_FAPI_RC_GENERAL_FAILURE \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_GENERAL_FAILURE))
#define TSS2_FAPI_RC_NOT_IMPLEMENTED \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NOT_IMPLEMENTED))
#define TSS2_FAPI_RC_BAD_REFERENCE \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_REFERENCE))
#define TSS2_FAPI_RC_BAD_SEQUENCE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_SEQUENCE))
#define TSS2_FAPI_RC_IO_ERROR \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_IO_ERROR))
#define TSS2_FAPI_RC_BAD_VALUE \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_VALUE))
#define TSS2_FAPI_RC_NO_DECRYPT_PARAM \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_DECRYPT_PARAM))
#define TSS2_FAPI_RC_NO_ENCRYPT_PARAM \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_ENCRYPT_PARAM))
#define TSS2_FAPI_RC_MEMORY \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_MEMORY))
#define TSS2_FAPI_RC_BAD_CONTEXT \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_CONTEXT))
#define  TSS2_FAPI_RC_NO_CONFIG \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_CONFIG))
#define TSS2_FAPI_RC_BAD_PATH \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_PATH))
#define TSS2_FAPI_RC_NOT_DELETABLE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NOT_DELETABLE))
#define TSS2_FAPI_RC_PATH_ALREADY_EXISTS \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_PATH_ALREADY_EXISTS))
#define TSS2_FAPI_RC_KEY_NOT_FOUND \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_KEY_NOT_FOUND))
#define TSS2_FAPI_RC_SIGNATURE_VERIFICATION_FAILED \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER |
TSS2_BASE_RC_SIGNATURE_VERIFICATION_FAILED))
#define TSS2_FAPI_RC_HASH_MISMATCH \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_HASH_MISMATCH))
#define TSS2_FAPI_RC_KEY_NOT_DUPLICABLE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_KEY_NOT_DUPLICABLE))
#define TSS2_FAPI_RC_PATH_NOT_FOUND \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_PATH_NOT_FOUND))
#define TSS2_FAPI_RC_NO_CERT ((TSS2_RC)(TSS2_FEATURE_RC_LAYER |
TSS2_BASE_RC_NO_CERT))
#define TSS2_FAPI_RC_NO_PCR\
```

```
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_PCR))
#define TSS2_FAPI_RC_PCR_NOT_RESETTABLE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_PCR_NOT_RESETTABLE))
#define TSS2_FAPI_RC_BAD_TEMPLATE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_TEMPLATE))
#define TSS2_FAPI_RC_AUTHORIZATION_FAILED \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_AUTHORIZATION_FAILED))
#define TSS2_FAPI_RC_AUTHORIZATION_UNKNOWN \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_AUTHORIZATION_UNKNOWN))
#define TSS2_FAPI_RC_NV_NOT_READABLE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NV_NOT_READABLE))
#define TSS2_FAPI_RC_NV_TOO_SMALL \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NV_TOO_SMALL))
#define TSS2_FAPI_RC_NV_NOT_WRITEABLE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NV_NOT_WRITEABLE))
#define TSS2_FAPI_RC_POLICY_UNKNOWN \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_POLICY_UNKNOWN))
#define TSS2_FAPI_RC_NV_WRONG_TYPE \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NV_WRONG_TYPE))
#define TSS2_FAPI_RC_NAME_ALREADY_EXISTS \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NAME_ALREADY_EXISTS))
#define TSS2_FAPI_RC_NO_TPM \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_TPM))
#define TSS2_FAPI_RC_TRY_AGAIN \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_TRY_AGAIN))
#define TSS2_FAPI_RC_BAD_KEY \
((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_BAD_KEY))
#define TSS2_FAPI_RC_NO_HANDLE \
 ((TSS2_RC)(TSS2_FEATURE_RC_LAYER | TSS2_BASE_RC_NO_HANDLE))
```

3.11 tss2_common.h Postlude

```
#endif /* TSS2_COMMON_H */
```

4 TPM 2 Types Header File

tss2_tpm2_types.h

4.1 tss2_tpm2_types.h Prelude

```
#ifndef TSS2_TPM2_TYPES_H
#define TSS2_TPM2_TYPES_H

#include "tss2_common.h"

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

4.2 tss2_tpm2_types.h ABI Constants

```
/*
 * ABI Constants
 */
```

The following set of ABI constants are the digest sizes of common algorithms.

```
#define TPM2_SHA_DIGEST_SIZE       20
#define TPM2_SHA1_DIGEST_SIZE      20
#define TPM2_SHA256_DIGEST_SIZE    32
#define TPM2_SHA384_DIGEST_SIZE    48
#define TPM2_SHA512_DIGEST_SIZE    64
#define TPM2_SM3_256_DIGEST_SIZE   32
```

The following set of ABI constants were chosen by the TSS Working Group. They represent reasonable, future-proof values.

```
#define TPM2_NUM_PCR_BANKS         16
#define TPM2_MAX_DIGEST_BUFFER   1024
#define TPM2_MAX_NV_BUFFER_SIZE  2048
#define TPM2_MAX_PCRS              32
#define TPM2_MAX_ALG_LIST_SIZE    128
#define TPM2_MAX_CAP_CC           256
#define TPM2_MAX_CAP_BUFFER      1024
#define TPM2_MAX_CONTEXT_SIZE    5120
```

The following set of ABI constants are parameters for cryptographic algorithms. These represent reasonable, future-proof values.

```
#define TPM2_MAX_SYM_BLOCK_SIZE    16
#define TPM2_MAX_SYM_DATA         256
#define TPM2_MAX_ECC_KEY_BYTES    128
#define TPM2_MAX_SYM_KEY_BYTES     32
#define TPM2_MAX_RSA_KEY_BYTES    512
```

The following set of ABI constants derived from the previous values or generic TPM constants.

```
#define TPM2_LABEL_MAX_BUFFER      32
#define TPM2_PCR_SELECT_MAX       ((TPM2_MAX_PCRS+7)/8)
#define TPM2_MAX_CAP_HANDLES      ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPM2_HANDLE))
#define TPM2_MAX_CAP_ALGS         ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPMS_ALG_PROPERTY))
#define TPM2_MAX_TPM_PROPERTIES   ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPMS_TAGGED_PROPERTY))
```

```
#define TPM2_MAX_PCR_PROPERTIES   ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPMS_TAGGED_PCR_SELECT))
#define TPM2_MAX_ECC_CURVES       ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPM2_ECC_CURVE))
#define TPM2_MAX_TAGGED_POLICIES  ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPMS_TAGGED_POLICY))
#define TPM2_MAX_ACT_DATA         ((TPM2_MAX_CAP_BUFFER -
sizeof(TPM2_CAP) - sizeof(UINT32))/sizeof(TPMS_ACT_DATA))
#define TPM2_PRIVATE_VENDOR_SPECIFIC_BYTES  ((TPM2_MAX_RSA_KEY_BYTES/2)
* (3 + 2))
```

4.3 tss2_tpm2_types.h Definition of Types and Associated Constants

```
/*
 * Type & Constant Definitions from the TPM Specification, Part 2
 * Level 00 Revision 01.59.
 */

/* Table 3 - Definition of Base Types */
#if defined(WIN32)
#include <Windows.h>
#else
typedef uint8_t     UINT8;
typedef uint8_t     BYTE;
typedef int8_t      INT8;
typedef uint16_t    UINT16;
typedef int16_t     INT16;
typedef uint32_t    UINT32;
typedef int32_t     INT32;
typedef uint64_t    UINT64;
typedef int64_t     INT64;
#endif

/* Table 5 - Definition of Types for Documentation Clarity */
typedef UINT32  TPM2_ALGORITHM_ID;
typedef UINT32  TPM2_MODIFIER_INDICATOR;
typedef UINT32  TPM2_AUTHORIZATION_SIZE;
typedef UINT32  TPM2_PARAMETER_SIZE;
typedef UINT16  TPM2_KEY_SIZE;
typedef UINT16  TPM2_KEY_BITS;

/* Table 6 - Definition of (UINT32) TPM2_SPEC Constants */
typedef UINT32 TPM2_SPEC;
#define TPM2_SPEC_FAMILY        ((TPM2_SPEC) 0x322E3000)
#define TPM2_SPEC_LEVEL         ((TPM2_SPEC) 00)
#define TPM2_SPEC_VERSION       ((TPM2_SPEC) 159)
#define TPM2_SPEC_YEAR          ((TPM2_SPEC) 2019)
#define TPM2_SPEC_DAY_OF_YEAR   ((TPM2_SPEC) 312)

/* Table 7 - Definition of (UINT32) TPM2_GENERATED Constants */
typedef UINT32 TPM2_GENERATED;
#define TPM2_GENERATED_VALUE    ((TPM2_GENERATED) 0xff544347)

/* Table 9 - Definition of (UINT16) TPM2_ALG_ID Constants */
typedef UINT16 TPM2_ALG_ID;
#define TPM2_ALG_ERROR              ((TPM2_ALG_ID) 0x0000)
```

```
#define TPM2_ALG_RSA                ((TPM2_ALG_ID) 0x0001)
#define TPM2_ALG_TDES               ((TPM2_ALG_ID) 0x0003)
#define TPM2_ALG_SHA                ((TPM2_ALG_ID) 0x0004)
#define TPM2_ALG_SHA1               ((TPM2_ALG_ID) 0x0004)
#define TPM2_ALG_HMAC               ((TPM2_ALG_ID) 0x0005)
#define TPM2_ALG_AES                ((TPM2_ALG_ID) 0x0006)
#define TPM2_ALG_MGF1               ((TPM2_ALG_ID) 0x0007)
#define TPM2_ALG_KEYEDHASH          ((TPM2_ALG_ID) 0x0008)
#define TPM2_ALG_XOR                ((TPM2_ALG_ID) 0x000A)
#define TPM2_ALG_SHA256             ((TPM2_ALG_ID) 0x000B)
#define TPM2_ALG_SHA384             ((TPM2_ALG_ID) 0x000C)
#define TPM2_ALG_SHA512             ((TPM2_ALG_ID) 0x000D)
#define TPM2_ALG_NULL               ((TPM2_ALG_ID) 0x0010)
#define TPM2_ALG_SM3_256            ((TPM2_ALG_ID) 0x0012)
#define TPM2_ALG_SM4                ((TPM2_ALG_ID) 0x0013)
#define TPM2_ALG_RSASSA             ((TPM2_ALG_ID) 0x0014)
#define TPM2_ALG_RSAES              ((TPM2_ALG_ID) 0x0015)
#define TPM2_ALG_RSAPSS             ((TPM2_ALG_ID) 0x0016)
#define TPM2_ALG_OAEP               ((TPM2_ALG_ID) 0x0017)
#define TPM2_ALG_ECDSA              ((TPM2_ALG_ID) 0x0018)
#define TPM2_ALG_ECDH               ((TPM2_ALG_ID) 0x0019)
#define TPM2_ALG_ECDAA              ((TPM2_ALG_ID) 0x001A)
#define TPM2_ALG_SM2                ((TPM2_ALG_ID) 0x001B)
#define TPM2_ALG_ECSCHNORR          ((TPM2_ALG_ID) 0x001C)
#define TPM2_ALG_ECMQV              ((TPM2_ALG_ID) 0x001D)
#define TPM2_ALG_KDF1_SP800_56A     ((TPM2_ALG_ID) 0x0020)
#define TPM2_ALG_KDF2               ((TPM2_ALG_ID) 0x0021)
#define TPM2_ALG_KDF1_SP800_108     ((TPM2_ALG_ID) 0x0022)
#define TPM2_ALG_ECC                ((TPM2_ALG_ID) 0x0023)
#define TPM2_ALG_SYMCIPHER          ((TPM2_ALG_ID) 0x0025)
#define TPM2_ALG_CAMELLIA           ((TPM2_ALG_ID) 0x0026)
#define TPM2_ALG_SHA3_256           ((TPM2_ALG_ID) 0x0027)
#define TPM2_ALG_SHA3_384           ((TPM2_ALG_ID) 0x0028)
#define TPM2_ALG_SHA3_512           ((TPM2_ALG_ID) 0x0029)
#define TPM2_ALG_CTR                ((TPM2_ALG_ID) 0x0040)
#define TPM2_ALG_OFB                ((TPM2_ALG_ID) 0x0041)
#define TPM2_ALG_CBC                ((TPM2_ALG_ID) 0x0042)
#define TPM2_ALG_CFB                ((TPM2_ALG_ID) 0x0043)
#define TPM2_ALG_ECB                ((TPM2_ALG_ID) 0x0044)

/* Table 10 - Definition of (UINT16) { ECC } TPM2_ECC_CURVE Constants
*/
typedef UINT16 TPM2_ECC_CURVE;
#define TPM2_ECC_NONE       ((TPM2_ECC_CURVE) 0x0000)
#define TPM2_ECC_NIST_P192  ((TPM2_ECC_CURVE) 0x0001)
#define TPM2_ECC_NIST_P224  ((TPM2_ECC_CURVE) 0x0002)
#define TPM2_ECC_NIST_P256  ((TPM2_ECC_CURVE) 0x0003)
#define TPM2_ECC_NIST_P384  ((TPM2_ECC_CURVE) 0x0004)
#define TPM2_ECC_NIST_P521  ((TPM2_ECC_CURVE) 0x0005)
#define TPM2_ECC_BN_P256    ((TPM2_ECC_CURVE) 0x0010)
#define TPM2_ECC_BN_P638    ((TPM2_ECC_CURVE) 0x0011)
#define TPM2_ECC_SM2_P256   ((TPM2_ECC_CURVE) 0x0020)

/* Table 12 - Definition of (UINT32) TPM2_CC Constants (Numeric Order)
*/
typedef UINT32 TPM2_CC;
#define TPM2_CC_FIRST                       ((TPM2_CC) 0x0000011F)
```

```
#define TPM2_CC_NV_UndefineSpaceSpecial     ((TPM2_CC) 0x0000011F)
#define TPM2_CC_EvictControl                ((TPM2_CC) 0x00000120)
#define TPM2_CC_HierarchyControl            ((TPM2_CC) 0x00000121)
#define TPM2_CC_NV_UndefineSpace            ((TPM2_CC) 0x00000122)
#define TPM2_CC_ChangeEPS                   ((TPM2_CC) 0x00000124)
#define TPM2_CC_ChangePPS                   ((TPM2_CC) 0x00000125)
#define TPM2_CC_Clear                       ((TPM2_CC) 0x00000126)
#define TPM2_CC_ClearControl                ((TPM2_CC) 0x00000127)
#define TPM2_CC_ClockSet                    ((TPM2_CC) 0x00000128)
#define TPM2_CC_HierarchyChangeAuth         ((TPM2_CC) 0x00000129)
#define TPM2_CC_NV_DefineSpace              ((TPM2_CC) 0x0000012A)
#define TPM2_CC_PCR_Allocate                ((TPM2_CC) 0x0000012B)
#define TPM2_CC_PCR_SetAuthPolicy           ((TPM2_CC) 0x0000012C)
#define TPM2_CC_PP_Commands                 ((TPM2_CC) 0x0000012D)
