0-DAY NO CVE HIGH PUBLIC

Bitwarden, EF cipher create skips collection ACL

Bitwarden Server 2026.9.2 lite on MariaDB/Postgres/SQLite nulls the caller UserId before attaching a new org cipher to collections. A confirmed member can plant a decryptable vault item into a collection they cannot write. Members sync it as a shared login. MSSQL/Dapper keeps the caller id. No CVE yet.

NAME	TYPE
Bitwarden, EF cipher create skips collection ACL	0-day analysis
CVE	**CVE RISK**
n/a	high
DISCLOSURE STATUS	**VENDOR**
public	Bitwarden
AFFECTED	**PUBLISHED**
Bitwarden Server v2026.9.2 (9ee4e0eb) self-host lite EF (MariaDB / Postgres / SQLite). Confirmed org member. Lab image ghcr.io/bitwarden/lite:2026.9.2.	02 Oct 2026
UPDATED	**TAGS**
02 Oct 2026	0-day, bitwarden, idor, collection-acl, cwe-863, cwe-639, authenticated

TL;DR

Lab files and proof-of-concept here.

01 the collection said read-only. the row said otherwise.

The proof of concept is on GitHub: **abraxas/bitwarden-cipher-create-acl** (loopback client; @abraxas_null). The lab stack is `lab/`: docker-compose.yml, run.sh, poc.py. Authorized lab only. It talks to loopback.

This is **Bitwarden Server v2026.9.2** (`9ee4e0eb`), Bitwarden. No CVE yet. CWE-863 / CWE-639. 7.1 High (`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:L`). Confirmed org member. Not unauthenticated. Not vault-key theft.

Lite on MariaDB, Postgres, or SQLite uses EF. MSSQL uses Dapper and keeps the caller id.

02 What an attacker can do

The attacker **adds** vault items to a shared collection. They do not read or replace the secrets already in it.

A confirmed org member already has the org key, so the new item decrypts for everyone who can see that collection. Bitwarden shows it as a normal shared login, note, card, or SSH key from inside the company vault.

- **Phish from the vault.** Plant a login named like a real shared item (`Okta`, `GitHub`, `VPN`, `Payroll`) whose URI points at a host they control. Collection members open it from Bitwarden and land on the attacker page, or auto-fill against a lookalike.
- **Poison ops.** Plant a secure note or login that looks like a rotation (`new AWS key`, `bastion SSH key`, `DB password after change`). Someone copies it into production or a terminal.

- **Stay after they are kicked.** The create is a new cipher in the collection. It remains for everyone with access after the attacker membership is removed.
- **Bypass read-only.** A contractor who may only read `Production` can still drop a new item into it on lite EF.

If they have no access to the target collection, `POST /api/ciphers/create` can return 404 after the write is already committed. Victims still get the item on sync.

03 How I found it

I pinned **v2026.9.2** after the 2026 wave. Admin-request TDE (CVE-2026-60104), SSO ExternalId truncation (CVE-2026-101878), org import empty collections (CVE-2026-43638) are closed on this tree. Collection ACL on **create** was not.

`CiphersController.PostCreate` only checks org membership, then passes `skipPermissionCheck: true` when `OrganizationId` is set. `SaveDetailsAsync` then treats "this collection exists in the org" as the only collection check.

`CreateAsyncReturnCipher` sets `cipher.UserId = null` because an org id is present, then hands that null into `UpdateCollectionsAsync`. Null user id means every collection in the org. Dapper `CipherDetails_CreateWithCollections` still passes `@UserId` into `Cipher_UpdateCollections` after it nulls the stored column. That is the leftover.

I stood up stock `ghcr.io/bitwarden/lite:2026.9.2` with MariaDB on loopback. Two users, one org, collection A for the attacker, collection B for the victim. Attacker `POST /api/ciphers/create` with B's id. Create returned 404. `CollectionCipher` already had the row. Victim `GET /api/ciphers` and `/api/sync` both showed the new id and `BITWARDEN-CIPHER-CREATE-ACL-WITNESS`. Read-only on B returned 200 and planted a second item. Foreign-org collection stayed 404.

PLAIN TEXT Copy

```
SUCCESS BITWARDEN-CIPHER-CREATE-ACL create-http=404 collectioncipher=1 victim-syr
```

Wrong turns already recorded: treating cloud `POST /organizations` as the self-host create (it is `[NotSelfHostedOnly]`; lite wants a license file, so the lab seeds org rows after register); reading HTTP 404 on a no-access plant as a miss (the `CollectionCipher` row is already committed); pointing lite at MSSQL and wondering why attach failed (Dapper keeps the caller id); leaving `BW_ENABLE_ADMIN` off so lite never migrates `User` / `OrganizationDomain` (admin is the EF migrator on this image); omitting `Bitwarden-Client-Version` on `/identity/connect/token` and watching password grant 400. A reverse shell. Theatre. The oracle is the victim sync plus the join row.

04 null means everyone

PLAIN TEXT Copy

```
cipher.UserId = cipher.OrganizationId.HasValue ? null : cipher.UserId;
// ...
await UpdateCollectionsAsync(dbContext, cipher.Id,
    cipher.UserId, cipher.OrganizationId, collectionIds);
```

PLAIN TEXT Copy

```
if (!userId.HasValue)
{
    availableCollectionsQuery = context.Collections
        .Where(c ⇒ c.OrganizationId == organizationId.Value)
        .Select(c ⇒ c.Id);
}
```

Pass the saving user id into EF `UpdateCollectionsAsync` the same way the stored procedure still does. Existence in the org is not write access.

I am @abraxas_null. Site abraxaslabs.tech. GitHub abraxas. Mail abraxas.null@proton.me.

© 2026 Abraxas Labs