

# FDO Voucher Signature Verification and Domain Separation AAD Across Protocol Versions



October 01, 2026

## This version:

<https://fidoalliance.org/specs/fidoiot/fdo-appnote-8-voucher-aad-versioning-v1.0-20261001.html>

## Issue Tracking:

[GitHub](#)

## Editor:

[Brad Goodman](#) (Dell Technologies)

## Version:

1.0

Copyright © 2026 [FIDO Alliance](#). All Rights Reserved.

---

## Abstract

FDO 2.0 introduces domain separation via COSE external Additional Authenticated Data (AAD) for all signing operations, including Ownership Voucher entry signatures and to1d signatures. FDO 1.x does not use AAD in any signing operations. When FDO 2.0 implementations encounter Ownership Vouchers that were created or extended by FDO 1.x systems, signature verification will fail if the verifier unconditionally applies FDO 2.0 AAD fields. This note explains the issue, its root cause, and the correct approach: inspecting the voucher's OVProtVer field to determine whether AAD should be included during verification.

## Table of Contents

|     |                                           |
|-----|-------------------------------------------|
| 1   | <b>Introduction</b>                       |
| 2   | <b>The Problem</b>                        |
| 3   | <b>Background: COSE Domain Separation</b> |
| 4   | <b>The Correct Approach</b>               |
| 4.1 | Rule for Verification                     |
| 4.2 | Rule for Signing                          |
| 5   | <b>Why This is Non-Obvious</b>            |
| 6   | <b>Example Scenario</b>                   |
| 7   | <b>Recommendations</b>                    |
| 8   | <b>Conclusion</b>                         |

## 1. Introduction§

FDO 2.0 introduces a significant security enhancement: domain separation for COSE\_Sign1 signatures using

the `external_aad` field (see RFC 9052 Section 4.3, "Externally Supplied Data"). Domain separation ensures that a signature produced for one protocol operation (e.g., `TO0.OwnerSign`) cannot be replayed as a valid signature for a different operation (e.g., `TO2.ProveOVHdr`), even if the same key is used for both.

This enhancement applies to all FDO 2.0 signing operations, including:

- **Ownership Voucher entry signatures** (`OVEEntry`)
- **TO0.OwnerSign** (to1d signatures)
- **TO1.ProveToRV**
- **TO2.ProveOVHdr, TO2.ProveDevice, TO2.SetupDevice**

FDO 1.x (versions 1.0 and 1.1) does **not** use `external_aad` in any signing operation. All `COSE_Sign1` signatures in FDO 1.x are produced and verified with an empty (nil) AAD value.

This difference creates an interoperability concern when FDO 2.0 implementations process Ownership Vouchers that were originally created or extended by FDO 1.x systems.

## 2. The Problem§

In real-world deployments, it is common for Ownership Vouchers to be created and extended through multiple supply-chain entities over a period of time. Not all entities in the supply chain will upgrade to FDO 2.0 simultaneously. The following scenarios can arise:

- **Vouchers created by 1.x manufacturers, onboarded by 2.0 owners.** A device manufacturer running FDO 1.x performs Device Initialization (DI) and creates an Ownership Voucher. The voucher passes through the supply chain, possibly being extended by other 1.x entities. Eventually, the voucher arrives at an Owner running FDO 2.0.
- **Vouchers extended by a mix of 1.x and 2.0 entities.** Some supply-chain entities may have upgraded to 2.0 while others remain on 1.x. Each extension adds a signed entry to the voucher chain, and the AAD used for each entry's signature depends on which version of the protocol the extending entity was running.
- **All systems running 2.0, but vouchers pre-date the upgrade.** Even in an environment where every endpoint has been upgraded to FDO 2.0, existing Ownership Vouchers may have been created and extended entirely under FDO 1.x. These vouchers remain valid and must continue to be verifiable.