#define TPM2_CC_SetPrimaryPolicy            ((TPM2_CC) 0x0000012E)
#define TPM2_CC_FieldUpgradeStart           ((TPM2_CC) 0x0000012F)
#define TPM2_CC_ClockRateAdjust             ((TPM2_CC) 0x00000130)
#define TPM2_CC_CreatePrimary               ((TPM2_CC) 0x00000131)
#define TPM2_CC_NV_GlobalWriteLock          ((TPM2_CC) 0x00000132)
#define TPM2_CC_GetCommandAuditDigest       ((TPM2_CC) 0x00000133)
#define TPM2_CC_NV_Increment                ((TPM2_CC) 0x00000134)
#define TPM2_CC_NV_SetBits                  ((TPM2_CC) 0x00000135)
#define TPM2_CC_NV_Extend                   ((TPM2_CC) 0x00000136)
#define TPM2_CC_NV_Write                    ((TPM2_CC) 0x00000137)
#define TPM2_CC_NV_WriteLock                ((TPM2_CC) 0x00000138)
#define TPM2_CC_DictionaryAttackLockReset   ((TPM2_CC) 0x00000139)
#define TPM2_CC_DictionaryAttackParameters  ((TPM2_CC) 0x0000013A)
#define TPM2_CC_NV_ChangeAuth               ((TPM2_CC) 0x0000013B)
#define TPM2_CC_PCR_Event                   ((TPM2_CC) 0x0000013C)
#define TPM2_CC_PCR_Reset                   ((TPM2_CC) 0x0000013D)
#define TPM2_CC_SequenceComplete            ((TPM2_CC) 0x0000013E)
#define TPM2_CC_SetAlgorithmSet             ((TPM2_CC) 0x0000013F)
#define TPM2_CC_SetCommandCodeAuditStatus   ((TPM2_CC) 0x00000140)
#define TPM2_CC_FieldUpgradeData            ((TPM2_CC) 0x00000141)
#define TPM2_CC_IncrementalSelfTest         ((TPM2_CC) 0x00000142)
#define TPM2_CC_SelfTest                    ((TPM2_CC) 0x00000143)
#define TPM2_CC_Startup                     ((TPM2_CC) 0x00000144)
#define TPM2_CC_Shutdown                    ((TPM2_CC) 0x00000145)
#define TPM2_CC_StirRandom                  ((TPM2_CC) 0x00000146)
#define TPM2_CC_ActivateCredential          ((TPM2_CC) 0x00000147)
#define TPM2_CC_Certify                     ((TPM2_CC) 0x00000148)
#define TPM2_CC_PolicyNV                    ((TPM2_CC) 0x00000149)
#define TPM2_CC_CertifyCreation             ((TPM2_CC) 0x0000014A)
#define TPM2_CC_Duplicate                   ((TPM2_CC) 0x0000014B)
#define TPM2_CC_GetTime                     ((TPM2_CC) 0x0000014C)
#define TPM2_CC_GetSessionAuditDigest       ((TPM2_CC) 0x0000014D)
#define TPM2_CC_NV_Read                     ((TPM2_CC) 0x0000014E)
#define TPM2_CC_NV_ReadLock                 ((TPM2_CC) 0x0000014F)
#define TPM2_CC_ObjectChangeAuth            ((TPM2_CC) 0x00000150)
#define TPM2_CC_PolicySecret                ((TPM2_CC) 0x00000151)
#define TPM2_CC_Rewrap                      ((TPM2_CC) 0x00000152)
#define TPM2_CC_Create                      ((TPM2_CC) 0x00000153)
#define TPM2_CC_ECDH_ZGen                   ((TPM2_CC) 0x00000154)
#define TPM2_CC_HMAC                        ((TPM2_CC) 0x00000155)
#define TPM2_CC_MAC                         ((TPM2_CC) 0x00000155)
#define TPM2_CC_Import                      ((TPM2_CC) 0x00000156)
#define TPM2_CC_Load                        ((TPM2_CC) 0x00000157)
```

```
#define TPM2_CC_Quote                        ((TPM2_CC) 0x00000158)
#define TPM2_CC_RSA_Decrypt                  ((TPM2_CC) 0x00000159)
#define TPM2_CC_HMAC_Start                   ((TPM2_CC) 0x0000015B)
#define TPM2_CC_MAC_Start                    ((TPM2_CC) 0x0000015B)
#define TPM2_CC_SequenceUpdate               ((TPM2_CC) 0x0000015C)
#define TPM2_CC_Sign                         ((TPM2_CC) 0x0000015D)
#define TPM2_CC_Unseal                       ((TPM2_CC) 0x0000015E)
#define TPM2_CC_PolicySigned                 ((TPM2_CC) 0x00000160)
#define TPM2_CC_ContextLoad                  ((TPM2_CC) 0x00000161)
#define TPM2_CC_ContextSave                  ((TPM2_CC) 0x00000162)
#define TPM2_CC_ECDH_KeyGen                  ((TPM2_CC) 0x00000163)
#define TPM2_CC_EncryptDecrypt               ((TPM2_CC) 0x00000164)
#define TPM2_CC_FlushContext                 ((TPM2_CC) 0x00000165)
#define TPM2_CC_LoadExternal                 ((TPM2_CC) 0x00000167)
#define TPM2_CC_MakeCredential               ((TPM2_CC) 0x00000168)
#define TPM2_CC_NV_ReadPublic                ((TPM2_CC) 0x00000169)
#define TPM2_CC_PolicyAuthorize              ((TPM2_CC) 0x0000016A)
#define TPM2_CC_PolicyAuthValue              ((TPM2_CC) 0x0000016B)
#define TPM2_CC_PolicyCommandCode            ((TPM2_CC) 0x0000016C)
#define TPM2_CC_PolicyCounterTimer           ((TPM2_CC) 0x0000016D)
#define TPM2_CC_PolicyCpHash                 ((TPM2_CC) 0x0000016E)
#define TPM2_CC_PolicyLocality               ((TPM2_CC) 0x0000016F)
#define TPM2_CC_PolicyNameHash               ((TPM2_CC) 0x00000170)
#define TPM2_CC_PolicyOR                     ((TPM2_CC) 0x00000171)
#define TPM2_CC_PolicyTicket                 ((TPM2_CC) 0x00000172)
#define TPM2_CC_ReadPublic                   ((TPM2_CC) 0x00000173)
#define TPM2_CC_RSA_Encrypt                  ((TPM2_CC) 0x00000174)
#define TPM2_CC_StartAuthSession             ((TPM2_CC) 0x00000176)
#define TPM2_CC_VerifySignature              ((TPM2_CC) 0x00000177)
#define TPM2_CC_ECC_Parameters               ((TPM2_CC) 0x00000178)
#define TPM2_CC_FirmwareRead                 ((TPM2_CC) 0x00000179)
#define TPM2_CC_GetCapability                ((TPM2_CC) 0x0000017A)
#define TPM2_CC_GetRandom                    ((TPM2_CC) 0x0000017B)
#define TPM2_CC_GetTestResult                ((TPM2_CC) 0x0000017C)
#define TPM2_CC_Hash                         ((TPM2_CC) 0x0000017D)
#define TPM2_CC_PCR_Read                     ((TPM2_CC) 0x0000017E)
#define TPM2_CC_PolicyPCR                    ((TPM2_CC) 0x0000017F)
#define TPM2_CC_PolicyRestart                ((TPM2_CC) 0x00000180)
#define TPM2_CC_ReadClock                    ((TPM2_CC) 0x00000181)
#define TPM2_CC_PCR_Extend                   ((TPM2_CC) 0x00000182)
#define TPM2_CC_PCR_SetAuthValue             ((TPM2_CC) 0x00000183)
#define TPM2_CC_NV_Certify                   ((TPM2_CC) 0x00000184)
#define TPM2_CC_EventSequenceComplete        ((TPM2_CC) 0x00000185)
#define TPM2_CC_HashSequenceStart            ((TPM2_CC) 0x00000186)
#define TPM2_CC_PolicyPhysicalPresence       ((TPM2_CC) 0x00000187)
#define TPM2_CC_PolicyDuplicationSelect      ((TPM2_CC) 0x00000188)
#define TPM2_CC_PolicyGetDigest              ((TPM2_CC) 0x00000189)
#define TPM2_CC_TestParms                    ((TPM2_CC) 0x0000018A)
#define TPM2_CC_Commit                       ((TPM2_CC) 0x0000018B)
#define TPM2_CC_PolicyPassword               ((TPM2_CC) 0x0000018C)
#define TPM2_CC_ZGen_2Phase                  ((TPM2_CC) 0x0000018D)
#define TPM2_CC_EC_Ephemeral                 ((TPM2_CC) 0x0000018E)
#define TPM2_CC_PolicyNvWritten              ((TPM2_CC) 0x0000018F)
#define TPM2_CC_PolicyTemplate               ((TPM2_CC) 0x00000190)
#define TPM2_CC_CreateLoaded                 ((TPM2_CC) 0x00000191)
#define TPM2_CC_PolicyAuthorizeNV            ((TPM2_CC) 0x00000192)
#define TPM2_CC_EncryptDecrypt2              ((TPM2_CC) 0x00000193)
```

```
#define TPM2_CC_AC_GetCapability             ((TPM2_CC) 0x00000194)
#define TPM2_CC_AC_Send                      ((TPM2_CC) 0x00000195)
#define TPM2_CC_Policy_AC_SendSelect         ((TPM2_CC) 0x00000196)
#define TPM2_CC_CertifyX509                  ((TPM2_CC) 0x00000197)
#define TPM2_CC_ACT_SetTimeout               ((TPM2_CC) 0x00000198)
#define TPM2_CC_LAST                         ((TPM2_CC) 0x00000198)
#define TPM2_CC_VEND                         ((TPM2_CC) 0x20000000)
#define TPM2_CC_Vendor_TCG_Test              ((TPM2_CC)
(TPM2_CC_VEND+0x0000))

/* Table 16 - Definition of (UINT32) TPM2_RC Constants (Actions) */
typedef UINT32 TPM2_RC;
#define TPM2_RC_SUCCESS              ((TPM2_RC) 0x000)
#define TPM2_RC_BAD_TAG              ((TPM2_RC) 0x01E)
#define TPM2_RC_VER1                 ((TPM2_RC) 0x100)
#define TPM2_RC_INITIALIZE           ((TPM2_RC) (TPM2_RC_VER1 + 0x000))
#define TPM2_RC_FAILURE              ((TPM2_RC) (TPM2_RC_VER1 + 0x001))
#define TPM2_RC_SEQUENCE             ((TPM2_RC) (TPM2_RC_VER1 + 0x003))
#define TPM2_RC_PRIVATE              ((TPM2_RC) (TPM2_RC_VER1 + 0x00B))
#define TPM2_RC_HMAC                 ((TPM2_RC) (TPM2_RC_VER1 + 0x019))
#define TPM2_RC_DISABLED             ((TPM2_RC) (TPM2_RC_VER1 + 0x020))
#define TPM2_RC_EXCLUSIVE            ((TPM2_RC) (TPM2_RC_VER1 + 0x021))
#define TPM2_RC_AUTH_TYPE            ((TPM2_RC) (TPM2_RC_VER1 + 0x024))
#define TPM2_RC_AUTH_MISSING         ((TPM2_RC) (TPM2_RC_VER1 + 0x025))
#define TPM2_RC_POLICY               ((TPM2_RC) (TPM2_RC_VER1 + 0x026))
#define TPM2_RC_PCR                  ((TPM2_RC) (TPM2_RC_VER1 + 0x027))
#define TPM2_RC_PCR_CHANGED          ((TPM2_RC) (TPM2_RC_VER1 + 0x028))
#define TPM2_RC_UPGRADE              ((TPM2_RC) (TPM2_RC_VER1 + 0x02D))
#define TPM2_RC_TOO_MANY_CONTEXTS    ((TPM2_RC) (TPM2_RC_VER1 + 0x02E))
#define TPM2_RC_AUTH_UNAVAILABLE     ((TPM2_RC) (TPM2_RC_VER1 + 0x02F))
#define TPM2_RC_REBOOT               ((TPM2_RC) (TPM2_RC_VER1 + 0x030))
#define TPM2_RC_UNBALANCED           ((TPM2_RC) (TPM2_RC_VER1 + 0x031))
#define TPM2_RC_COMMAND_SIZE         ((TPM2_RC) (TPM2_RC_VER1 + 0x042))
#define TPM2_RC_COMMAND_CODE         ((TPM2_RC) (TPM2_RC_VER1 + 0x043))
#define TPM2_RC_AUTHSIZE             ((TPM2_RC) (TPM2_RC_VER1 + 0x044))
#define TPM2_RC_AUTH_CONTEXT         ((TPM2_RC) (TPM2_RC_VER1 + 0x045))
#define TPM2_RC_NV_RANGE             ((TPM2_RC) (TPM2_RC_VER1 + 0x046))
#define TPM2_RC_NV_SIZE              ((TPM2_RC) (TPM2_RC_VER1 + 0x047))
#define TPM2_RC_NV_LOCKED            ((TPM2_RC) (TPM2_RC_VER1 + 0x048))
#define TPM2_RC_NV_AUTHORIZATION     ((TPM2_RC) (TPM2_RC_VER1 + 0x049))
#define TPM2_RC_NV_UNINITIALIZED     ((TPM2_RC) (TPM2_RC_VER1 + 0x04A))
#define TPM2_RC_NV_SPACE             ((TPM2_RC) (TPM2_RC_VER1 + 0x04B))
#define TPM2_RC_NV_DEFINED           ((TPM2_RC) (TPM2_RC_VER1 + 0x04C))
#define TPM2_RC_BAD_CONTEXT          ((TPM2_RC) (TPM2_RC_VER1 + 0x050))
#define TPM2_RC_CPHASH               ((TPM2_RC) (TPM2_RC_VER1 + 0x051))
#define TPM2_RC_PARENT               ((TPM2_RC) (TPM2_RC_VER1 + 0x052))
#define TPM2_RC_NEEDS_TEST           ((TPM2_RC) (TPM2_RC_VER1 + 0x053))
#define TPM2_RC_NO_RESULT            ((TPM2_RC) (TPM2_RC_VER1 + 0x054))
#define TPM2_RC_SENSITIVE            ((TPM2_RC) (TPM2_RC_VER1 + 0x055))
#define TPM2_RC_MAX_FM0              ((TPM2_RC) (TPM2_RC_VER1 + 0x07F))
#define TPM2_RC_FMT1                 ((TPM2_RC) 0x080)
#define TPM2_RC_ASYMMETRIC           ((TPM2_RC) (TPM2_RC_FMT1 + 0x001))
#define TPM2_RC_ATTRIBUTES           ((TPM2_RC) (TPM2_RC_FMT1 + 0x002))
#define TPM2_RC_HASH                 ((TPM2_RC) (TPM2_RC_FMT1 + 0x003))
#define TPM2_RC_VALUE                ((TPM2_RC) (TPM2_RC_FMT1 + 0x004))
#define TPM2_RC_HIERARCHY            ((TPM2_RC) (TPM2_RC_FMT1 + 0x005))
#define TPM2_RC_KEY_SIZE             ((TPM2_RC) (TPM2_RC_FMT1 + 0x007))
```

```
#define TPM2_RC_MGF                 ((TPM2_RC) (TPM2_RC_FMT1 + 0x008))
#define TPM2_RC_MODE                ((TPM2_RC) (TPM2_RC_FMT1 + 0x009))
#define TPM2_RC_TYPE                ((TPM2_RC) (TPM2_RC_FMT1 + 0x00A))
#define TPM2_RC_HANDLE              ((TPM2_RC) (TPM2_RC_FMT1 + 0x00B))
#define TPM2_RC_KDF                 ((TPM2_RC) (TPM2_RC_FMT1 + 0x00C))
#define TPM2_RC_RANGE               ((TPM2_RC) (TPM2_RC_FMT1 + 0x00D))
#define TPM2_RC_AUTH_FAIL           ((TPM2_RC) (TPM2_RC_FMT1 + 0x00E))
#define TPM2_RC_NONCE               ((TPM2_RC) (TPM2_RC_FMT1 + 0x00F))
#define TPM2_RC_PP                  ((TPM2_RC) (TPM2_RC_FMT1 + 0x010))
#define TPM2_RC_SCHEME              ((TPM2_RC) (TPM2_RC_FMT1 + 0x012))
#define TPM2_RC_SIZE                ((TPM2_RC) (TPM2_RC_FMT1 + 0x015))
#define TPM2_RC_SYMMETRIC           ((TPM2_RC) (TPM2_RC_FMT1 + 0x016))
#define TPM2_RC_TAG                 ((TPM2_RC) (TPM2_RC_FMT1 + 0x017))
