

**E
R
R
A
T
A**

Errata for TCG Trusted Platform Module Library v184

Version 1
August 11, 2026

Contact: admin@trustedcomputinggroup.org

Published

DISCLAIMERS, NOTICES, AND LICENSE TERMS

THIS SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

Without limitation, TCG disclaims 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.

This document is copyrighted by Trusted Computing Group (TCG), and no license, express or implied, is granted herein other than as follows: You may not copy or reproduce the document or distribute it to others without written permission from TCG, except that you may freely do so for the purposes of (a) examining or implementing TCG specifications or (b) developing, testing, or promoting information technology standards and best practices, so long as you distribute the document with these disclaimers, notices, and license terms.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

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

CHANGE HISTORY

Version	Date	Description
1	2026/02/01	Added Clause 2.2 through Clause 2.5

Contents

1 Introduction..... 5

2 Errata 6

2.1 TPM_SPEC Date Constants [specification text, code]..... 6

2.2 Sequence object [code]..... 6

2.3 TPM2_ActivateCredential label 8

2.4 RSA-OAEP decode [code]..... 8

2.5 Object Flush [code]..... 8

1 Introduction

This document describes errata and clarifications for the TCG Trusted Platform Module Library Family Version 185 as published. The information in this document is likely – but not certain – to be incorporated into a future version of the specification. Suggested fixes proposed in this document may be modified before being published in a later TCG Specification. Therefore, the contents of this document are not normative and only become normative when included in an updated version of the published specification. Note that since the errata in this document are non-normative, the patent licensing rights granted by Section 16.4 of the Bylaws do not apply.

The heading of each errata in Clause 2 indicates whether the errata affects the specification text or the reference code implementation. This is indicated by the word “specification text” or “code” in square brackets (“[]”).

2 Errata

2.1 TPM_SPEC Date Constants [specification text, code]

The table in Part 2, 6.1 “TPM_SPEC (Specification Version Values)” should be replaced with

Table 1: Definition of (UINT32) TPM_SPEC Constants

Name	Value	Comments
TPM_SPEC_FAMILY	0x322E3000	ASCII “2.0” with null terminator
TPM_SPEC_LEVEL	00	the level number for the specification
TPM_SPEC_VERSION	184	the version number of the specification
TPM_SPEC_YEAR	2026	the year of the version
TPM_SPEC_DAY_OF_YEAR	32	the day of the year (February 1)

That is, the spec date fields TPM_SPEC_YEAR and TPM_SPEC_DAY_OF_YEAR should be set to the date of the most recent Errata document that revised the behavior of the TPM.

Note:

The date in Table 1 reflects Errata v1.

2.2 Sequence object [code]

The reference code in Part 3 does not check if certain handle variables reference a sequence object. The affected cases are:

- activateHandle in TPM2_ActivateCredential(),
- objectHandle in TPM2_Certify() and TPM2_CertifyCreation(), and
- sendObject in TPM2_AC_Send().

To fix the issue, the commands will check that the object referenced by these handles is not a (hash, MAC, or Event) sequence object or otherwise the TPM will return TPM_RC_TYPE or TPM_RC_SEQUENCE. Minimal patches that add these checks in the reference code are:

TPM2_AC_Send:

Code:

```
--- a/TPMcmd/tpm/src/command/AttachedComponent/AC_Send.c
+++ b/TPMcmd/tpm/src/command/AttachedComponent/AC_Send.c
@@ -52,6 +52,10 @@ TPM2_AC_Send(AC_Send_In* in, // IN: input parameter list
    // permanent handle.
    else if(HandleGetType(in->authHandle) != TPM_HT_PERMANENT)
        return TPM_RCS_HANDLE + RC_AC_Send_authHandle;
+
+    // Cannot send a sequence object to an attached component
+    if(ObjectIsSequence(object))
+        return TPM_RCS_TYPE + RC_AC_Send_sendObject;
    // Make sure that the object to be duplicated has the right attributes
    if(IS_ATTRIBUTE(
        object->publicArea.objectAttributes, TPMA_OBJECT, encryptedDuplication)
```

TPM2_Certify:

Code:

```
--- a/TPMCmd/tpm/src/command/Attestation/Certify.c
+++ b/TPMCmd/tpm/src/command/Attestation/Certify.c
@@ -28,6 +28,10 @@ TPM2_Certify(Certify_In* in, // IN: input parameter list
    if(!CryptSelectSignScheme(signObject, &in->inScheme))
        return TPM_RCS_SCHEME + RC_Certify inScheme;

+    // Cannot certify a sequence object
+    if(ObjectIsSequence(certifiedObject))
+        return TPM_RCS_TYPE + RC_Certify_objectHandle;
+
    // Command Output
    // Filling in attest information
    // Common fields
```