In all of these cases, an FDO 2.0 verifier that unconditionally includes domain-specific AAD (e.g., the CBOR-encoded tag `"FD0-0VEEntry-v1"` for `OVEEntry` signatures, or `"FD0-TO0-OwnerSign-v1"` for to1d signatures) will fail to verify signatures that were originally produced with empty AAD under FDO 1.x.

The symptom is straightforward: `COSE_Sign1` signature verification fails, even though the signatures are valid and the keys are correct. The root cause is a mismatch in the `external_aad` value used during signing (empty, under 1.x) versus verification (domain-tagged, under 2.0).

## 3. Background: COSE Domain Separation§

`COSE_Sign1` signatures (RFC 9052) incorporate an `external_aad` field into the `Sig` structure that is hashed and signed. This field is **not** transmitted in the `COSE_Sign1` output; both the signer and verifier must independently construct the same value based on the protocol context.

FDO 2.0 defines domain separation tags as CBOR-encoded arrays containing a domain string. For example:

```
FD0ExternalAAD = [FD0DomainTag]
FD0DomainTag = tstr
```

; Example values:

```
AADOwnerSign    = FD0ExternalAAD with tag "FD0-T00-OwnerSign-v1"
AAD0VEntry      = FD0ExternalAAD with tag "FD0-0VEntry-v1"
AADProveToRV    = FD0ExternalAAD with tag "FD0-T01-ProveToRV-v1"
AADProveOVHdr   = FD0ExternalAAD with tag "FD0-T02-ProveOVHdr-v1"
AADProveDevice  = FD0ExternalAAD with tag "FD0-T02-ProveDevice-v1"
AADSetupDevice  = FD0ExternalAAD with tag "FD0-T02-SetupDevice-v1"
```

When `external_aad` is included, it becomes part of the data that is signed. If the verifier uses a different `external_aad` value than the signer used, the signature will not verify, even if the key and payload are identical.

## 4. The Correct Approach§

The correct approach is to use the **Ownership Voucher's version field** (`OVProtVer`) to determine whether domain-specific AAD should be applied during signature verification.

The Ownership Voucher structure includes a protocol version at the top level:

```
OwnershipVoucher = [
  OVProtVer:      protver,          ;; protocol version
  OVHeaderTag:    bstr .cbor OVHeader,
  OVHeaderHMac:   HMac,
  OVDevCertChain: OVDevCertChainOrNull,
  OVEntryArray:   OVEntries
]
```

The `OVProtVer` field records the protocol version under which the voucher was created. This field is set at Device Initialization (DI) and does not change as the voucher is extended through the supply chain.

### 4.1. Rule for Verification§

When verifying Ownership Voucher entry signatures or to1d signatures:

- If `OVProtVer` indicates FDO 2.0 (version 200) or higher, use the appropriate domain-specific AAD value (e.g., `AAD0VEntry` for voucher entries, `AAD0wnerSign` for to1d).
- If `OVProtVer` indicates FDO 1.x (version 101 or lower), use empty (nil) AAD.

In pseudocode:

```
if voucher.OVProtVer >= 200:
  aad = domain_specific_aad    // e.g., AAD0VEntry or AAD0wnerSign
else:
  aad = nil                    // empty AAD, as used by FDO 1.x
```

```
verify_cose_sign1(signature, key, aad)
```

This rule applies to:

- **OVEntry verification:** Each entry in the voucher's `OVEntryArray` is a `COSE_Sign1` signature. The voucher's `OVProtVer` determines whether AAD is included in verification.
- **to1d verification:** The to1d blob (signed by the Owner during `TO0.OwnerSign`) is verified during `TO2`. Here, the protocol version from the current session context (negotiated during the `TO2` handshake) determines whether AAD is used, since to1d is produced at runtime rather than at manufacturing time.