#define TPM2_RC_SELECTOR            ((TPM2_RC) (TPM2_RC_FMT1 + 0x018))
#define TPM2_RC_INSUFFICIENT        ((TPM2_RC) (TPM2_RC_FMT1 + 0x01A))
#define TPM2_RC_SIGNATURE           ((TPM2_RC) (TPM2_RC_FMT1 + 0x01B))
#define TPM2_RC_KEY                 ((TPM2_RC) (TPM2_RC_FMT1 + 0x01C))
#define TPM2_RC_POLICY_FAIL         ((TPM2_RC) (TPM2_RC_FMT1 + 0x01D))
#define TPM2_RC_INTEGRITY           ((TPM2_RC) (TPM2_RC_FMT1 + 0x01F))
#define TPM2_RC_TICKET              ((TPM2_RC) (TPM2_RC_FMT1 + 0x020))
#define TPM2_RC_RESERVED_BITS       ((TPM2_RC) (TPM2_RC_FMT1 + 0x021))
#define TPM2_RC_BAD_AUTH            ((TPM2_RC) (TPM2_RC_FMT1 + 0x022))
#define TPM2_RC_EXPIRED             ((TPM2_RC) (TPM2_RC_FMT1 + 0x023))
#define TPM2_RC_POLICY_CC           ((TPM2_RC) (TPM2_RC_FMT1 + 0x024))
#define TPM2_RC_BINDING             ((TPM2_RC) (TPM2_RC_FMT1 + 0x025))
#define TPM2_RC_CURVE               ((TPM2_RC) (TPM2_RC_FMT1 + 0x026))
#define TPM2_RC_ECC_POINT           ((TPM2_RC) (TPM2_RC_FMT1 + 0x027))
#define TPM2_RC_WARN                ((TPM2_RC) 0x900)
#define TPM2_RC_CONTEXT_GAP         ((TPM2_RC) (TPM2_RC_WARN + 0x001))
#define TPM2_RC_OBJECT_MEMORY       ((TPM2_RC) (TPM2_RC_WARN + 0x002))
#define TPM2_RC_SESSION_MEMORY      ((TPM2_RC) (TPM2_RC_WARN + 0x003))
#define TPM2_RC_MEMORY              ((TPM2_RC) (TPM2_RC_WARN + 0x004))
#define TPM2_RC_SESSION_HANDLES     ((TPM2_RC) (TPM2_RC_WARN + 0x005))
#define TPM2_RC_OBJECT_HANDLES      ((TPM2_RC) (TPM2_RC_WARN + 0x006))
#define TPM2_RC_LOCALITY            ((TPM2_RC) (TPM2_RC_WARN + 0x007))
#define TPM2_RC_YIELDED             ((TPM2_RC) (TPM2_RC_WARN + 0x008))
#define TPM2_RC_CANCELED            ((TPM2_RC) (TPM2_RC_WARN + 0x009))
#define TPM2_RC_TESTING             ((TPM2_RC) (TPM2_RC_WARN + 0x00A))
#define TPM2_RC_REFERENCE_H0        ((TPM2_RC) (TPM2_RC_WARN + 0x010))
#define TPM2_RC_REFERENCE_H1        ((TPM2_RC) (TPM2_RC_WARN + 0x011))
#define TPM2_RC_REFERENCE_H2        ((TPM2_RC) (TPM2_RC_WARN + 0x012))
#define TPM2_RC_REFERENCE_H3        ((TPM2_RC) (TPM2_RC_WARN + 0x013))
#define TPM2_RC_REFERENCE_H4        ((TPM2_RC) (TPM2_RC_WARN + 0x014))
#define TPM2_RC_REFERENCE_H5        ((TPM2_RC) (TPM2_RC_WARN + 0x015))
#define TPM2_RC_REFERENCE_H6        ((TPM2_RC) (TPM2_RC_WARN + 0x016))
#define TPM2_RC_REFERENCE_S0        ((TPM2_RC) (TPM2_RC_WARN + 0x018))
#define TPM2_RC_REFERENCE_S1        ((TPM2_RC) (TPM2_RC_WARN + 0x019))
#define TPM2_RC_REFERENCE_S2        ((TPM2_RC) (TPM2_RC_WARN + 0x01A))
#define TPM2_RC_REFERENCE_S3        ((TPM2_RC) (TPM2_RC_WARN + 0x01B))
#define TPM2_RC_REFERENCE_S4        ((TPM2_RC) (TPM2_RC_WARN + 0x01C))
#define TPM2_RC_REFERENCE_S5        ((TPM2_RC) (TPM2_RC_WARN + 0x01D))
#define TPM2_RC_REFERENCE_S6        ((TPM2_RC) (TPM2_RC_WARN + 0x01E))
#define TPM2_RC_NV_RATE             ((TPM2_RC) (TPM2_RC_WARN + 0x020))
#define TPM2_RC_LOCKOUT             ((TPM2_RC) (TPM2_RC_WARN + 0x021))
#define TPM2_RC_RETRY               ((TPM2_RC) (TPM2_RC_WARN + 0x022))
#define TPM2_RC_NV_UNAVAILABLE      ((TPM2_RC) (TPM2_RC_WARN + 0x023))
#define TPM2_RC_NOT_USED            ((TPM2_RC) (TPM2_RC_WARN + 0x7F))
```

```
#define TPM2_RC_H                    ((TPM2_RC) 0x000)
#define TPM2_RC_P                    ((TPM2_RC) 0x040)
#define TPM2_RC_S                    ((TPM2_RC) 0x800)
#define TPM2_RC_1                    ((TPM2_RC) 0x100)
#define TPM2_RC_2                    ((TPM2_RC) 0x200)
#define TPM2_RC_3                    ((TPM2_RC) 0x300)
#define TPM2_RC_4                    ((TPM2_RC) 0x400)
#define TPM2_RC_5                    ((TPM2_RC) 0x500)
#define TPM2_RC_6                    ((TPM2_RC) 0x600)
#define TPM2_RC_7                    ((TPM2_RC) 0x700)
#define TPM2_RC_8                    ((TPM2_RC) 0x800)
#define TPM2_RC_9                    ((TPM2_RC) 0x900)
#define TPM2_RC_A                    ((TPM2_RC) 0xA00)
#define TPM2_RC_B                    ((TPM2_RC) 0xB00)
#define TPM2_RC_C                    ((TPM2_RC) 0xC00)
#define TPM2_RC_D                    ((TPM2_RC) 0xD00)
#define TPM2_RC_E                    ((TPM2_RC) 0xE00)
#define TPM2_RC_F                    ((TPM2_RC) 0xF00)
#define TPM2_RC_N_MASK               ((TPM2_RC) 0xF00)

/* Table 17 - Definition of (INT8) TPM2_CLOCK_ADJUST Constants */
typedef INT8 TPM2_CLOCK_ADJUST;
#define TPM2_CLOCK_COARSE_SLOWER     ((TPM2_CLOCK_ADJUST) -3)
#define TPM2_CLOCK_MEDIUM_SLOWER     ((TPM2_CLOCK_ADJUST) -2)
#define TPM2_CLOCK_FINE_SLOWER       ((TPM2_CLOCK_ADJUST) -1)
#define TPM2_CLOCK_NO_CHANGE         ((TPM2_CLOCK_ADJUST) 0)
#define TPM2_CLOCK_FINE_FASTER       ((TPM2_CLOCK_ADJUST) 1)
#define TPM2_CLOCK_MEDIUM_FASTER     ((TPM2_CLOCK_ADJUST) 2)
#define TPM2_CLOCK_COARSE_FASTER     ((TPM2_CLOCK_ADJUST) 3)

/* Table 18 - Definition of (UINT16) TPM2_EO Constants */
typedef UINT16 TPM2_EO;
#define TPM2_EO_EQ              ((TPM2_EO) 0x0000)
#define TPM2_EO_NEQ             ((TPM2_EO) 0x0001)
#define TPM2_EO_SIGNED_GT       ((TPM2_EO) 0x0002)
#define TPM2_EO_UNSIGNED_GT     ((TPM2_EO) 0x0003)
#define TPM2_EO_SIGNED_LT       ((TPM2_EO) 0x0004)
#define TPM2_EO_UNSIGNED_LT     ((TPM2_EO) 0x0005)
#define TPM2_EO_SIGNED_GE       ((TPM2_EO) 0x0006)
#define TPM2_EO_UNSIGNED_GE     ((TPM2_EO) 0x0007)
#define TPM2_EO_SIGNED_LE       ((TPM2_EO) 0x0008)
#define TPM2_EO_UNSIGNED_LE     ((TPM2_EO) 0x0009)
#define TPM2_EO_BITSET          ((TPM2_EO) 0x000A)
#define TPM2_EO_BITCLEAR        ((TPM2_EO) 0x000B)

/* Table 19 - Definition of (UINT16) TPM2_ST Constants */
typedef UINT16 TPM2_ST;
#define TPM2_ST_RSP_COMMAND             ((TPM2_ST) 0x00C4)
#define TPM2_ST_NULL                    ((TPM2_ST) 0X8000)
#define TPM2_ST_NO_SESSIONS             ((TPM2_ST) 0x8001)
#define TPM2_ST_SESSIONS                ((TPM2_ST) 0x8002)
#define TPM2_ST_ATTEST_NV               ((TPM2_ST) 0x8014)
#define TPM2_ST_ATTEST_COMMAND_AUDIT    ((TPM2_ST) 0x8015)
#define TPM2_ST_ATTEST_SESSION_AUDIT    ((TPM2_ST) 0x8016)
#define TPM2_ST_ATTEST_CERTIFY          ((TPM2_ST) 0x8017)
#define TPM2_ST_ATTEST_QUOTE            ((TPM2_ST) 0x8018)
#define TPM2_ST_ATTEST_TIME             ((TPM2_ST) 0x8019)
```

```
#define TPM2_ST_ATTEST_CREATION         ((TPM2_ST) 0x801A)
#define TPM2_ST_CREATION                ((TPM2_ST) 0x8021)
#define TPM2_ST_VERIFIED                ((TPM2_ST) 0x8022)
#define TPM2_ST_AUTH_SECRET             ((TPM2_ST) 0x8023)
#define TPM2_ST_HASHCHECK               ((TPM2_ST) 0x8024)
#define TPM2_ST_AUTH_SIGNED             ((TPM2_ST) 0x8025)
#define TPM2_ST_FU_MANIFEST             ((TPM2_ST) 0x8029)

/* Table 20 - Definition of (UINT16) TPM2_SU Constants */
typedef UINT16 TPM2_SU;
#define TPM2_SU_CLEAR   ((TPM2_SU) 0x0000)
#define TPM2_SU_STATE   ((TPM2_SU) 0x0001)

/* Table 21 - Definition of (UINT8) TPM2_SE Constants */
typedef UINT8 TPM2_SE;
#define TPM2_SE_HMAC    ((TPM2_SE) 0x00)
#define TPM2_SE_POLICY  ((TPM2_SE) 0x01)
#define TPM2_SE_TRIAL   ((TPM2_SE) 0x03)

/* Table 22 - Definition of (UINT32) TPM2_CAP Constants */
typedef UINT32 TPM2_CAP;
#define TPM2_CAP_FIRST              ((TPM2_CAP) 0x00000000)
#define TPM2_CAP_ALGS               ((TPM2_CAP) 0x00000000)
#define TPM2_CAP_HANDLES            ((TPM2_CAP) 0x00000001)
#define TPM2_CAP_COMMANDS           ((TPM2_CAP) 0x00000002)
#define TPM2_CAP_PP_COMMANDS        ((TPM2_CAP) 0x00000003)
#define TPM2_CAP_AUDIT_COMMANDS     ((TPM2_CAP) 0x00000004)
#define TPM2_CAP_PCRS               ((TPM2_CAP) 0x00000005)
#define TPM2_CAP_TPM_PROPERTIES     ((TPM2_CAP) 0x00000006)
#define TPM2_CAP_PCR_PROPERTIES     ((TPM2_CAP) 0x00000007)
#define TPM2_CAP_ECC_CURVES         ((TPM2_CAP) 0x00000008)
#define TPM2_CAP_AUTH_POLICIES      ((TPM2_CAP) 0x00000009)
#define TPM2_CAP_ACT                ((TPM2_CAP) 0x0000000A)
#define TPM2_CAP_LAST               ((TPM2_CAP) 0x0000000A)
#define TPM2_CAP_VENDOR_PROPERTY    ((TPM2_CAP) 0x00000100)

/* Table 23 - Definition of (UINT32) TPM2_PT Constants */
typedef UINT32 TPM2_PT;
#define TPM2_PT_NONE                    ((TPM2_PT) 0x00000000)
#define TPM2_PT_GROUP                   ((TPM2_PT) 0x00000100)
#define TPM2_PT_FIXED                   ((TPM2_PT) (TPM2_PT_GROUP * 1))
#define TPM2_PT_FAMILY_INDICATOR        ((TPM2_PT) (TPM2_PT_FIXED + 0))
#define TPM2_PT_LEVEL                   ((TPM2_PT) (TPM2_PT_FIXED + 1))
#define TPM2_PT_REVISION                ((TPM2_PT) (TPM2_PT_FIXED + 2))
#define TPM2_PT_DAY_OF_YEAR             ((TPM2_PT) (TPM2_PT_FIXED + 3))
#define TPM2_PT_YEAR                    ((TPM2_PT) (TPM2_PT_FIXED + 4))
#define TPM2_PT_MANUFACTURER            ((TPM2_PT) (TPM2_PT_FIXED + 5))
#define TPM2_PT_VENDOR_STRING_1         ((TPM2_PT) (TPM2_PT_FIXED + 6))
#define TPM2_PT_VENDOR_STRING_2         ((TPM2_PT) (TPM2_PT_FIXED + 7))
#define TPM2_PT_VENDOR_STRING_3         ((TPM2_PT) (TPM2_PT_FIXED + 8))
#define TPM2_PT_VENDOR_STRING_4         ((TPM2_PT) (TPM2_PT_FIXED + 9))
#define TPM2_PT_VENDOR_TPM_TYPE         ((TPM2_PT) (TPM2_PT_FIXED +
10))
#define TPM2_PT_FIRMWARE_VERSION_1      ((TPM2_PT) (TPM2_PT_FIXED +
11))
#define TPM2_PT_FIRMWARE_VERSION_2      ((TPM2_PT) (TPM2_PT_FIXED +
12))
```

```
#define TPM2_PT_INPUT_BUFFER            ((TPM2_PT) (TPM2_PT_FIXED +
13))
#define TPM2_PT_HR_TRANSIENT_MIN        ((TPM2_PT) (TPM2_PT_FIXED +
14))
#define TPM2_PT_HR_PERSISTENT_MIN       ((TPM2_PT) (TPM2_PT_FIXED +
15))
#define TPM2_PT_HR_LOADED_MIN           ((TPM2_PT) (TPM2_PT_FIXED +
16))
#define TPM2_PT_ACTIVE_SESSIONS_MAX     ((TPM2_PT) (TPM2_PT_FIXED +
17))
#define TPM2_PT_PCR_COUNT               ((TPM2_PT) (TPM2_PT_FIXED +
18))
#define TPM2_PT_PCR_SELECT_MIN          ((TPM2_PT) (TPM2_PT_FIXED +
19))
#define TPM2_PT_CONTEXT_GAP_MAX         ((TPM2_PT) (TPM2_PT_FIXED +
20))
#define TPM2_PT_NV_COUNTERS_MAX         ((TPM2_PT) (TPM2_PT_FIXED +
22))
#define TPM2_PT_NV_INDEX_MAX            ((TPM2_PT) (TPM2_PT_FIXED +
23))
#define TPM2_PT_MEMORY                  ((TPM2_PT) (TPM2_PT_FIXED +
24))
#define TPM2_PT_CLOCK_UPDATE            ((TPM2_PT) (TPM2_PT_FIXED +
25))
#define TPM2_PT_CONTEXT_HASH            ((TPM2_PT) (TPM2_PT_FIXED +
26))
#define TPM2_PT_CONTEXT_SYM             ((TPM2_PT) (TPM2_PT_FIXED +
27))
#define TPM2_PT_CONTEXT_SYM_SIZE        ((TPM2_PT) (TPM2_PT_FIXED +
28))
#define TPM2_PT_ORDERLY_COUNT           ((TPM2_PT) (TPM2_PT_FIXED +
29))
#define TPM2_PT_MAX_COMMAND_SIZE        ((TPM2_PT) (TPM2_PT_FIXED +
30))
#define TPM2_PT_MAX_RESPONSE_SIZE       ((TPM2_PT) (TPM2_PT_FIXED +
31))
#define TPM2_PT_MAX_DIGEST              ((TPM2_PT) (TPM2_PT_FIXED +
32))
#define TPM2_PT_MAX_OBJECT_CONTEXT      ((TPM2_PT) (TPM2_PT_FIXED +
33))
#define TPM2_PT_MAX_SESSION_CONTEXT     ((TPM2_PT) (TPM2_PT_FIXED +
34))
#define TPM2_PT_PS_FAMILY_INDICATOR     ((TPM2_PT) (TPM2_PT_FIXED +
35))
#define TPM2_PT_PS_LEVEL                ((TPM2_PT) (TPM2_PT_FIXED +
36))
#define TPM2_PT_PS_REVISION             ((TPM2_PT) (TPM2_PT_FIXED +
37))
#define TPM2_PT_PS_DAY_OF_YEAR          ((TPM2_PT) (TPM2_PT_FIXED +
38))
#define TPM2_PT_PS_YEAR                 ((TPM2_PT) (TPM2_PT_FIXED +
39))
#define TPM2_PT_SPLIT_MAX               ((TPM2_PT) (TPM2_PT_FIXED +
40))
#define TPM2_PT_TOTAL_COMMANDS          ((TPM2_PT) (TPM2_PT_FIXED +
41))
```

```
#define TPM2_PT_LIBRARY_COMMANDS        ((TPM2_PT) (TPM2_PT_FIXED +
42))
#define TPM2_PT_VENDOR_COMMANDS         ((TPM2_PT) (TPM2_PT_FIXED +
43))
#define TPM2_PT_NV_BUFFER_MAX           ((TPM2_PT) (TPM2_PT_FIXED +
44))
#define TPM2_PT_MODES                   ((TPM2_PT) (TPM2_PT_FIXED +
45))
#define TPM2_PT_MAX_CAP_BUFFER          ((TPM2_PT) (TPM2_PT_FIXED +
46))
#define TPM2_PT_VAR                     ((TPM2_PT) (TPM2_PT_GROUP * 2))
#define TPM2_PT_PERMANENT               ((TPM2_PT) (TPM2_PT_VAR + 0))