TPM2_CertifyCreation:

Code:

```
--- a/TPMCmd/tpm/src/command/Attestation/CertifyCreation.c
+++ b/TPMCmd/tpm/src/command/Attestation/CertifyCreation.c
@@ -26,6 +26,10 @@ TPM2_CertifyCreation(CertifyCreation_In* in, // IN: input parameter list
    OBJECT* certified = HandleToObject(in->objectHandle);
    OBJECT* signObject = HandleToObject(in->signHandle);
    // Input Validation
+    // Cannot certify creation of a sequence object
+    if(ObjectIsSequence(certified))
+        return TPM_RCS_TYPE + RC_CertifyCreation_objectHandle;
+
    if(!IsSigningObject(signObject))
        return TPM_RCS_KEY + RC_CertifyCreation_signHandle;
    if(!CryptSelectSignScheme(signObject, &in->inScheme))
```

TPM2_ActivateCredential:

Code:

```
--- a/TPMCmd/tpm/src/command/Object/ActivateCredential.c
+++ b/TPMCmd/tpm/src/command/Object/ActivateCredential.c
@@ -43,6 +43,10 @@ TPM2_ActivateCredential(ActivateCredential_In* in, // IN: input parameter
    || !IS_ATTRIBUTE(object->publicArea.objectAttributes, TPMA_OBJECT, restricted))
    return TPM_RCS_TYPE + RC_ActivateCredential_keyHandle;

+    // Cannot activate credential of sequence object
+    if(ObjectIsSequence(activateObject))
+        return TPM_RCS_TYPE + RC_ActivateCredential_activateHandle;
+
    // Command output

    // Decrypt input credential data via asymmetric decryption. A
```

2.3 TPM2_ActivateCredential label

TPM2_ActivateCredential() may support the label “IDENTITY2” in addition to “IDENTITY” defined in Part 1, A.10.4 and B.6.4 to decrypt the credential blob to identify that a TPM firmware version has fixed the errata documented in Clause 2.2. The use of “IDENTITY2” is not intended to be a general change to TPM2_ActivateCredential and is not expected to be added as an alternative in future versions of the TPM Specification.

2.4 RSA-OAEP decode [code]

The reference code uses a timing-dependent implementation for RSA-OAEP decryption. RSA private key operations (including padding) should be provided by an implementation’s crypto library. Cryptographic libraries are expected to take special care to avoid timing side-channels like this. A minimal patch that significantly reduces (but does not totally eliminate) secret-dependent execution time in OAEP decryption in the reference code is:

Code:

```
--- a/TPMcmd/tpm/src/crypt/CryptRsa.c
+++ b/TPMcmd/tpm/src/crypt/CryptRsa.c
@@ -381,8 +381,7 @@ static TPM_RC OaepDecode(
    TPM_RC retVal = TPM_RC_SUCCESS;

    // Strange size (anything smaller can't be an OAEP padded block)
-   // Also check for no leading 0
-   if((padded->size < (unsigned)((2 * hLen) + 2)) || (padded->buffer[0] != 0))
+   if(padded->size < (unsigned)((2 * hLen) + 2))
        ERROR_EXIT(TPM_RC_VALUE);
    // Use the hash size to determine what to put through MGF1 in order
    // to recover the seedMask
@@ -414,7 +413,11 @@ static TPM_RC OaepDecode(
    if((CryptHashBlock(hashAlg, label->size, (BYTE*)label->buffer, hLen, seedMask))
        != hLen)
        FAIL(FATAL_ERROR_INTERNAL);
-   if(memcmp(seedMask, mask, hLen) != 0)
+   if(!MemoryEqual(seedMask, mask, hLen))
+       ERROR_EXIT(TPM_RC_VALUE);
+
+   // Check for the leading 0x00 byte.
+   if(padded->buffer[0] != 0)
        ERROR_EXIT(TPM_RC_VALUE);

    // find the start of the data
```

2.5 Object Flush [code]

The reference code only marks the occupied flag in an OBJECT structure during ObjectFlush. The implementation should entirely clear the structure to ensure no potentially sensitive information remains. A patch for the reference code is below.

Code:

```
void ObjectFlush(OBJECT* object)
{
+   MemorySet(object, 0, sizeof(*object));
    object->attributes.occupied = CLEAR;
}
```