## 4.2. Rule for Signing§

When **producing** new signatures:

- Implementations running FDO 2.0 SHOULD always sign with domain-specific AAD.
- When extending an existing FDO 1.x voucher (one whose `OVProtVer` is 101), the extending entity signs the new entry using the AAD policy matching the voucher's `OVProtVer`. Since the voucher's version is 1.x, empty AAD is used for the new entry signature as well. This maintains consistency: all entries in the voucher chain use the same AAD policy.

## 5. Why This is Non-Obvious§

This issue is easy to overlook for several reasons:

1. **All endpoints may be running 2.0.** An operator who has upgraded all systems to FDO 2.0 may reasonably expect everything to "just work." However, if existing Ownership Vouchers were created under 1.x, their embedded signatures use empty AAD. The protocol version running on the endpoints is irrelevant; what matters is the version recorded in the voucher.
2. **The failure mode is cryptographic.** When AAD is mismatched, the error manifests as a signature verification failure. There is no clear indication that the root cause is an AAD mismatch rather than a corrupted voucher or wrong key.
3. **Domain separation is an "invisible" field.** The `external_aad` value is not transmitted in the `COSE_Sign1` structure. It is not visible in the voucher data itself. Both signer and verifier must independently agree on its value. When they disagree, there is no obvious diagnostic clue in the on-wire data.
4. **Mixed supply chains are common.** In practice, Ownership Vouchers pass through many entities, and not all will upgrade on the same schedule. A voucher may have been created years before the Owner endpoint is upgraded.

## 6. Example Scenario§

Consider the following timeline:

1. In 2024, a device manufacturer running FDO 1.1 creates a batch of 1000 devices. Each device receives an Ownership Voucher with `OVProtVer` = 101. All voucher entry signatures are produced with empty AAD.
2. The vouchers are extended through the supply chain by distributors also running FDO 1.1. Additional entries are signed with empty AAD.
3. In 2025, the end-customer (Owner) upgrades their onboarding infrastructure to FDO 2.0.
4. Devices begin arriving and attempting onboarding. During TO2, the Owner's FDO 2.0 implementation verifies the Ownership Voucher.
5. If the implementation naively applies `AAD0VEntry` to all voucher entry verifications (because it is running FDO 2.0), every signature check will fail. The devices cannot onboard.
6. The correct behavior is to inspect `OVProtVer` in each voucher. Since these vouchers have `OVProtVer` = 101, verification proceeds with empty AAD, and the signatures verify successfully.

## 7. Recommendations§

- **Implementors** should always check the voucher's `OVProtVer` field before performing signature verification on voucher entries or to1d blobs. Never assume that because the local implementation is FDO 2.0, all vouchers will have been created under 2.0.
- **Operators** should be aware that upgrading all endpoints to FDO 2.0 does not retroactively change existing

Ownership Vouchers. Vouchers created under 1.x will retain `0VProtVer = 101` and their signatures will remain valid only when verified with empty AAD.

- **Diagnostics:** If signature verification fails on vouchers that are known to be valid, check the `0VProtVer` field. A mismatch between the voucher version and the AAD policy applied during verification is a likely root cause.
- **Testing:** Implementations should include test cases that verify correct handling of FDO 1.x vouchers by FDO 2.0 code paths, and vice versa. This includes:
  - Verifying 1.x vouchers with a 2.0 verifier (must use empty AAD)
  - Verifying 2.0 vouchers with a 2.0 verifier (must use domain AAD)
  - Extending 1.x vouchers from a 2.0 system (must sign with empty AAD)

## 8. Conclusion§

Domain separation via `COSE_external_aad` is a valuable security improvement in FDO 2.0. However, its introduction creates a backward compatibility concern for Ownership Vouchers produced under FDO 1.x. The key insight is that the **voucher's own version field** (`0VProtVer`) -- not the running protocol version of the endpoints -- determines whether AAD should be applied during voucher signature operations. Implementations must inspect this field and apply AAD accordingly to ensure seamless interoperability across protocol versions.