#define TPM2_PT_STARTUP_CLEAR           ((TPM2_PT) (TPM2_PT_VAR + 1))
#define TPM2_PT_HR_NV_INDEX             ((TPM2_PT) (TPM2_PT_VAR + 2))
#define TPM2_PT_HR_LOADED               ((TPM2_PT) (TPM2_PT_VAR + 3))
#define TPM2_PT_HR_LOADED_AVAIL         ((TPM2_PT) (TPM2_PT_VAR + 4))
#define TPM2_PT_HR_ACTIVE               ((TPM2_PT) (TPM2_PT_VAR + 5))
#define TPM2_PT_HR_ACTIVE_AVAIL         ((TPM2_PT) (TPM2_PT_VAR + 6))
#define TPM2_PT_HR_TRANSIENT_AVAIL      ((TPM2_PT) (TPM2_PT_VAR + 7))
#define TPM2_PT_HR_PERSISTENT           ((TPM2_PT) (TPM2_PT_VAR + 8))
#define TPM2_PT_HR_PERSISTENT_AVAIL     ((TPM2_PT) (TPM2_PT_VAR + 9))
#define TPM2_PT_NV_COUNTERS             ((TPM2_PT) (TPM2_PT_VAR + 10))
#define TPM2_PT_NV_COUNTERS_AVAIL       ((TPM2_PT) (TPM2_PT_VAR + 11))
#define TPM2_PT_ALGORITHM_SET           ((TPM2_PT) (TPM2_PT_VAR + 12))
#define TPM2_PT_LOADED_CURVES           ((TPM2_PT) (TPM2_PT_VAR + 13))
#define TPM2_PT_LOCKOUT_COUNTER         ((TPM2_PT) (TPM2_PT_VAR + 14))
#define TPM2_PT_MAX_AUTH_FAIL           ((TPM2_PT) (TPM2_PT_VAR + 15))
#define TPM2_PT_LOCKOUT_INTERVAL        ((TPM2_PT) (TPM2_PT_VAR + 16))
#define TPM2_PT_LOCKOUT_RECOVERY        ((TPM2_PT) (TPM2_PT_VAR + 17))
#define TPM2_PT_NV_WRITE_RECOVERY       ((TPM2_PT) (TPM2_PT_VAR + 18))
#define TPM2_PT_AUDIT_COUNTER_0         ((TPM2_PT) (TPM2_PT_VAR + 19))
#define TPM2_PT_AUDIT_COUNTER_1         ((TPM2_PT) (TPM2_PT_VAR + 20))

/* Table 24 - Definition of (UINT32) TPM2_PT_PCR Constants */
typedef UINT32 TPM2_PT_PCR;
#define TPM2_PT_PCR_FIRST           ((TPM2_PT_PCR) 0x00000000)
#define TPM2_PT_PCR_SAVE            ((TPM2_PT_PCR) 0x00000000)
#define TPM2_PT_PCR_EXTEND_L0       ((TPM2_PT_PCR) 0x00000001)
#define TPM2_PT_PCR_RESET_L0        ((TPM2_PT_PCR) 0x00000002)
#define TPM2_PT_PCR_EXTEND_L1       ((TPM2_PT_PCR) 0x00000003)
#define TPM2_PT_PCR_RESET_L1        ((TPM2_PT_PCR) 0x00000004)
#define TPM2_PT_PCR_EXTEND_L2       ((TPM2_PT_PCR) 0x00000005)
#define TPM2_PT_PCR_RESET_L2        ((TPM2_PT_PCR) 0x00000006)
#define TPM2_PT_PCR_EXTEND_L3       ((TPM2_PT_PCR) 0x00000007)
#define TPM2_PT_PCR_RESET_L3        ((TPM2_PT_PCR) 0x00000008)
#define TPM2_PT_PCR_EXTEND_L4       ((TPM2_PT_PCR) 0x00000009)
#define TPM2_PT_PCR_RESET_L4        ((TPM2_PT_PCR) 0x0000000A)
#define TPM2_PT_PCR_NO_INCREMENT    ((TPM2_PT_PCR) 0x00000011)
#define TPM2_PT_PCR_DRTM_RESET      ((TPM2_PT_PCR) 0x00000012)
#define TPM2_PT_PCR_POLICY          ((TPM2_PT_PCR) 0x00000013)
#define TPM2_PT_PCR_AUTH            ((TPM2_PT_PCR) 0x00000014)
#define TPM2_PT_PCR_LAST            ((TPM2_PT_PCR) 0x00000014)

/* Table 25 - Definition of (UINT32) TPM2_PS Constants */
typedef UINT32 TPM2_PS;
#define TPM2_PS_MAIN            ((TPM2_PS) 0x00000000)
#define TPM2_PS_PC              ((TPM2_PS) 0x00000001)
```

```
#define TPM2_PS_PDA             ((TPM2_PS) 0x00000002)
#define TPM2_PS_CELL_PHONE      ((TPM2_PS) 0x00000003)
#define TPM2_PS_SERVER          ((TPM2_PS) 0x00000004)
#define TPM2_PS_PERIPHERAL      ((TPM2_PS) 0x00000005)
#define TPM2_PS_TSS             ((TPM2_PS) 0x00000006)
#define TPM2_PS_STORAGE         ((TPM2_PS) 0x00000007)
#define TPM2_PS_AUTHENTICATION  ((TPM2_PS) 0x00000008)
#define TPM2_PS_EMBEDDED        ((TPM2_PS) 0x00000009)
#define TPM2_PS_HARDCOPY        ((TPM2_PS) 0x0000000A)
#define TPM2_PS_INFRASTRUCTURE  ((TPM2_PS) 0x0000000B)
#define TPM2_PS_VIRTUALIZATION  ((TPM2_PS) 0x0000000C)
#define TPM2_PS_TNC             ((TPM2_PS) 0x0000000D)
#define TPM2_PS_MULTI_TENANT    ((TPM2_PS) 0x0000000E)
#define TPM2_PS_TC              ((TPM2_PS) 0x0000000F)

/* Table 26 - Definition of Types for Handles */
typedef UINT32  TPM2_HANDLE;

/* Table 27 - Definition of (UINT8) TPM2_HT Constants */
typedef UINT8 TPM2_HT;
#define TPM2_HT_PCR             ((TPM2_HT) 0x00)
#define TPM2_HT_NV_INDEX        ((TPM2_HT) 0x01)
#define TPM2_HT_HMAC_SESSION    ((TPM2_HT) 0x02)
#define TPM2_HT_LOADED_SESSION  ((TPM2_HT) 0x02)
#define TPM2_HT_POLICY_SESSION  ((TPM2_HT) 0x03)
#define TPM2_HT_SAVED_SESSION   ((TPM2_HT) 0x03)
#define TPM2_HT_PERMANENT       ((TPM2_HT) 0x40)
#define TPM2_HT_TRANSIENT       ((TPM2_HT) 0x80)
#define TPM2_HT_PERSISTENT      ((TPM2_HT) 0x81)

/* Table 28 - Definition of (TPM2_HANDLE) TPM2_RH Constants */
typedef TPM2_HANDLE TPM2_RH;
#define TPM2_RH_FIRST           ((TPM2_RH) 0x40000000)
#define TPM2_RH_SRK             ((TPM2_RH) 0x40000000)
#define TPM2_RH_OWNER           ((TPM2_RH) 0x40000001)
#define TPM2_RH_REVOKE          ((TPM2_RH) 0x40000002)
#define TPM2_RH_TRANSPORT       ((TPM2_RH) 0x40000003)
#define TPM2_RH_OPERATOR        ((TPM2_RH) 0x40000004)
#define TPM2_RH_ADMIN           ((TPM2_RH) 0x40000005)
#define TPM2_RH_EK              ((TPM2_RH) 0x40000006)
#define TPM2_RH_NULL            ((TPM2_RH) 0x40000007)
#define TPM2_RH_UNASSIGNED      ((TPM2_RH) 0x40000008)
#define TPM2_RS_PW              ((TPM2_RH) 0x40000009)
#define TPM2_RH_LOCKOUT         ((TPM2_RH) 0x4000000A)
#define TPM2_RH_ENDORSEMENT     ((TPM2_RH) 0x4000000B)
#define TPM2_RH_PLATFORM        ((TPM2_RH) 0x4000000C)
#define TPM2_RH_PLATFORM_NV     ((TPM2_RH) 0x4000000D)
#define TPM2_RH_AUTH_00         ((TPM2_RH) 0x40000010)
#define TPM2_RH_AUTH_FF         ((TPM2_RH) 0x4000010F)
#define TPM2_RH_ACT_0           ((TPM2_RH) 0x40000110)
#define TPM2_RH_ACT_1           ((TPM2_RH) 0x40000111)
#define TPM2_RH_ACT_2           ((TPM2_RH) 0x40000112)
#define TPM2_RH_ACT_3           ((TPM2_RH) 0x40000113)
#define TPM2_RH_ACT_4           ((TPM2_RH) 0x40000114)
#define TPM2_RH_ACT_5           ((TPM2_RH) 0x40000115)
#define TPM2_RH_ACT_6           ((TPM2_RH) 0x40000116)
#define TPM2_RH_ACT_7           ((TPM2_RH) 0x40000117)
```

```
#define TPM2_RH_ACT_8           ((TPM2_RH) 0x40000118)
#define TPM2_RH_ACT_9           ((TPM2_RH) 0x40000119)
#define TPM2_RH_ACT_A           ((TPM2_RH) 0x4000011A)
#define TPM2_RH_ACT_B           ((TPM2_RH) 0x4000011B)
#define TPM2_RH_ACT_C           ((TPM2_RH) 0x4000011C)
#define TPM2_RH_ACT_D           ((TPM2_RH) 0x4000011D)
#define TPM2_RH_ACT_E           ((TPM2_RH) 0x4000011E)
#define TPM2_RH_ACT_F           ((TPM2_RH) 0x4000011F)
#define TPM2_RH_LAST            ((TPM2_RH) 0x4000011F)
#define TPM2_RH_LAST            ((TPM2_RH) 0x4000011F)

/* Table 29 - Definition of (TPM2_HANDLE) TPM2_HC Constants */
typedef TPM2_HANDLE TPM2_HC;
#define TPM2_HR_HANDLE_MASK         ((TPM2_HC) 0x00FFFFFF)
#define TPM2_HR_RANGE_MASK          ((TPM2_HC) 0xFF000000)
#define TPM2_HR_SHIFT               ((TPM2_HC) 24)
#define TPM2_HR_PCR                 ((TPM2_HC) (TPM2_HT_PCR <<
TPM2_HR_SHIFT))
#define TPM2_HR_HMAC_SESSION        ((TPM2_HC) (TPM2_HT_HMAC_SESSION <<
TPM2_HR_SHIFT))
#define TPM2_HR_POLICY_SESSION      ((TPM2_HC) (TPM2_HT_POLICY_SESSION
<< TPM2_HR_SHIFT))
#define TPM2_HR_TRANSIENT           ((TPM2_HC) (TPM2_HT_TRANSIENT <<
TPM2_HR_SHIFT))
#define TPM2_HR_PERSISTENT          ((TPM2_HC) (TPM2_HT_PERSISTENT <<
TPM2_HR_SHIFT))
#define TPM2_HR_NV_INDEX            ((TPM2_HC) (TPM2_HT_NV_INDEX <<
TPM2_HR_SHIFT))
#define TPM2_HR_PERMANENT           ((TPM2_HC) (TPM2_HT_PERMANENT <<
TPM2_HR_SHIFT))
#define TPM2_PCR_FIRST              ((TPM2_HC) (TPM2_HR_PCR + 0))
#define TPM2_PCR_LAST               ((TPM2_HC) (TPM2_PCR_FIRST +
TPM2_MAX_PCRS-1))
#define TPM2_HMAC_SESSION_FIRST     ((TPM2_HC) (TPM2_HR_HMAC_SESSION+
0))
#define TPM2_HMAC_SESSION_LAST      ((TPM2_HC)
(TPM2_HMAC_SESSION_FIRST+0x00fffffe))
#define TPM2_LOADED_SESSION_FIRST   ((TPM2_HC) TPM2_HMAC_SESSION_FIRST)
#define TPM2_LOADED_SESSION_LAST    ((TPM2_HC) TPM2_HMAC_SESSION_LAST)
#define TPM2_POLICY_SESSION_FIRST   ((TPM2_HC) (TPM2_HR_POLICY_SESSION+
0))
#define TPM2_POLICY_SESSION_LAST    ((TPM2_HC)
(TPM2_POLICY_SESSION_FIRST + 0x00fffffe))
#define TPM2_TRANSIENT_FIRST        ((TPM2_HC) (TPM2_HR_TRANSIENT +0))
#define TPM2_ACTIVE_SESSION_FIRST   ((TPM2_HC)
TPM2_POLICY_SESSION_FIRST)
#define TPM2_ACTIVE_SESSION_LAST    ((TPM2_HC)
TPM2_POLICY_SESSION_LAST)
#define TPM2_TRANSIENT_LAST         ((TPM2_HC)
(TPM2_TRANSIENT_FIRST+0x00ffffe))
#define TPM2_PERSISTENT_FIRST       ((TPM2_HC) (TPM2_HR_PERSISTENT+0))
#define TPM2_PERSISTENT_LAST        ((TPM2_HC)
(TPM2_PERSISTENT_FIRST+0x00FFFFFF))
#define TPM2_PLATFORM_PERSISTENT    ((TPM2_HC) (TPM2_PERSISTENT_FIRST +
0x00800000))
#define TPM2_NV_INDEX_FIRST         ((TPM2_HC) (TPM2_HR_NV_INDEX + 0))
```

```
#define TPM2_NV_INDEX_LAST          ((TPM2_HC) (TPM2_NV_INDEX_FIRST +
0x00FFFFFF))
#define TPM2_PERMANENT_FIRST        ((TPM2_HC) TPM2_RH_FIRST)
#define TPM2_PERMANENT_LAST         ((TPM2_HC) TPM2_RH_LAST)
#define TPM2_HR_NV_AC               ((TPM2_HC) ((TPM2_HT_NV_INDEX <<
HR_SHIFT) + 0xD00000))
#define TPM2_NV_AC_FIRST            ((TPM2_HC) (TPM2_HR_NV_AC + 0))
#define TPM2_NV_AC_LAST             ((TPM2_HC) (TPM2_HR_NV_AC +
0x0000FFFF))
#define TPM2_HR_AC                  ((TPM2_HC) (TPM2_HT_AC <<
HR_SHIFT))
#define TPM2_AC_FIRST               ((TPM2_HC) (TPM2_HR_AC + 0))
#define TPM2_AC_LAST                ((TPM2_HC) (TPM2_HR_AC +
0x0000FFFF))

/* Table 30 - Definition of (UINT32) TPMA_ALGORITHM Bits */
typedef UINT32 TPMA_ALGORITHM;
#define TPMA_ALGORITHM_ASYMMETRIC         ((TPMA_ALGORITHM) 0x00000001)
#define TPMA_ALGORITHM_SYMMETRIC          ((TPMA_ALGORITHM) 0x00000002)
#define TPMA_ALGORITHM_HASH               ((TPMA_ALGORITHM) 0x00000004)
#define TPMA_ALGORITHM_OBJECT             ((TPMA_ALGORITHM) 0x00000008)
#define TPMA_ALGORITHM_RESERVED1_MASK     ((TPMA_ALGORITHM) 0x000000f0)
#define TPMA_ALGORITHM_SIGNING            ((TPMA_ALGORITHM) 0x00000100)
#define TPMA_ALGORITHM_ENCRYPTING         ((TPMA_ALGORITHM) 0x00000200)
#define TPMA_ALGORITHM_METHOD             ((TPMA_ALGORITHM) 0x00000400)
#define TPMA_ALGORITHM_RESERVED2_MASK     ((TPMA_ALGORITHM) 0xfffff800)

/* Table 31 - Definition of (UINT32) TPMA_OBJECT Bits */
typedef UINT32 TPMA_OBJECT;
#define TPMA_OBJECT_RESERVED1_MASK            ((TPMA_OBJECT)
0x00000001)
#define TPMA_OBJECT_FIXEDTPM                  ((TPMA_OBJECT)
0x00000002)
#define TPMA_OBJECT_STCLEAR                   ((TPMA_OBJECT)
0x00000004)
#define TPMA_OBJECT_RESERVED2_MASK            ((TPMA_OBJECT)
0x00000008)
#define TPMA_OBJECT_FIXEDPARENT               ((TPMA_OBJECT)
0x00000010)
#define TPMA_OBJECT_SENSITIVEDATAORIGIN       ((TPMA_OBJECT)
0x00000020)
#define TPMA_OBJECT_USERWITHAUTH              ((TPMA_OBJECT)
0x00000040)
#define TPMA_OBJECT_ADMINWITHPOLICY           ((TPMA_OBJECT)
0x00000080)
#define TPMA_OBJECT_RESERVED3_MASK            ((TPMA_OBJECT)
0x00000300)
#define TPMA_OBJECT_NODA                      ((TPMA_OBJECT)
0x00000400)
#define TPMA_OBJECT_ENCRYPTEDDUPLICATION      ((TPMA_OBJECT)
0x00000800)
#define TPMA_OBJECT_RESERVED4_MASK            ((TPMA_OBJECT)
0x0000f000)
#define TPMA_OBJECT_RESTRICTED                ((TPMA_OBJECT)
0x00010000)
#define TPMA_OBJECT_DECRYPT                   ((TPMA_OBJECT)
0x00020000)
```

```
#define TPMA_OBJECT_SIGN_ENCRYPT              ((TPMA_OBJECT)
0x00040000)
#define TPMA_OBJECT_X509SIGN                  ((TPMA_OBJECT)
0x00080000)
#define TPMA_OBJECT_RESERVED5_MASK            ((TPMA_OBJECT)
0xfff80000)

/* Table 32 - Definition of (UINT8) TPMA_SESSION Bits */
typedef UINT8 TPMA_SESSION;
#define TPMA_SESSION_CONTINUESESSION      ((TPMA_SESSION) 0x01)
#define TPMA_SESSION_AUDITEXCLUSIVE       ((TPMA_SESSION) 0x02)
#define TPMA_SESSION_AUDITRESET           ((TPMA_SESSION) 0x04)
#define TPMA_SESSION_RESERVED1_MASK       ((TPMA_SESSION) 0x18)
#define TPMA_SESSION_DECRYPT              ((TPMA_SESSION) 0x20)
#define TPMA_SESSION_ENCRYPT              ((TPMA_SESSION) 0x40)
#define TPMA_SESSION_AUDIT                ((TPMA_SESSION) 0x80)

/* Table 33 - Definition of (UINT8) TPMA_LOCALITY Bits */
