Title: TPM 2.0 Improper Object Slot Reuse

ID: TCGVRT0010

Released: 2026-AUG-11

CVE Identifier: CVE-2026-6726

Weaknesses:

CWE-704: Incorrect Type Conversion or Cast

CWE-226: Sensitive Information in Resource Not Removed Before Reuse

CVSS:

CVSS v3.1 Base Score: 7.9 High
Vector: CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:N

CVSS v4.0 Base Score: 8.5 High
Vector: CVSS:4.0/AV:L/AC:L/AT:P/PR:H/UI:N/VC:H/VI:H/VA:N/SC:H/SI:H/SA:N

Overview:

A vulnerability was discovered in the TCG TPM 2.0 reference code affecting all revisions up to and including v184 (2025). This flaw allows improper reuse of object slots between key/data objects and hash contexts, enabling an attacker to falsify TPM attestations and credentials. The vulnerability is rooted in insufficient separation between key/data object storage and hash context storage, which can be exploited by manipulating object slots and sequence creation.

Description:

The TPM 2.0 reference code does not adequately separate the storage of key/data objects (such as those loaded by TPM2_Load or TPM2_LoadExternal) from hash context objects (such as those created by TPM2_HashSequenceStart). An attacker can exploit this by:

- Loading an external object into the first object slot,
- Flushing the object,
- Creating a hash sequence in the same slot,
- Invoking TPM2_ActivateCredential or TPM2_Certify on the slot.

Because the slot is reused without erasing the previous data, the Name of the external object is improperly used in subsequent operations. This allows an attacker to obtain credentials for falsified TPM

keys (e.g., Attestation Key, DevID Key, TLS authentication key) and to falsify other TPM 2.0 attestations with these keys.

Impact:

An adversary with local, elevated privileges can leverage this vulnerability to obtain credentials from a TPM-aware Certificate Authority (CA) for a falsified TPM key. The attacker can then use these credentials to falsify device attestations, such as PCR quotes for health attestation, undermining the integrity of device identity and attestation processes. This impacts systems relying on TPM-based trust, including device onboarding, secure boot, and remote attestation.

Solution and Protective Measures:

TPM Implementors

Review and implement TCG Publications

1. TPM 2.0 Library Specifications v184 Errata Version 1 or higher.
2. TPM 2.0 Library Specifications v1.83 Errata Version 3 or higher.
3. TPM 2.0 Library Specifications v1.59 Errata Version 1.8 or higher.
4. TPM 2.0 Library Specifications v1.38 Errata Version 1.16 or higher.
5. TPM 2.0 Library Specifications v1.16 Errata Version 1.8 or higher.

Relevant sections: “Sequence object”, “TPM2_ActivateCredential label”, and “Object Flush”.

CA Implementors

Attestation based on imported HMAC key and TPM2_Certify/TPM2_CertifyCreation

(TPM-aware) CA implementors should review and incorporate "EK-Based Key Attestation with TPM Firmware Version" into their attestation protocols, with special care to the following details:

- (CRITICAL) The Verifier should check that the certified TPM key (e.g. AK) is not an external object, by either
    - using TPM2_CertifyCreation with the HMAC key (which does not allow attesting to external objects), or
    - if using TPM2_Certify (and if *signHandle* does not reference an ECDAA key), checking that all of the following conditions on the returned TPMS_CERTIFY_INFO structure are met:

(1) *name.name.digest.digest* is unequal to *qualifiedName.name.digest.digest*

(2) *name.name.digest.hashAlg* is equal to *qualifiedName.name.digest.hashAlg*

(3) *name.size* is equal to *qualifiedName.size*

- *name* and *qualifiedName* will be the same if and only if the object is an external object. Note that a TPM without CVE-2026-6726 will never produce a TPMS_CERTIFY_INFO where *name* = *qualifiedName*.
- Note that *name* and *qualifiedName* must not be an Empty Buffer.
- If *signHandle* in TPM2_Certify references an ECDAA key, *qualifiedName* is always set to an Empty Buffer. Therefore, for an ECDAA key, TPM2_CertifyCreation must be used instead.

- (RECOMMENDED) The Verifier should check that the TPM's current *firmwareVersion* corresponds to a version of the TPM firmware from the TPM vendor that is known to have the fix to CVE-2026-6726. Note that this step will only succeed if the TPM has been fixed. If a Verifier wishes to protect itself from CVE-2026-6726 without asserting that the bug was fixed, they can perform just the (CRITICAL) check above.

Note that section 3 “Protocol” of "EK-Based Key Attestation with TPM Firmware Version" describes the steps to issue a credential for a newly created TPM key. If a CA wants to re-issue a credential (or determine whether to revoke a credential) for an existing TPM key, the TPM2_Create command (in step 4) can be omitted (i.e., the key is only loaded).

The advantage of this mitigation solution is:

- It provides a workaround for CVE-2026-6726, i.e., it can be used by a CA (to issue credentials for TPM keys) even before a TPM has been fixed
- It allows a CA to verify a TPM’s firmware version (before a credential is issued)

Attestation based on TPM2_ActivateCredential with modified Label

This second mitigation solution is provided for CA implementors that are unable to change their existing attestation protocol from using TPM2_ActivateCredential to use the new protocol flow based on TPM2_Import and TPM2_Certify/TPM2_CertifyCreation as described in “EK-Based Key Attestation with TPM Firmware Version". This solution can only be used for issuing credentials after a TPM has been fixed.

The advantage of this solution is that it allows an attestation protocol to continue to use TPM2_ActivateCredential with only a minimal modification and it requires no changes for a host platform.

The TPM Errata versions (listed in section 4.1) allow TPM2_ActivateCredential to support the label "IDENTITY2" in addition to "IDENTITY" (as defined in the TPM 2.0 Library Specification Part 1, clause B.10.4 and C.6.4) to decrypt the credential blob with the purpose to identify that a TPM firmware version has fixed CVE-2026-6726. That means, TPM2_ActivateCredential first tries to decrypt the credential blob using the label “IDENTITY”, if that fails, it tries to decrypt it using the label “IDENTITY2”.

The following describes the high-level attestation steps:

1. A CA determines from the EK certificate whether a TPM firmware version is affected by CVE-2026-6726 (e.g., by inspecting the TPM part number and FW version in the EK certificate)
    a. Note: The exact identification method is TPM vendor-specific
2. The CA creates a credential blob
    a. If the TPM firmware is not affected, it uses the normal “IDENTITY” string
    b. If the TPM firmware is affected, it uses the new “IDENTITY2” string
3. A Host performs TPM2_ActivateCredential as usual
4. The CA has assurance that a TPM firmware has been fixed if the decrypted credential is successfully returned by the Host.

Note that a CA (in step 2b) cannot use TPM2_MakeCredential as convenience function to create the credential blob.

Acknowledgement:

The vulnerabilities were found by Liran Perez and Zecharye Galitzky from Intel.

The TCG VRT thanks the external researcher and the CERT/CC for a coordinated vulnerability disclosure with TPM Vendors and the ecosystem. The CERT/CC Vulnerability Note can be found at: https://kb.cert.org/vuls/id/431093

References:

VRT0010/VRT0011 Extended Guidance Document

https://kb.cert.org/vuls/id/431093

https://www.cve.org/CVERecord?id=CVE-2026-6726