typedef UINT8 TPMA_LOCALITY;
#define TPMA_LOCALITY_TPM2_LOC_ZERO       ((TPMA_LOCALITY) 0x01)
#define TPMA_LOCALITY_TPM2_LOC_ONE        ((TPMA_LOCALITY) 0x02)
#define TPMA_LOCALITY_TPM2_LOC_TWO        ((TPMA_LOCALITY) 0x04)
#define TPMA_LOCALITY_TPM2_LOC_THREE      ((TPMA_LOCALITY) 0x08)
#define TPMA_LOCALITY_TPM2_LOC_FOUR       ((TPMA_LOCALITY) 0x10)
#define TPMA_LOCALITY_EXTENDED_MASK       ((TPMA_LOCALITY) 0xe0)
#define TPMA_LOCALITY_EXTENDED_SHIFT      (5)

/* Table 34 - Definition of (UINT32) TPMA_PERMANENT Bits */
typedef UINT32 TPMA_PERMANENT;
#define TPMA_PERMANENT_OWNERAUTHSET           ((TPMA_PERMANENT)
0x00000001)
#define TPMA_PERMANENT_ENDORSEMENTAUTHSET     ((TPMA_PERMANENT)
0x00000002)
#define TPMA_PERMANENT_LOCKOUTAUTHSET         ((TPMA_PERMANENT)
0x00000004)
#define TPMA_PERMANENT_RESERVED1_MASK         ((TPMA_PERMANENT)
0x000000f8)
#define TPMA_PERMANENT_DISABLECLEAR           ((TPMA_PERMANENT)
0x00000100)
#define TPMA_PERMANENT_INLOCKOUT              ((TPMA_PERMANENT)
0x00000200)
#define TPMA_PERMANENT_TPMGENERATEDEPS        ((TPMA_PERMANENT)
0x00000400)
#define TPMA_PERMANENT_RESERVED2_MASK         ((TPMA_PERMANENT)
0xfffff800)

/* Table 35 - Definition of (UINT32) TPMA_STARTUP_CLEAR Bits */
typedef UINT32 TPMA_STARTUP_CLEAR;
#define TPMA_STARTUP_CLEAR_PHENABLE           ((TPMA_STARTUP_CLEAR)
0x00000001)
#define TPMA_STARTUP_CLEAR_SHENABLE           ((TPMA_STARTUP_CLEAR)
0x00000002)
#define TPMA_STARTUP_CLEAR_EHENABLE           ((TPMA_STARTUP_CLEAR)
0x00000004)
#define TPMA_STARTUP_CLEAR_PHENABLENV         ((TPMA_STARTUP_CLEAR)
0x00000008)
```

```
#define TPMA_STARTUP_CLEAR_RESERVED1_MASK     ((TPMA_STARTUP_CLEAR)
0x7ffffff0)
#define TPMA_STARTUP_CLEAR_ORDERLY            ((TPMA_STARTUP_CLEAR)
0x80000000)

/* Table 36 - Definition of (UINT32) TPMA_MEMORY Bits */
typedef UINT32 TPMA_MEMORY;
#define TPMA_MEMORY_SHAREDRAM              ((TPMA_MEMORY) 0x00000001)
#define TPMA_MEMORY_SHAREDNV               ((TPMA_MEMORY) 0x00000002)
#define TPMA_MEMORY_OBJECTCOPIEDTORAM      ((TPMA_MEMORY) 0x00000004)
#define TPMA_MEMORY_RESERVED1_MASK         ((TPMA_MEMORY) 0xfffffff8)

/* Table 37 - Definition of (TPM2_CC) TPMA_CC Bits */
typedef TPM2_CC TPMA_CC;
#define TPMA_CC_COMMANDINDEX_MASK      ((TPMA_CC) 0x0000ffff)
#define TPMA_CC_COMMANDINDEX_SHIFT     (0)
#define TPMA_CC_RESERVED1_MASK         ((TPMA_CC) 0x003f0000)
#define TPMA_CC_NV                     ((TPMA_CC) 0x00400000)
#define TPMA_CC_EXTENSIVE              ((TPMA_CC) 0x00800000)
#define TPMA_CC_FLUSHED                ((TPMA_CC) 0x01000000)
#define TPMA_CC_CHANDLES_MASK          ((TPMA_CC) 0x0e000000)
#define TPMA_CC_CHANDLES_SHIFT         (25)
#define TPMA_CC_RHANDLE                ((TPMA_CC) 0x10000000)
#define TPMA_CC_V                      ((TPMA_CC) 0x20000000)
#define TPMA_CC_RES_MASK               ((TPMA_CC) 0xc0000000)
#define TPMA_CC_RES_SHIFT              (30)

/* Table 38 - Definition of (UINT32) TPMA_MODES Bits */
typedef UINT32 TPMA_MODES;
#define TPMA_MODES_FIPS_140_2          ((TPMA_MODES) 0x00000001)
#define TPMA_MODES_RESERVED1_MASK      ((TPMA_MODES) 0xfffffffe)

/* Table 39 - Definition of (UINT32) TPMA_X509_KEY_USAGE Bits */
typedef UINT32 TPMA_X509_KEY_USAGE;

#define TPMA_X509_KEY_USAGE_RESERVED_MASK    ((TPMA_X509_KEY_USAGE)
0x007FFFFF)
#define TPMA_X509_KEY_USAGE_DECIPHER_ONLY    ((TPMA_X509_KEY_USAGE)
0x00800000)
#define TPMA_X509_KEY_USAGE_ENCIPHER_ONLY    ((TPMA_X509_KEY_USAGE)
0x01000000)
#define TPMA_X509_KEY_USAGE_CRLSIGN          ((TPMA_X509_KEY_USAGE)
0x02000000)
#define TPMA_X509_KEY_USAGE_KEYCERTSIGN      ((TPMA_X509_KEY_USAGE)
0x04000000)
#define TPMA_X509_KEY_USAGE_KEYAGREEMENT     ((TPMA_X509_KEY_USAGE)
0x08000000)
#define TPMA_X509_KEY_USAGE_DATAENCIPHERMENT ((TPMA_X509_KEY_USAGE)
0x10000000)
#define TPMA_X509_KEY_USAGE_KEYENCIPHERMENT  ((TPMA_X509_KEY_USAGE)
0x20000000)
#define TPMA_X509_KEY_USAGE_NONREPUDIATION   ((TPMA_X509_KEY_USAGE)
0x40000000)
#define TPMA_X509_KEY_USAGE_DIGITALSIGNATURE ((TPMA_X509_KEY_USAGE)
0x80000000)

/* Table 39 - Definition of (UINT32) TPMA_ACT Bits */
```

```
typedef UINT32 TPMA_ACT;

#define TPMA_ACT_SIGNALED          ((TPMA_ACT) 0x00000000)
#define TPMA_ACT_PRESERVESIGNALED ((TPMA_ACT) 0x00000001)
#define TPMA_ACT_RESERVED_MASK     ((TPMA_ACT) 0xFFFFFFFC)

/* Table 40 - Definition of (BYTE) TPMI_YES_NO Type */
typedef BYTE  TPMI_YES_NO;

/* Table 41 - Definition of (TPM2_HANDLE) TPMI_DH_OBJECT Type */
typedef TPM2_HANDLE  TPMI_DH_OBJECT;

/* Table 42 - Definition of (TPM2_HANDLE) TPMI_DH_PARENT Type */
typedef TPM2_HANDLE  TPMI_DH_PARENT;

/* Table 43 - Definition of (TPM2_HANDLE) TPMI_DH_PERSISTENT Type */
typedef TPM2_HANDLE  TPMI_DH_PERSISTENT;

/* Table 44 - Definition of (TPM2_HANDLE) TPMI_DH_ENTITY Type */
typedef TPM2_HANDLE  TPMI_DH_ENTITY;

/* Table 45 - Definition of (TPM2_HANDLE) TPMI_DH_PCR Type */
typedef TPM2_HANDLE  TPMI_DH_PCR;

/* Table 46 - Definition of (TPM2_HANDLE) TPMI_SH_AUTH_SESSION Type */
typedef TPM2_HANDLE  TPMI_SH_AUTH_SESSION;

/* Table 47 - Definition of (TPM2_HANDLE) TPMI_SH_HMAC Type */
typedef TPM2_HANDLE  TPMI_SH_HMAC;

/* Table 48 - Definition of (TPM2_HANDLE) TPMI_SH_POLICY Type */
typedef TPM2_HANDLE  TPMI_SH_POLICY;

/* Table 49 - Definition of (TPM2_HANDLE) TPMI_DH_CONTEXT Type */
typedef TPM2_HANDLE  TPMI_DH_CONTEXT;

/* Table 50 - Definition of TPM2_HANDLE TPMI_DH_SAVED Type */
typedef TPM2_HANDLE TPMI_DH_SAVED;
#define TPMI_DH_SAVED_TRANSIENT       ((TPMI_DH_SAVED) 0x80000000)
#define TPMI_DH_SAVED_SEQUENCE        ((TPMI_DH_SAVED) 0x80000001)
#define TPMI_DH_SAVED_TRANSIENT_CLEAR ((TPMI_DH_SAVED) 0x80000002)

/* Table 51 - Definition of (TPM2_HANDLE) TPMI_RH_HIERARCHY Type */
typedef TPM2_HANDLE  TPMI_RH_HIERARCHY;

/* Table 52 - Definition of (TPM2_HANDLE) TPMI_RH_ENABLES Type */
typedef TPM2_HANDLE  TPMI_RH_ENABLES;

/* Table 53 - Definition of (TPM2_HANDLE) TPMI_RH_HIERARCHY_AUTH Type
*/
typedef TPM2_HANDLE  TPMI_RH_HIERARCHY_AUTH;

/* Table 55 - Definition of (TPM2_HANDLE) TPMI_RH_PLATFORM Type */
typedef TPM2_HANDLE  TPMI_RH_PLATFORM;

/* Table 56 - Definition of (TPM2_HANDLE) TPMI_RH_OWNER Type */
typedef TPM2_HANDLE  TPMI_RH_OWNER;
```

```
/* Table 57 - Definition of (TPM2_HANDLE) TPMI_RH_ENDORSEMENT Type */
typedef TPM2_HANDLE  TPMI_RH_ENDORSEMENT;

/* Table 58 - Definition of (TPM2_HANDLE) TPMI_RH_PROVISION Type */
typedef TPM2_HANDLE  TPMI_RH_PROVISION;

/* Table 59 - Definition of (TPM2_HANDLE) TPMI_RH_CLEAR Type */
typedef TPM2_HANDLE  TPMI_RH_CLEAR;

/* Table 60 - Definition of (TPM2_HANDLE) TPMI_RH_NV_AUTH Type */
typedef TPM2_HANDLE  TPMI_RH_NV_AUTH;

/* Table 61 - Definition of (TPM2_HANDLE) TPMI_RH_LOCKOUT Type */
typedef TPM2_HANDLE  TPMI_RH_LOCKOUT;

/* Table 62 - Definition of (TPM2_HANDLE) TPMI_RH_NV_INDEX Type */
typedef TPM2_HANDLE  TPMI_RH_NV_INDEX;

/* Table 63 - Definition of (TPM_HANDLE) TPMI_RH_AC Type */
typedef TPM2_HANDLE  TPMI_RH_AC;

/* Table 64 - Definition of (TPM_HANDLE) TPMI_RH_ACT Type */
typedef TPM2_HANDLE  TPMI_RH_ACT;

/* Table 65 - Definition of (TPM2_ALG_ID) TPMI_ALG_HASH Type */
typedef TPM2_ALG_ID  TPMI_ALG_HASH;

/* Table 66 - Definition of (TPM2_ALG_ID) TPMI_ALG_ASYM Type */
typedef TPM2_ALG_ID  TPMI_ALG_ASYM;

/* Table 67 - Definition of (TPM2_ALG_ID) TPMI_ALG_SYM Type */
typedef TPM2_ALG_ID  TPMI_ALG_SYM;

/* Table 68 - Definition of (TPM2_ALG_ID) TPMI_ALG_SYM_OBJECT Type */
typedef TPM2_ALG_ID  TPMI_ALG_SYM_OBJECT;

/* Table 69 - Definition of (TPM2_ALG_ID) TPMI_ALG_SYM_MODE Type */
typedef TPM2_ALG_ID  TPMI_ALG_SYM_MODE;

/* Table 70 - Definition of (TPM2_ALG_ID) TPMI_ALG_KDF Type */
typedef TPM2_ALG_ID  TPMI_ALG_KDF;

/* Table 71 - Definition of (TPM2_ALG_ID) TPMI_ALG_SIG_SCHEME Type */
typedef TPM2_ALG_ID  TPMI_ALG_SIG_SCHEME;

/* Table 72 - Definition of (TPM2_ALG_ID){ECC} TPMI_ECC_KEY_EXCHANGE
Type */
typedef TPM2_ALG_ID  TPMI_ECC_KEY_EXCHANGE;

/* Table 73 - Definition of (TPM2_ST) TPMI_ST_COMMAND_TAG Type */
typedef TPM2_ST  TPMI_ST_COMMAND_TAG;

/* Table 74 — Definition of (TPM2_ALG_ID) TPMI_ALG_MAC_SCHEME Type */
typedef TPM2_ALG_ID  TPMI_ALG_MAC_SCHEME;

/* Table 75 — Definition of (TPM2_ALG_ID) TPMI_ALG_CIPHER_MODE Type */
```

```
typedef TPM2_ALG_ID TPMI_ALG_CIPHER_MODE

/* Table 76 - Definition of TPMS_EMPTY Structure */
typedef struct TPMS_EMPTY TPMS_EMPTY;
struct TPMS_EMPTY {
    UINT8   empty[1];
};

/* Table 77 - Definition of TPMS_ALGORITHM_DESCRIPTION Structure */
typedef struct TPMS_ALGORITHM_DESCRIPTION TPMS_ALGORITHM_DESCRIPTION;
struct TPMS_ALGORITHM_DESCRIPTION {
    TPM2_ALG_ID     alg;
    TPMA_ALGORITHM  attributes;
};

/* Table 78 - Definition of TPMU_HA Union */
typedef union TPMU_HA TPMU_HA;
union TPMU_HA {
    BYTE        sha [TPM2_SHA_DIGEST_SIZE];          /* TPM2_ALG_SHA */
    BYTE        sha1 [TPM2_SHA1_DIGEST_SIZE];        /* TPM2_ALG_SHA1
*/
    BYTE        sha256 [TPM2_SHA256_DIGEST_SIZE];    /* TPM2_ALG_SHA256
*/
    BYTE        sha384 [TPM2_SHA384_DIGEST_SIZE];    /* TPM2_ALG_SHA384
*/
    BYTE        sha512 [TPM2_SHA512_DIGEST_SIZE];    /* TPM2_ALG_SHA512
*/
    BYTE        sm3_256 [TPM2_SM3_256_DIGEST_SIZE];  /*
TPM2_ALG_SM3_256 */
    TPMS_EMPTY  null;                                /* TPM2_ALG_NULL
*/
};

/* Table 79 - Definition of TPMT_HA Structure */
typedef struct TPMT_HA  TPMT_HA;
struct TPMT_HA {
    TPMI_ALG_HASH   hashAlg;
    TPMU_HA         digest;
} TPMT_HA;

/* Table 80 - Definition of TPM2B_DIGEST Structure */
typedef struct TPM2B_DIGEST TPM2B_DIGEST;
struct TPM2B_DIGEST {
    UINT16  size;
    BYTE    buffer[sizeof(TPMU_HA)];
};

/* Table 81 - Definition of TPM2B_DATA Structure */
typedef struct TPM2B_DATA TPM2B_DATA;
struct TPM2B_DATA {
    UINT16  size;
    BYTE    buffer[sizeof(TPMT_HA)];
};

/* Table 82 - Definition of Types for TPM2B_NONCE */
typedef TPM2B_DIGEST  TPM2B_NONCE;
```

```
/* Table 83 - Definition of Types for TPM2B_AUTH */
typedef TPM2B_DIGEST  TPM2B_AUTH;

/* Table 84 - Definition of Types for TPM2B_OPERAND */
typedef TPM2B_DIGEST  TPM2B_OPERAND;

/* Table 85 - Definition of TPM2B_EVENT Structure */
typedef struct TPM2B_EVENT TPM2B_EVENT
struct TPM2B_EVENT {
    UINT16  size;
    BYTE    buffer[1024];
};

/* Table 86 - Definition of TPM2B_MAX_BUFFER Structure */
typedef struct TPM2B_MAX_BUFFER TPM2B_MAX_BUFFER;
struct TPM2B_MAX_BUFFER {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_DIGEST_BUFFER];
};

/* Table 87 - Definition of TPM2B_MAX_NV_BUFFER Structure */
typedef struct TPM2B_MAX_NV_BUFFER TPM2B_MAX_NV_BUFFER;
struct TPM2B_MAX_NV_BUFFER {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_NV_BUFFER_SIZE];
};

/* Table 88 - Definition of Types for TPM2B_TIMEOUT */
typedef TPM2B_DIGEST  TPM2B_TIMEOUT;

/* Table 89 - Definition of TPM2B_IV Structure */
typedef struct TPM2B_IV TPM2B_IV;
struct TPM2B_IV {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_SYM_BLOCK_SIZE];
};

/* Table 90 - Definition of TPMU_NAME Union */
typedef union TPMU_NAME TPMU_NAME;
union TPMU_NAME {
    TPMT_HA         digest;
    TPM2_HANDLE     handle;
};

/* Table 91 - Definition of TPM2B_NAME Structure */
typedef struct TPM2B_NAME TPM2B_NAME;
struct TPM2B_NAME {
    UINT16  size;
    BYTE    name[sizeof(TPMU_NAME)];
};

/* Table 92 - Definition of TPMS_PCR_SELECT Structure */
typedef struct TPMS_PCR_SELECT TPMS_PCR_SELECT;
struct TPMS_PCR_SELECT {
    UINT8   sizeofSelect;
    BYTE    pcrSelect[TPM2_PCR_SELECT_MAX];
};
```

```
/* Table 93 - Definition of TPMS_PCR_SELECTION Structure */
typedef struct TPMS_PCR_SELECTION TPMS_PCR_SELECTION;
struct TPMS_PCR_SELECTION {
    TPMI_ALG_HASH   hash;
    UINT8           sizeofSelect;
    BYTE            pcrSelect[TPM2_PCR_SELECT_MAX];
};

/* Table 96 - Definition of TPMT_TK_CREATION Structure */
typedef struct TPMT_TK_CREATION TPMT_TK_CREATION;
struct TPMT_TK_CREATION {
    TPM2_ST             tag;
    TPMI_RH_HIERARCHY   hierarchy;
    TPM2B_DIGEST        digest;
};

/* Table 97 - Definition of TPMT_TK_VERIFIED Structure */
typedef struct TPMT_TK_VERIFIED TPMT_TK_VERIFIED;
struct TPMT_TK_VERIFIED {
    TPM2_ST             tag;
    TPMI_RH_HIERARCHY   hierarchy;
    TPM2B_DIGEST        digest;
};

/* Table 98 - Definition of TPMT_TK_AUTH Structure */
typedef struct TPMT_TK_AUTH TPMT_TK_AUTH;
struct TPMT_TK_AUTH {
    TPM2_ST             tag;
    TPMI_RH_HIERARCHY   hierarchy;
    TPM2B_DIGEST        digest;
};

/* Table 99 - Definition of TPMT_TK_HASHCHECK Structure */
typedef struct TPMT_TK_HASHCHECK TPMT_TK_HASHCHECK;
struct TPMT_TK_HASHCHECK {
    TPM2_ST             tag;
    TPMI_RH_HIERARCHY   hierarchy;
    TPM2B_DIGEST        digest;
} TPMT_TK_HASHCHECK;

/* Table 100 - Definition of TPMS_ALG_PROPERTY Structure */
typedef struct TPMS_ALG_PROPERTY TPMS_ALG_PROPERTY;
struct TPMS_ALG_PROPERTY {
    TPM2_ALG_ID     alg;
    TPMA_ALGORITHM  algProperties;
};

/* Table 101 - Definition of TPMS_TAGGED_PROPERTY Structure */
typedef struct TPMS_TAGGED_PROPERTY TPMS_TAGGED_PROPERTY;
struct TPMS_TAGGED_PROPERTY {
    TPM2_PT     property;
    UINT32      value;
};

/* Table 102 - Definition of TPMS_TAGGED_PCR_SELECT Structure */
typedef struct TPMS_TAGGED_PCR_SELECT TPMS_TAGGED_PCR_SELECT;
```

```
struct TPMS_TAGGED_PCR_SELECT {
    TPM2_PT_PCR     tag;
    UINT8           sizeofSelect;
    BYTE            pcrSelect[TPM2_PCR_SELECT_MAX];
};

/* Table 103 - Definition of TPMS_TAGGED_POLICY Structure */
typedef struct TPMS_TAGGED_POLICY TPMS_TAGGED_POLICY;
struct TPMS_TAGGED_POLICY {
    TPM2_HANDLE     handle;
    TPMT_HA         policyHash;
};

/* Table 104 - Definition of TPMS_ACT_DATA Structure */
typedef struct TPMS_ACT_DATA TPMS_ACT_DATA;
struct TPMS_ACT_DATA {
    TPM2_HANDLE handle;
    UINT32      timeout;
    TPMA_ACT    attributes;
};

/* Table 105 - Definition of TPML_CC Structure */
typedef struct TPML_CC TPML_CC;
struct TPML_CC {
    UINT32      count;
    TPM2_CC     commandCodes[TPM2_MAX_CAP_CC];
};

/* Table 106 - Definition of TPML_CCA Structure */
typedef struct TPML_CCA TPML_CCA;
struct TPML_CCA {
    UINT32      count;
    TPMA_CC     commandAttributes[TPM2_MAX_CAP_CC];
};

/* Table 107 - Definition of TPML_ALG Structure */
typedef struct TPML_ALG TPML_ALG;
struct TPML_ALG {
    UINT32          count;
    TPM2_ALG_ID     algorithms[TPM2_MAX_ALG_LIST_SIZE];
};

/* Table 108 - Definition of TPML_HANDLE Structure */
typedef struct TPML_HANDLE TPML_HANDLE;
struct TPML_HANDLE {
    UINT32          count;
    TPM2_HANDLE     handle[ TPM2_MAX_CAP_HANDLES];
};

/* Table 109 - Definition of TPML_DIGEST Structure */
typedef struct TPML_DIGEST TPML_DIGEST;
struct TPML_DIGEST {
    UINT32          count;
    TPM2B_DIGEST    digests[8];
};

/* Table 110 - Definition of TPML_DIGEST_VALUES Structure */
```

```
typedef struct TPML_DIGEST_VALUES TPML_DIGEST_VALUES;
struct TPML_DIGEST_VALUES {
    UINT32      count;
    TPMT_HA     digests[TPM2_NUM_PCR_BANKS];
};

/* Table 111 - Definition of TPML_PCR_SELECTION Structure */
typedef struct TPML_PCR_SELECTION TPML_PCR_SELECTION;
struct TPML_PCR_SELECTION {
    UINT32              count;
    TPMS_PCR_SELECTION  pcrSelections[TPM2_NUM_PCR_BANKS];
};

/* Table 112 - Definition of TPML_ALG_PROPERTY Structure */
typedef struct TPML_ALG_PROPERTY TPML_ALG_PROPERTY;
struct TPML_ALG_PROPERTY {
    UINT32              count;
    TPMS_ALG_PROPERTY   algProperties[TPM2_MAX_CAP_ALGS];
};

/* Table 113 - Definition of TPML_TAGGED_TPM_PROPERTY Structure */
typedef struct TPML_TAGGED_TPM_PROPERTY TPML_TAGGED_TPM_PROPERTY;
struct TPML_TAGGED_TPM_PROPERTY {
    UINT32                  count;
    TPMS_TAGGED_PROPERTY    tpmProperty[TPM2_MAX_TPM_PROPERTIES];
};

/* Table 114 - Definition of TPML_TAGGED_PCR_PROPERTY Structure */
typedef struct TPML_TAGGED_PCR_PROPERTY TPML_TAGGED_PCR_PROPERTY;
struct TPML_TAGGED_PCR_PROPERTY {
    UINT32                  count;
    TPMS_TAGGED_PCR_SELECT  pcrProperty[TPM2_MAX_PCR_PROPERTIES];
};

/* Table 115 - Definition of {ECC} TPML_ECC_CURVE Structure */
typedef struct TPML_ECC_CURVE TPML_ECC_CURVE;
struct TPML_ECC_CURVE {
    UINT32          count;
    TPM2_ECC_CURVE  eccCurves[TPM2_MAX_ECC_CURVES];
};

/* Table 116 - Definition of TPML_TAGGED_POLICY Structure */
typedef struct TPML_TAGGED_POLICY TPML_TAGGED_POLICY;
struct TPML_TAGGED_POLICY {
    UINT32              count;
    TPMS_TAGGED_POLICY  policies[TPM2_MAX_TAGGED_POLICIES];
};

/* Table 117 - Definition of ECC TPML_ACT_DATA Structure */
typedef struct TPML_ACT_DATA TPML_ACT_DATA;
struct TPML_ACT_DATA {
    UINT32        count;
    TPMS_ACT_DATA actData[TPM2_MAX_ACT_DATA];
};

/* Table 109 - Definition of TPMU_CAPABILITIES Union */
typedef union TPMU_CAPABILITIES TPMU_CAPABILITIES;
```

```
union TPMU_CAPABILITIES {
    TPML_ALG_PROPERTY           algorithms;      /* TPM2_CAP_ALGS */
    TPML_HANDLE                 handles;         /* TPM2_CAP_HANDLES */
    TPML_CCA                    command;         /* TPM2_CAP_COMMANDS
*/
    TPML_CC                     ppCommands;      /*
TPM2_CAP_PP_COMMANDS */
    TPML_CC                     auditCommands;   /*
TPM2_CAP_AUDIT_COMMANDS */
    TPML_PCR_SELECTION          assignedPCR;     /* TPM2_CAP_PCRS */
    TPML_TAGGED_TPM_PROPERTY    tpmProperties;   /*
TPM2_CAP_TPM_PROPERTIES */
    TPML_TAGGED_PCR_PROPERTY    pcrProperties;   /*
TPM2_CAP_PCR_PROPERTIES */
    TPML_ECC_CURVE              eccCurves;       /* TPM2_CAP_ECC_CURVES
*/
    TPML_TAGGED_POLICY          authPolicies;    /*
TPM2_CAP_AUTH_POLICIES */
    TPML_ACT_DATA               actData          /* TPM2_CAP_ACT */
};

/* Table 119 - Definition of TPMS_CAPABILITY_DATA Structure */
typedef struct TPMS_CAPABILITY_DATA TPMS_CAPABILITY_DATA;
struct TPMS_CAPABILITY_DATA {
    TPM2_CAP            capability;
    TPMU_CAPABILITIES   data;
};

/* Table 120 - Definition of TPMS_CLOCK_INFO Structure */
typedef struct TPMS_CLOCK_INFO TPMS_CLOCK_INFO;
struct TPMS_CLOCK_INFO {
    UINT64          clock;
    UINT32          resetCount;
    UINT32          restartCount;
    TPMI_YES_NO     safe;
};

/* Table 121 - Definition of TPMS_TIME_INFO Structure */
typedef struct TPMS_TIME_INFO TPMS_TIME_INFO;
struct TPMS_TIME_INFO {
    UINT64              time;
    TPMS_CLOCK_INFO     clockInfo;
};

/* Table 122 - Definition of TPMS_TIME_ATTEST_INFO Structure */
typedef struct TPMS_TIME_ATTEST_INFO TPMS_TIME_ATTEST_INFO;
struct TPMS_TIME_ATTEST_INFO {
    TPMS_TIME_INFO  time;
    UINT64          firmwareVersion;
};

/* Table 123 - Definition of TPMS_CERTIFY_INFO Structure */
typedef struct TPMS_CERTIFY_INFO TPMS_CERTIFY_INFO;
struct TPMS_CERTIFY_INFO {
    TPM2B_NAME  name;
    TPM2B_NAME  qualifiedName;
};
```

```
/* Table 124 - Definition of TPMS_QUOTE_INFO Structure */
typedef struct TPMS_QUOTE_INFO TPMS_QUOTE_INFO;
struct TPMS_QUOTE_INFO {
    TPML_PCR_SELECTION  pcrSelect;
    TPM2B_DIGEST        pcrDigest;
};

/* Table 125 - Definition of TPMS_COMMAND_AUDIT_INFO Structure */
typedef struct TPMS_COMMAND_AUDIT_INFO TPMS_COMMAND_AUDIT_INFO;
struct TPMS_COMMAND_AUDIT_INFO {
    UINT64          auditCounter;
    TPM2_ALG_ID     digestAlg;
    TPM2B_DIGEST    auditDigest;
    TPM2B_DIGEST    commandDigest;
};

/* Table 126 - Definition of TPMS_SESSION_AUDIT_INFO Structure */
typedef struct TPMS_SESSION_AUDIT_INFO TPMS_SESSION_AUDIT_INFO;
struct TPMS_SESSION_AUDIT_INFO {
    TPMI_YES_NO     exclusiveSession;
    TPM2B_DIGEST    sessionDigest;
};

/* Table 127 - Definition of TPMS_CREATION_INFO Structure */
typedef struct TPMS_CREATION_INFO TPMS_CREATION_INFO;
struct TPMS_CREATION_INFO {
    TPM2B_NAME      objectName;
    TPM2B_DIGEST    creationHash;
};

/* Table 128 - Definition of TPMS_NV_CERTIFY_INFO Structure */
typedef struct TPMS_NV_CERTIFY_INFO TPMS_NV_CERTIFY_INFO;
struct TPMS_NV_CERTIFY_INFO {
    TPM2B_NAME              indexName;
    UINT16                  offset;
    TPM2B_MAX_NV_BUFFER     nvContents;
};

/* Table 129 - Definition of TPMS_NV_DIGEST_CERTIFY_INFO Structure */
typedef struct TPMS_NV_DIGEST_CERTIFY_INFO TPMS_NV_DIGEST_CERTIFY_INFO;
struct TPMS_NV_DIGEST_CERTIFY_INFO {
    TPM2B_NAME              indexName;
    TPM2B_DIGEST            nvDigest;
};

/* Table 130 - Definition of (TPM2_ST) TPMI_ST_ATTEST Type */
typedef TPM2_ST  TPMI_ST_ATTEST;

/* Table 131 - Definition of TPMU_ATTEST Union */
typedef union TPMU_ATTEST TPMU_ATTEST;
union TPMU_ATTEST {
    TPMS_CERTIFY_INFO           certify;         /*
TPM2_ST_ATTEST_CERTIFY */
    TPMS_CREATION_INFO          creation;        /*
TPM2_ST_ATTEST_CREATION */
```

```
    TPMS_QUOTE_INFO             quote;          /*
TPM2_ST_ATTEST_QUOTE */
    TPMS_COMMAND_AUDIT_INFO     commandAudit;   /*
TPM2_ST_ATTEST_COMMAND_AUDIT */
    TPMS_SESSION_AUDIT_INFO     sessionAudit;   /*
TPM2_ST_ATTEST_SESSION_AUDIT */
    TPMS_TIME_ATTEST_INFO       time;           /* TPM2_ST_ATTEST_TIME
*/
    TPMS_NV_CERTIFY_INFO        nv;             /* TPM2_ST_ATTEST_NV
*/
};

/* Table 132 - Definition of TPMS_ATTEST Structure */
typedef struct TPMS_ATTEST TPMS_ATTEST;
struct TPMS_ATTEST {
    TPM2_GENERATED      magic;
    TPMI_ST_ATTEST      type;
    TPM2B_NAME          qualifiedSigner;
    TPM2B_DATA          extraData;
    TPMS_CLOCK_INFO     clockInfo;
    UINT64              firmwareVersion;
    TPMU_ATTEST         attested;
};

/* Table 133 - Definition of TPM2B_ATTEST Structure */
typedef struct TPM2B_ATTEST TPM2B_ATTEST;
struct TPM2B_ATTEST {
    UINT16  size;
    BYTE    attestationData[sizeof(TPMS_ATTEST)];
};

/* Table 134 - Definition of TPMS_AUTH_COMMAND Structure */
typedef struct TPMS_AUTH_COMMAND TPMS_AUTH_COMMAND;
struct TPMS_AUTH_COMMAND {
    TPMI_SH_AUTH_SESSION    sessionHandle;
    TPM2B_NONCE             nonce;
    TPMA_SESSION            sessionAttributes;
    TPM2B_AUTH              hmac;
};

/* Table 135 - Definition of TPMS_AUTH_RESPONSE Structure */
typedef struct TPMS_AUTH_RESPONSE TPMS_AUTH_RESPONSE;
struct TPMS_AUTH_RESPONSE {
    TPM2B_NONCE     nonce;
    TPMA_SESSION    sessionAttributes;
    TPM2B_AUTH      hmac;
};

/* Table 136 - Definition of { !ALG.S} (TPM2_KEY_BITS)
TPMI_!ALG.S_KEY_BITS Type */
typedef TPM2_KEY_BITS  TPMI_AES_KEY_BITS;
typedef TPM2_KEY_BITS  TPMI_SM4_KEY_BITS;
typedef TPM2_KEY_BITS  TPMI_CAMELLIA_KEY_BITS;

/* Table 137 - Definition of TPMU_SYM_KEY_BITS Union */
typedef union TPMU_SYM_KEY_BITS TPMU_SYM_KEY_BITS;
union TPMU_SYM_KEY_BITS {
```

```
    TPMI_AES_KEY_BITS       aes;            /* TPM2_ALG_AES */
    TPMI_SM4_KEY_BITS       sm4;            /* TPM2_ALG_SM4 */
    TPMI_CAMELLIA_KEY_BITS  camellia;       /* TPM2_ALG_CAMELLIA */
    TPM2_KEY_BITS           sym;
    TPMI_ALG_HASH           exclusiveOr;    /* TPM2_ALG_XOR */
    TPMS_EMPTY              null;           /* TPM2_ALG_NULL */
};

/* Table 138 - Definition of TPMU_SYM_MODE Union */
typedef union TPMU_SYM_MODE TPMU_SYM_MODE;
union TPMU_SYM_MODE {
    TPMI_ALG_SYM_MODE   aes;            /* TPM2_ALG_AES */
    TPMI_ALG_SYM_MODE   sm4;            /* TPM2_ALG_SM4 */
    TPMI_ALG_SYM_MODE   camellia;       /* TPM2_ALG_CAMELLIA */
    TPMI_ALG_SYM_MODE   sym;
    TPMS_EMPTY          exclusiveOr;    /* TPM2_ALG_XOR */
    TPMS_EMPTY          null;           /* TPM2_ALG_NULL */
};

/* Table 140 - Definition of TPMT_SYM_DEF Structure */
typedef struct TPMT_SYM_DEF TPMT_SYM_DEF;
struct TPMT_SYM_DEF {
    TPMI_ALG_SYM        algorithm;
    TPMU_SYM_KEY_BITS   keyBits;
    TPMU_SYM_MODE       mode;
};

/* Table 141 - Definition of TPMT_SYM_DEF_OBJECT Structure */
typedef struct TPMT_SYM_DEF_OBJECT TPMT_SYM_DEF_OBJECT;
struct TPMT_SYM_DEF_OBJECT {
    TPMI_ALG_SYM_OBJECT     algorithm;
    TPMU_SYM_KEY_BITS       keyBits;
    TPMU_SYM_MODE           mode;
};

/* Table 142 - Definition of TPM2B_SYM_KEY Structure */
typedef struct TPM2B_SYM_KEY TPM2B_SYM_KEY;
struct TPM2B_SYM_KEY {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_SYM_KEY_BYTES];
};

/* Table 143 - Definition of TPMS_SYMCIPHER_PARMS Structure */
typedef struct TPMS_SYMCIPHER_PARMS TPMS_SYMCIPHER_PARMS;
struct TPMS_SYMCIPHER_PARMS {
    TPMT_SYM_DEF_OBJECT     sym;
};

/* Table 144 - Definition of TPM2B_LABEL Structure */
typedef struct TPM2B_LABEL TPM2B_LABEL;
struct TPM2B_LABEL {
    UINT16  size;
    BYTE    buffer[TPM2_LABEL_MAX_BUFFER];
};

/* Table 145 - Definition of TPMS_DERIVE Structure */
typedef struct TPMS_DERIVE TPMS_DERIVE;
```

```
struct TPMS_DERIVE {
    TPM2B_LABEL     label;
    TPM2B_LABEL     context;
};

/* Table 146 - Definition of TPM2B_DERIVE Structure */
typedef struct TPM2B_DERIVE TPM2B_DERIVE;
struct TPM2B_DERIVE {
    UINT16  size;
    BYTE    buffer[ sizeof(TPMS_DERIVE)];
};

/* Table 147 - Definition of TPMU_SENSITIVE_CREATE Union */
typedef union TPMU_SENSITIVE_CREATE TPMU_SENSITIVE_CREATE;
union TPMU_SENSITIVE_CREATE {
    BYTE            create[TPM2_MAX_SYM_DATA];
    TPMS_DERIVE     derive;
};

/* Table 148 - Definition of TPM2B_SENSITIVE_DATA Structure */
typedef struct TPM2B_SENSITIVE_DATA TPM2B_SENSITIVE_DATA;
struct TPM2B_SENSITIVE_DATA {
    UINT16  size;
    BYTE    buffer[ sizeof(TPMU_SENSITIVE_CREATE)];
};

/* Table 149 - Definition of TPMS_SENSITIVE_CREATE Structure */
typedef struct TPMS_SENSITIVE_CREATE TPMS_SENSITIVE_CREATE;
struct TPMS_SENSITIVE_CREATE {
    TPM2B_AUTH              userAuth;
    TPM2B_SENSITIVE_DATA    data;
};

/* Table 150 - Definition of TPM2B_SENSITIVE_CREATE Structure */
typedef struct TPM2B_SENSITIVE_CREATE TPM2B_SENSITIVE_CREATE;
struct TPM2B_SENSITIVE_CREATE {
    UINT16                  size;
    TPMS_SENSITIVE_CREATE   sensitive;
};

/* Table 151 - Definition of TPMS_SCHEME_HASH Structure */
typedef struct TPMS_SCHEME_HASH TPMS_SCHEME_HASH;
struct TPMS_SCHEME_HASH {
    TPMI_ALG_HASH   hashAlg;
};

/* Table 152 - Definition of { ECC } TPMS_SCHEME_ECDAA Structure */
typedef struct TPMS_SCHEME_ECDAA TPMS_SCHEME_ECDAA;
struct TPMS_SCHEME_ECDAA {
    TPMI_ALG_HASH   hashAlg;
    UINT16          count;
};

/* Table 153 - Definition of (TPM2_ALG_ID) TPMI_ALG_KEYEDHASH_SCHEME
Type */
typedef TPM2_ALG_ID  TPMI_ALG_KEYEDHASH_SCHEME;
```

```
/* Table 154 - Definition of Types for HMAC_SIG_SCHEME */
typedef TPMS_SCHEME_HASH  TPMS_SCHEME_HMAC;

/* Table 155 - Definition of TPMS_SCHEME_XOR Structure */
typedef struct TPMS_SCHEME_XOR TPMS_SCHEME_XOR;
struct TPMS_SCHEME_XOR {
    TPMI_ALG_HASH   hashAlg;
    TPMI_ALG_KDF    kdf;
};

/* Table 156 - Definition of TPMU_SCHEME_KEYEDHASH Union */
typedef union TPMU_SCHEME_KEYEDHASH TPMU_SCHEME_KEYEDHASH;
union TPMU_SCHEME_KEYEDHASH {
    TPMS_SCHEME_HMAC    hmac;           /* TPM2_ALG_HMAC */
    TPMS_SCHEME_XOR     exclusiveOr;    /* TPM2_ALG_XOR */
    TPMS_EMPTY          null;           /* TPM2_ALG_NULL */
};

/* Table 157 - Definition of TPMT_KEYEDHASH_SCHEME Structure */
typedef struct TPMT_KEYEDHASH_SCHEME TPMT_KEYEDHASH_SCHEME;
struct TPMT_KEYEDHASH_SCHEME {
    TPMI_ALG_KEYEDHASH_SCHEME   scheme;
    TPMU_SCHEME_KEYEDHASH       details;
};

/* Table 158 - Definition of {RSA} Types for RSA Signature Schemes */
typedef TPMS_SCHEME_HASH  TPMS_SIG_SCHEME_RSASSA;
typedef TPMS_SCHEME_HASH  TPMS_SIG_SCHEME_RSAPSS;

/* Table 159 - Definition of {ECC} Types for ECC Signature Schemes */
typedef TPMS_SCHEME_HASH  TPMS_SIG_SCHEME_ECDSA;
typedef TPMS_SCHEME_HASH  TPMS_SIG_SCHEME_SM2;
typedef TPMS_SCHEME_HASH  TPMS_SIG_SCHEME_ECSCHNORR;
typedef TPMS_SCHEME_ECDAA  TPMS_SIG_SCHEME_ECDAA;

/* Table 160 - Definition of TPMU_SIG_SCHEME Union */
typedef union TPMU_SIG_SCHEME TPMU_SIG_SCHEME;
union TPMU_SIG_SCHEME {
    TPMS_SIG_SCHEME_RSASSA      rsassa;     /* TPM2_ALG_RSASSA */
    TPMS_SIG_SCHEME_RSAPSS      rsapss;     /* TPM2_ALG_RSAPSS */
    TPMS_SIG_SCHEME_ECDSA       ecdsa;      /* TPM2_ALG_ECDSA */
    TPMS_SIG_SCHEME_ECDAA       ecdaa;      /* TPM2_ALG_ECDAA */
    TPMS_SIG_SCHEME_SM2         sm2;        /* TPM2_ALG_SM2 */
    TPMS_SIG_SCHEME_ECSCHNORR   ecschnorr;  /* TPM2_ALG_ECSCHNORR */
    TPMS_SCHEME_HMAC            hmac;       /* TPM2_ALG_HMAC */
    TPMS_SCHEME_HASH            any;
    TPMS_EMPTY                  null;       /* TPM2_ALG_NULL */
};

/* Table 161 - Definition of TPMT_SIG_SCHEME Structure */
typedef struct TPMT_SIG_SCHEME TPMT_SIG_SCHEME;
struct TPMT_SIG_SCHEME {
    TPMI_ALG_SIG_SCHEME     scheme;
    TPMU_SIG_SCHEME         details;
};

/* Table 162 - Definition of Types for {RSA} Encryption Schemes */
```

```
typedef TPMS_SCHEME_HASH  TPMS_ENC_SCHEME_OAEP;
typedef TPMS_EMPTY  TPMS_ENC_SCHEME_RSAES;

/* Table 163 - Definition of Types for {ECC} ECC Key Exchange */
typedef TPMS_SCHEME_HASH  TPMS_KEY_SCHEME_ECDH;
typedef TPMS_SCHEME_HASH  TPMS_KEY_SCHEME_ECMQV;

/* Table 164 - Definition of Types for KDF Schemes */
typedef TPMS_SCHEME_HASH  TPMS_SCHEME_MGF1;
typedef TPMS_SCHEME_HASH  TPMS_SCHEME_KDF1_SP800_56A;
typedef TPMS_SCHEME_HASH  TPMS_SCHEME_KDF2;
typedef TPMS_SCHEME_HASH  TPMS_SCHEME_KDF1_SP800_108;

/* Table 165 - Definition of TPMU_KDF_SCHEME Union */
typedef union TPMU_KDF_SCHEME TPMU_KDF_SCHEME;
union TPMU_KDF_SCHEME {
    TPMS_SCHEME_MGF1            mgf1;            /* TPM2_ALG_MGF1 */
    TPMS_SCHEME_KDF1_SP800_56A  kdf1_sp800_56a;  /*
TPM2_ALG_KDF1_SP800_56A */
    TPMS_SCHEME_KDF2            kdf2;            /* TPM2_ALG_KDF2 */
    TPMS_SCHEME_KDF1_SP800_108  kdf1_sp800_108;  /*
TPM2_ALG_KDF1_SP800_108 */
    TPMS_EMPTY                  null;            /* TPM2_ALG_NULL */
};

/* Table 166 - Definition of TPMT_KDF_SCHEME Structure */
typedef struct TPMT_KDF_SCHEME TPMT_KDF_SCHEME;
struct TPMT_KDF_SCHEME {
    TPMI_ALG_KDF        scheme;
    TPMU_KDF_SCHEME     details;
};

/* Table 167 - Definition of (TPM2_ALG_ID) TPMI_ALG_ASYM_SCHEME Type */
typedef TPM2_ALG_ID  TPMI_ALG_ASYM_SCHEME;

/* Table 168 - Definition of TPMU_ASYM_SCHEME Union */
typedef union TPMU_ASYM_SCHEME TPMU_ASYM_SCHEME;
union TPMU_ASYM_SCHEME {
    TPMS_KEY_SCHEME_ECDH        ecdh;        /* TPM2_ALG_ECDH */
    TPMS_KEY_SCHEME_ECMQV       ecmqv;       /* TPM2_ALG_ECMQV */
    TPMS_SIG_SCHEME_RSASSA      rsassa;      /* TPM2_ALG_RSASSA */
    TPMS_SIG_SCHEME_RSAPSS      rsapss;      /* TPM2_ALG_RSAPSS */
    TPMS_SIG_SCHEME_ECDSA       ecdsa;       /* TPM2_ALG_ECDSA */
    TPMS_SIG_SCHEME_ECDAA       ecdaa;       /* TPM2_ALG_ECDAA */
    TPMS_SIG_SCHEME_SM2         sm2;         /* TPM2_ALG_SM2 */
    TPMS_SIG_SCHEME_ECSCHNORR   ecschnorr;   /* TPM2_ALG_ECSCHNORR */
    TPMS_ENC_SCHEME_RSAES       rsaes;       /* TPM2_ALG_RSAES */
    TPMS_ENC_SCHEME_OAEP        oaep;        /* TPM2_ALG_OAEP */
    TPMS_SCHEME_HASH            anySig;
    TPMS_EMPTY                  null;        /* TPM2_ALG_NULL */
};

/* Table 169 - Definition of TPMT_ASYM_SCHEME Structure */
typedef struct TPMT_ASYM_SCHEME TPMT_ASYM_SCHEME;
struct TPMT_ASYM_SCHEME {
    TPMI_ALG_ASYM_SCHEME    scheme;
    TPMU_ASYM_SCHEME        details;
```

```
};

/* Table 170 - Definition of (TPM2_ALG_ID) { RSA } TPMI_ALG_RSA_SCHEME
Type */
typedef TPM2_ALG_ID  TPMI_ALG_RSA_SCHEME;

/* Table 171 - Definition of { RSA } TPMT_RSA_SCHEME Structure */
typedef struct TPMT_RSA_SCHEME TPMT_RSA_SCHEME;
struct TPMT_RSA_SCHEME {
    TPMI_ALG_RSA_SCHEME     scheme;
    TPMU_ASYM_SCHEME        details;
};

/* Table 172 - Definition of (TPM2_ALG_ID) { RSA } TPMI_ALG_RSA_DECRYPT
Type */
typedef TPM2_ALG_ID  TPMI_ALG_RSA_DECRYPT;

/* Table 173 - Definition of { RSA } TPMT_RSA_DECRYPT Structure */
typedef struct TPMT_RSA_DECRYPT TPMT_RSA_DECRYPT;
struct TPMT_RSA_DECRYPT {
    TPMI_ALG_RSA_DECRYPT    scheme;
    TPMU_ASYM_SCHEME        details;
};

/* Table 174 - Definition of { RSA } TPM2B_PUBLIC_KEY_RSA Structure */
typedef struct TPM2B_PUBLIC_KEY_RSA TPM2B_PUBLIC_KEY_RSA;
struct TPM2B_PUBLIC_KEY_RSA {
    UINT16  size;
    BYTE    buffer[ TPM2_MAX_RSA_KEY_BYTES];
};

/* Table 175 - Definition of { RSA } (TPM2_KEY_BITS) TPMI_RSA_KEY_BITS
Type */
typedef TPM2_KEY_BITS  TPMI_RSA_KEY_BITS;

/* Table 176 - Definition of { RSA } TPM2B_PRIVATE_KEY_RSA Structure */
typedef struct TPM2B_PRIVATE_KEY_RSA TPM2B_PRIVATE_KEY_RSA;
struct TPM2B_PRIVATE_KEY_RSA {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_RSA_KEY_BYTES/2 * 5];
};

/* Table 177 - Definition of TPM2B_ECC_PARAMETER Structure */
typedef struct TPM2B_ECC_PARAMETER TPM2B_ECC_PARAMETER;
struct TPM2B_ECC_PARAMETER {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_ECC_KEY_BYTES];
};

/* Table 178 - Definition of { ECC } TPMS_ECC_POINT Structure */
typedef struct TPMS_ECC_POINT TPMS_ECC_POINT;
struct TPMS_ECC_POINT {
    TPM2B_ECC_PARAMETER     x;
    TPM2B_ECC_PARAMETER     y;
};

/* Table 179 - Definition of { ECC } TPM2B_ECC_POINT Structure */
```

```
typedef struct TPM2B_ECC_POINT TPM2B_ECC_POINT;
struct TPM2B_ECC_POINT {
    UINT16          size;
    TPMS_ECC_POINT  point;
};

/* Table 180 - Definition of (TPM2_ALG_ID) { ECC } TPMI_ALG_ECC_SCHEME
Type */
typedef TPM2_ALG_ID  TPMI_ALG_ECC_SCHEME;

/* Table 181 - Definition of { ECC } (TPM2_ECC_CURVE) TPMI_ECC_CURVE
Type */
typedef TPM2_ECC_CURVE  TPMI_ECC_CURVE;

/* Table 182 - Definition of (TPMT_SIG_SCHEME) { ECC } TPMT_ECC_SCHEME
Structure */
typedef struct TPMT_ECC_SCHEME TPMT_ECC_SCHEME;
struct TPMT_ECC_SCHEME {
    TPMI_ALG_ECC_SCHEME     scheme;
    TPMU_ASYM_SCHEME        details;
};

/* Table 183 - Definition of { ECC } TPMS_ALGORITHM_DETAIL_ECC
Structure */
typedef struct TPMS_ALGORITHM_DETAIL_ECC TPMS_ALGORITHM_DETAIL_ECC;
struct TPMS_ALGORITHM_DETAIL_ECC {
    TPM2_ECC_CURVE          curveID;
    UINT16                  keySize;
    TPMT_KDF_SCHEME         kdf;
    TPMT_ECC_SCHEME         sign;
    TPM2B_ECC_PARAMETER     p;
    TPM2B_ECC_PARAMETER     a;
    TPM2B_ECC_PARAMETER     b;
    TPM2B_ECC_PARAMETER     gX;
    TPM2B_ECC_PARAMETER     gY;
    TPM2B_ECC_PARAMETER     n;
    TPM2B_ECC_PARAMETER     h;
};

/* Table 184 - Definition of { RSA } TPMS_SIGNATURE_RSA Structure */
typedef struct TPMS_SIGNATURE_RSA TPMS_SIGNATURE_RSA;
struct TPMS_SIGNATURE_RSA {
    TPMI_ALG_HASH           hash;
    TPM2B_PUBLIC_KEY_RSA    sig;
};

/* Table 185 - Definition of Types for {RSA} Signature */
typedef TPMS_SIGNATURE_RSA  TPMS_SIGNATURE_RSASSA;
typedef TPMS_SIGNATURE_RSA  TPMS_SIGNATURE_RSAPSS;

/* Table 186 - Definition of { ECC } TPMS_SIGNATURE_ECC Structure */
typedef struct TPMS_SIGNATURE_ECC TPMS_SIGNATURE_ECC;
struct TPMS_SIGNATURE_ECC {
    TPMI_ALG_HASH           hash;
    TPM2B_ECC_PARAMETER     signatureR;
    TPM2B_ECC_PARAMETER     signatureS;
};
```

```
/* Table 187 - Definition of Types for {ECC} TPMS_SIGNATURE_ECC */
typedef TPMS_SIGNATURE_ECC  TPMS_SIGNATURE_ECDSA;
typedef TPMS_SIGNATURE_ECC  TPMS_SIGNATURE_ECDAA;
typedef TPMS_SIGNATURE_ECC  TPMS_SIGNATURE_SM2;
typedef TPMS_SIGNATURE_ECC  TPMS_SIGNATURE_ECSCHNORR;

/* Table 188 - Definition of TPMU_SIGNATURE Union */
typedef union TPMU_SIGNATURE TPMU_SIGNATURE;
union TPMU_SIGNATURE {
    TPMS_SIGNATURE_RSASSA       rsassa;      /* TPM2_ALG_RSASSA */
    TPMS_SIGNATURE_RSAPSS       rsapss;      /* TPM2_ALG_RSAPSS */
    TPMS_SIGNATURE_ECDSA        ecdsa;       /* TPM2_ALG_ECDSA */
    TPMS_SIGNATURE_ECDAA        ecdaa;       /* TPM2_ALG_ECDAA */
    TPMS_SIGNATURE_SM2          sm2;         /* TPM2_ALG_SM2 */
    TPMS_SIGNATURE_ECSCHNORR    ecschnorr;   /* TPM2_ALG_ECSCHNORR */
    TPMT_HA                     hmac;        /* TPM2_ALG_HMAC */
    TPMS_SCHEME_HASH            any;
    TPMS_EMPTY                  null;        /* TPM2_ALG_NULL */
};

/* Table 189 - Definition of TPMT_SIGNATURE Structure */
typedef struct TPMT_SIGNATURE TPMT_SIGNATURE;
struct TPMT_SIGNATURE {
    TPMI_ALG_SIG_SCHEME     sigAlg;
    TPMU_SIGNATURE          signature;
};

/* Table 190 - Definition of TPMU_ENCRYPTED_SECRET Union */
typedef union TPMU_ENCRYPTED_SECRET TPMU_ENCRYPTED_SECRET;
union TPMU_ENCRYPTED_SECRET {
    BYTE    ecc[sizeof(TPMS_ECC_POINT)];         /* TPM2_ALG_ECC */
    BYTE    rsa[TPM2_MAX_RSA_KEY_BYTES];         /* TPM2_ALG_RSA */
    BYTE    symmetric[sizeof(TPM2B_DIGEST)];     /* TPM2_ALG_SYMCIPHER
*/
    BYTE    keyedHash[sizeof(TPM2B_DIGEST)];     /* TPM2_ALG_KEYEDHASH
*/
};

/* Table 191 - Definition of TPM2B_ENCRYPTED_SECRET Structure */
typedef struct TPM2B_ENCRYPTED_SECRET TPM2B_ENCRYPTED_SECRET;
struct TPM2B_ENCRYPTED_SECRET {
    UINT16  size;
    BYTE    secret[sizeof(TPMU_ENCRYPTED_SECRET)];
};

/* Table 192 - Definition of (TPM2_ALG_ID) TPMI_ALG_PUBLIC Type */
typedef TPM2_ALG_ID  TPMI_ALG_PUBLIC;

/* Table 193 - Definition of TPMU_PUBLIC_ID Union */
typedef union TPMU_PUBLIC_ID TPMU_PUBLIC_ID;
union TPMU_PUBLIC_ID {
    TPM2B_DIGEST            keyedHash;   /* TPM2_ALG_KEYEDHASH */
    TPM2B_DIGEST            sym;         /* TPM2_ALG_SYMCIPHER */
    TPM2B_PUBLIC_KEY_RSA    rsa;         /* TPM2_ALG_RSA */
    TPMS_ECC_POINT          ecc;         /* TPM2_ALG_ECC */
    TPMS_DERIVE             derive;
```

```
};

/* Table 194 - Definition of TPMS_KEYEDHASH_PARMS Structure */
typedef struct TPMS_KEYEDHASH_PARMS TPMS_KEYEDHASH_PARMS;
struct TPMS_KEYEDHASH_PARMS {
    TPMT_KEYEDHASH_SCHEME   scheme;
};

/* Table 195 - Definition of TPMS_ASYM_PARMS Structure */
typedef struct TPMS_ASYM_PARMS TPMS_ASYM_PARMS;
struct TPMS_ASYM_PARMS {
    TPMT_SYM_DEF_OBJECT     symmetric;
    TPMT_ASYM_SCHEME        scheme;
};

/* Table 196 - Definition of { RSA } TPMS_RSA_PARMS Structure */
typedef struct TPMS_RSA_PARMS TPMS_RSA_PARMS;
struct TPMS_RSA_PARMS {
    TPMT_SYM_DEF_OBJECT     symmetric;
    TPMT_RSA_SCHEME         scheme;
    TPMI_RSA_KEY_BITS       keyBits;
    UINT32                  exponent;
};

/* Table 197 - Definition of { ECC } TPMS_ECC_PARMS Structure */
typedef struct TPMS_ECC_PARMS TPMS_ECC_PARMS;
struct TPMS_ECC_PARMS {
    TPMT_SYM_DEF_OBJECT     symmetric;
    TPMT_ECC_SCHEME         scheme;
    TPMI_ECC_CURVE          curveID;
    TPMT_KDF_SCHEME         kdf;
};

/* Table 198 - Definition of TPMU_PUBLIC_PARMS Union */
typedef union TPMU_PUBLIC_PARMS TPMU_PUBLIC_PARMS;
union TPMU_PUBLIC_PARMS {
    TPMS_KEYEDHASH_PARMS    keyedHashDetail;     /* TPM2_ALG_KEYEDHASH
*/
    TPMS_SYMCIPHER_PARMS    symDetail;           /* TPM2_ALG_SYMCIPHER
*/
    TPMS_RSA_PARMS          rsaDetail;           /* TPM2_ALG_RSA */
    TPMS_ECC_PARMS          eccDetail;           /* TPM2_ALG_ECC */
    TPMS_ASYM_PARMS         asymDetail;
};

/* Table 199 - Definition of TPMT_PUBLIC_PARMS Structure */
typedef struct TPMT_PUBLIC_PARMS TPMT_PUBLIC_PARMS;
struct TPMT_PUBLIC_PARMS {
    TPMI_ALG_PUBLIC     type;
    TPMU_PUBLIC_PARMS   parameters;
};

/* Table 200 - Definition of TPMT_PUBLIC Structure */
typedef struct TPMT_PUBLIC TPMT_PUBLIC;
struct TPMT_PUBLIC {
    TPMI_ALG_PUBLIC     type;
    TPMI_ALG_HASH       nameAlg;
```

```
    TPMA_OBJECT         objectAttributes;
    TPM2B_DIGEST        authPolicy;
    TPMU_PUBLIC_PARMS   parameters;
    TPMU_PUBLIC_ID      unique;
};

/* Table 201 - Definition of TPM2B_PUBLIC Structure */
typedef struct TPM2B_PUBLIC TPM2B_PUBLIC;
struct TPM2B_PUBLIC {
    UINT16          size;
    TPMT_PUBLIC     publicArea;
};

/* Table 202 - Definition of TPM2B_TEMPLATE Structure */
typedef struct TPM2B_TEMPLATE TPM2B_TEMPLATE;
struct TPM2B_TEMPLATE {
    UINT16  size;
    BYTE    buffer[sizeof(TPMT_PUBLIC)];
};

/* Table 203 - Definition of TPM2B_PRIVATE_VENDOR_SPECIFIC Structure */
typedef struct TPM2B_PRIVATE_VENDOR_SPECIFIC
TPM2B_PRIVATE_VENDOR_SPECIFIC;
struct TPM2B_PRIVATE_VENDOR_SPECIFIC {
    UINT16  size;
    BYTE    buffer[TPM2_PRIVATE_VENDOR_SPECIFIC_BYTES];
};

/* Table 204 - Definition of TPMU_SENSITIVE_COMPOSITE Union */
typedef union TPMU_SENSITIVE_COMPOSITE TPMU_SENSITIVE_COMPOSITE;
union TPMU_SENSITIVE_COMPOSITE {
    TPM2B_PRIVATE_KEY_RSA           rsa;     /* TPM2_ALG_RSA */
    TPM2B_ECC_PARAMETER             ecc;     /* TPM2_ALG_ECC */
    TPM2B_SENSITIVE_DATA            bits;    /* TPM2_ALG_KEYEDHASH */
    TPM2B_SYM_KEY                   sym;     /* TPM2_ALG_SYMCIPHER */
    TPM2B_PRIVATE_VENDOR_SPECIFIC   any;
};

/* Table 205 - Definition of TPMT_SENSITIVE Structure */
typedef struct TPMT_SENSITIVE TPMT_SENSITIVE;
struct TPMT_SENSITIVE {
    TPMI_ALG_PUBLIC             sensitiveType;
    TPM2B_AUTH                  authValue;
    TPM2B_DIGEST                seedValue;
    TPMU_SENSITIVE_COMPOSITE    sensitive;
};

/* Table 206 - Definition of TPM2B_SENSITIVE Structure */
typedef struct TPM2B_SENSITIVE TPM2B_SENSITIVE;
struct TPM2B_SENSITIVE {
    UINT16          size;
    TPMT_SENSITIVE  sensitiveArea;
};

/* Table 207 - Definition of _PRIVATE Structure */
typedef struct _PRIVATE _PRIVATE;
struct _PRIVATE {
```

```
    TPM2B_DIGEST        integrityOuter;
    TPM2B_DIGEST        integrityInner;
    TPM2B_SENSITIVE     sensitive;
};

/* Table 208 - Definition of TPM2B_PRIVATE Structure */
typedef struct TPM2B_PRIVATE TPM2B_PRIVATE;
struct TPM2B_PRIVATE {
    UINT16  size;
    BYTE    buffer[sizeof(_PRIVATE)];
};

/* Table 209 - Definition of TPMS_ID_OBJECT Structure */
typedef struct TPMS_ID_OBJECT TPMS_ID_OBJECT;
struct TPMS_ID_OBJECT {
    TPM2B_DIGEST    integrityHMAC;
    TPM2B_DIGEST    encIdentity;
};

/* Table 210 - Definition of TPM2B_ID_OBJECT Structure */
typedef struct TPM2B_ID_OBJECT TPM2B_ID_OBJECT;
struct TPM2B_ID_OBJECT {
    UINT16  size;
    BYTE    credential[sizeof(TPMS_ID_OBJECT)];
};

/* Table 211 - Definition of (UINT32) TPM2_NV_INDEX Bits */
typedef UINT32 TPM2_NV_INDEX;
#define TPM2_NV_INDEX_INDEX_MASK      ((TPM2_NV_INDEX) 0x00ffffff)
#define TPM2_NV_INDEX_INDEX_SHIFT     (0)
#define TPM2_NV_INDEX_RH_NV_MASK      ((TPM2_NV_INDEX) 0xff000000)
#define TPM2_NV_INDEX_RH_NV_SHIFT     (24)

/* Table 212 - Definition of TPM2_NT Constants */
typedef UINT8 TPM2_NT;
#define TPM2_NT_ORDINARY    ((TPM2_NT) 0x0)
#define TPM2_NT_COUNTER     ((TPM2_NT) 0x1)
#define TPM2_NT_BITS        ((TPM2_NT) 0x2)
#define TPM2_NT_EXTEND      ((TPM2_NT) 0x4)
#define TPM2_NT_PIN_FAIL    ((TPM2_NT) 0x8)
#define TPM2_NT_PIN_PASS    ((TPM2_NT) 0x9)

/* Table 213 - Definition of TPMS_NV_PIN_COUNTER_PARAMETERS Structure
*/
typedef struct TPMS_NV_PIN_COUNTER_PARAMETERS
TPMS_NV_PIN_COUNTER_PARAMETERS;
struct TPMS_NV_PIN_COUNTER_PARAMETERS {
    UINT32  pinCount;
    UINT32  pinLimit;
};

/* Table 214 - Definition of (UINT32) TPMA_NV Bits */
typedef UINT32 TPMA_NV;
#define TPMA_NV_PPWRITE           ((TPMA_NV) 0x00000001)
#define TPMA_NV_OWNERWRITE        ((TPMA_NV) 0x00000002)
#define TPMA_NV_AUTHWRITE         ((TPMA_NV) 0x00000004)
#define TPMA_NV_POLICYWRITE       ((TPMA_NV) 0x00000008)
```

```
#define TPMA_NV_TPM2_NT_MASK      ((TPMA_NV) 0x000000f0)
#define TPMA_NV_TPM2_NT_SHIFT     (4)
#define TPMA_NV_RESERVED1_MASK    ((TPMA_NV) 0x00000300)
#define TPMA_NV_POLICY_DELETE     ((TPMA_NV) 0x00000400)
#define TPMA_NV_WRITELOCKED       ((TPMA_NV) 0x00000800)
#define TPMA_NV_WRITEALL          ((TPMA_NV) 0x00001000)
#define TPMA_NV_WRITEDEFINE       ((TPMA_NV) 0x00002000)
#define TPMA_NV_WRITE_STCLEAR     ((TPMA_NV) 0x00004000)
#define TPMA_NV_GLOBALLOCK        ((TPMA_NV) 0x00008000)
#define TPMA_NV_PPREAD            ((TPMA_NV) 0x00010000)
#define TPMA_NV_OWNERREAD         ((TPMA_NV) 0x00020000)
#define TPMA_NV_AUTHREAD          ((TPMA_NV) 0x00040000)
#define TPMA_NV_POLICYREAD        ((TPMA_NV) 0x00080000)
#define TPMA_NV_RESERVED2_MASK    ((TPMA_NV) 0x01f00000)
#define TPMA_NV_NO_DA             ((TPMA_NV) 0x02000000)
#define TPMA_NV_ORDERLY           ((TPMA_NV) 0x04000000)
#define TPMA_NV_CLEAR_STCLEAR     ((TPMA_NV) 0x08000000)
#define TPMA_NV_READLOCKED        ((TPMA_NV) 0x10000000)
#define TPMA_NV_WRITTEN           ((TPMA_NV) 0x20000000)
#define TPMA_NV_PLATFORMCREATE    ((TPMA_NV) 0x40000000)
#define TPMA_NV_READ_STCLEAR      ((TPMA_NV) 0x80000000)

/* Table 215 - Definition of TPMS_NV_PUBLIC Structure */
typedef struct TPMS_NV_PUBLIC TPMS_NV_PUBLIC;
struct TPMS_NV_PUBLIC {
    TPMI_RH_NV_INDEX    nvIndex;
    TPMI_ALG_HASH       nameAlg;
    TPMA_NV             attributes;
    TPM2B_DIGEST        authPolicy;
    UINT16              dataSize;
};

/* Table 216 - Definition of TPM2B_NV_PUBLIC Structure */
typedef struct TPM2B_NV_PUBLIC TPM2B_NV_PUBLIC;
struct TPM2B_NV_PUBLIC {
    UINT16          size;
    TPMS_NV_PUBLIC  nvPublic;
};

/* Table 217 - Definition of TPM2B_CONTEXT_SENSITIVE Structure */
typedef struct TPM2B_CONTEXT_SENSITIVE TPM2B_CONTEXT_SENSITIVE;
struct TPM2B_CONTEXT_SENSITIVE {
    UINT16  size;
    BYTE    buffer[TPM2_MAX_CONTEXT_SIZE];
};

/* Table 218 - Definition of TPMS_CONTEXT_DATA Structure */
typedef struct TPMS_CONTEXT_DATA TPMS_CONTEXT_DATA;
struct TPMS_CONTEXT_DATA {
    TPM2B_DIGEST                integrity;
    TPM2B_CONTEXT_SENSITIVE     encrypted;
};

/* Table 219 - Definition of TPM2B_CONTEXT_DATA Structure */
typedef struct TPM2B_CONTEXT_DATA TPM2B_CONTEXT_DATA;
struct TPM2B_CONTEXT_DATA {
    UINT16  size;
```

```
    BYTE    buffer[sizeof(TPMS_CONTEXT_DATA)];
};

/* Table 220 - Definition of TPMS_CONTEXT Structure */
typedef struct TPMS_CONTEXT TPMS_CONTEXT;
struct TPMS_CONTEXT {
    UINT64              sequence;
    TPMI_DH_CONTEXT     savedHandle;
    TPMI_RH_HIERARCHY   hierarchy;
    TPM2B_CONTEXT_DATA  contextBlob;
};

/* Table 222 - Definition of TPMS_CREATION_DATA Structure */
typedef struct TPMS_CREATION_DATA TPMS_CREATION_DATA;
struct TPMS_CREATION_DATA {
    TPML_PCR_SELECTION  pcrSelect;
    TPM2B_DIGEST        pcrDigest;
    TPMA_LOCALITY       locality;
    TPM2_ALG_ID         parentNameAlg;
    TPM2B_NAME          parentName;
    TPM2B_NAME          parentQualifiedName;
    TPM2B_DATA          outsideInfo;
};

/* Table 223 - Definition of TPM2B_CREATION_DATA Structure */
typedef struct TPM2B_CREATION_DATA TPM2B_CREATION_DATA;
struct TPM2B_CREATION_DATA {
    UINT16              size;
    TPMS_CREATION_DATA  creationData;
};

/* Table 224 - Definition of (UINT32) TPM2_AT Constants */
#define TPM2_AT_ANY      ((UINT32) 0x00000000)
#define TPM2_AT_ERROR    ((UINT32) 0x00000001)
#define TPM2_AT_PV1      ((UINT32) 0x00000002)
#define TPM2_AT_PV1      ((UINT32) 0x00000002)
#define TPM2_AT_VEND     ((UINT32) 0x80000000)

/* Table 225 - Definition of (UINT32) TPM_AE Constants */
#define TPM_AE_NONE             ((UINT32) 0x00000000)

/* Table 226 - Definition of TPMS_AC_OUTPUT Structure */
typedef struct TPMS_AC_OUTPUT TPMS_AC_OUTPUT;
struct TPMS_AC_OUTPUT {
    TPM_AT              tag;
    UINT32              data;
};

/* Table 227 - Definition of TPML_AC_CAPABILITIES Structure */
typedef struct TPML_AC_CAPABILITIES TPML_AC_CAPABILITIES;
struct TPML_AC_CAPABILITIES {
    UINT32              count;
    TPMS_AC_OUTPUT      acCapabilities[TPM2_MAX_AC_CAPABILITIES];
};
```

4.4 tss2_tpm2_types.h Postlude

```
#endif /* TSS2_TPM2_TYPES_H */
```