0-DAY NO CVE HIGH PUBLIC

Forgejo, GET /user/keys/{id} IDOR

Forgejo 16.0.5 GET /api/v1/user/keys/{id} loads public_key by numeric ID with no owner check. A read:user token inventories other people's SSH and deploy keys. GPG on the same pin 404s a foreign id. No CVE yet.

NAME	TYPE
Forgejo, GET /user/keys/{id} IDOR	0-day analysis
CVE	**CVE RISK**
n/a	high
DISCLOSURE STATUS	**VENDOR**
public	Forgejo
AFFECTED	**PUBLISHED**
Forgejo through v16.0.5 (8d24b7d). Authenticated current-user API. Web installer now prefers DISABLE_REGISTRATION=true; any existing account is enough.	12 Oct 2026
UPDATED	**TAGS**
12 Oct 2026	0-day, forgejo, idor, ssh, cwe-639, authenticated

TL;DR

Lab files and proof-of-concept **here**.

01 GPG 404s. SSH does not.

I am @abraxas_null. The proof of concept is on GitHub: **abraxas/forgejo-keys-idor** (loopback client). The lab stack is `lab/`: docker-compose.yml, run.sh, poc.py. Authorized lab only. It talks to loopback.

This is **Forgejo v16.0.5** (`8d24b7d52df327118cd776cf461d34e52e0ec1ce`). No CVE yet. CWE-639. 7.5 High (`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N`). Remote IDOR. Authenticated.

`GET /api/v1/user/keys/{id}` sits in the current-user group. The handler loads the row by numeric ID. There is no owner check. GPG on the same pin uses `GetGPGKeyForUserByID(ctx, ctx.Doer().ID, id)` and 404s a foreign id. SSH does not. That is the leftover.

02 Why Forgejo

I had just finished the Gitea 1.27.3 wave. Forgejo is the fork people actually self-host when they want the forge without the Gitea org. It has diverged. Copying a Gitea pack onto a Forgejo tag is how you invent a finding that is already closed, or miss one that Gitea never had. I re-checked leftover *classes* on this pin instead.

I only wanted Critical or High. Local is in-scope. Medium stays on the floor unless it chains.

03 Pinning v16.0.5

Latest GA on the day I cloned was **v16.0.5**, commit `8d24b7d`, "2026-09-17 security patches (#14370)". Image `codeberg.org/forgejo/forgejo:16.0.5`. That is the pin.

04 The skip list is half the hunt

Before you look for leftovers, you write down what is already public so you do not "discover" a CVE with a press release. Forgejo's list lives at codeberg.org/forgejo/security-announcements. On 16.0.5 the interesting ones were already closed:

- CVE-2026-89094 template `.git` recreate. `root.RemoveAll(".git")` before `git init`.
- ApplyDiffPatch `git apply --index` RCE leftover of PR 13705. Unrelated temp cwd.
- Draft-release attachment leak (Gitea CVE-2026-27660 class). `checkReleaseAccess` is in.
- CVE-2026-89151 restricted tokens plus allow-maintainer-edit. Reducer on.
- OpenID association CSRF. Session `openIDState` nonce.
- CVE-2026-82556 migrate `LookupIP` SSRF. DialContext on HTTP migrate/LFS.
- v16.0.2 CGNAT hostmatcher, org-mode `ReadFile` stub, Actions XSS.
- v16.0.3 reusable-workflow / collab-owner.
- Git clone/push/migrate `http.followRedirects=false`. The Gitea git-redirect class is closed here.

CVE-2026-90679 ActivityPub actor/signature bind is still in tree. Published Medium. Identity spoof only. I left it.

05 Twenty surfaces, one IDOR on keys

I split the tree. Templates, git-apply, Actions, packages, migrate SSRF, hooks, API tokens, federation, OpenID, attachments, restricted users, hostmatcher, LFS, keys IDOR, restore/dump, OAuth/install, markup, internal API, secrets/wiki, visibility.

Most of that is Forgejo doing what Forgejo said it would do after the 2026-09 wave.

Git hooks are `DISABLE_GIT_HOOKS` default true plus a centralized `core.hooksPath`. Writers cannot plant hooks. Internal `/api/internal` is on the public mux and 403s without `INTERNAL_TOKEN`. Artifact v4 `ReadAll` is unbounded and Medium DoS. Stargazers / Follow 204 vs 404 is Medium enum. Public-only PAT still skips `checkTokenPublicOnly` on private orgs and stays below High. Reverse-proxy `X-WEBAUTH-USER` is off unless reverse-proxy auth is enabled.

The current-user key handler is where the temperature changed.

06 The lookup that forgot the owner

`/user/keys/{id}` is in the current-user group (`/api/v1/user`, `reqToken`, user scope). The handler loads the shared `public_key` row by numeric ID only.

GO Copy

```go
key, err := asymkey_model.GetPublicKeyByID(ctx, ctx.ParamsInt64(":id"))
// ...
apiKey := convert.ToPublicKey(apiLink, key)
if ctx.IsUserSiteAdmin() || ctx.Doer().ID == key.OwnerID {
    apiKey, _ = appendPrivateInformation(ctx, apiKey, key, ctx.Doer())
}
ctx.JSON(http.StatusOK, apiKey)
```

`GetPublicKeyByID` is `ID(keyID).Get(key)` with no owner or key-type filter. `ToPublicKey` always emits `Key` (blob), `Title`, `Fingerprint`. The owner test only

gates extra JSON (`user` , `key_type` , `read_only`).

The same table holds user keys, deploy keys (`OwnerID` 0, `KeyTypeDeploy`), and SSH certificate principals (`KeyTypePrincipal`). `ListPublicKeys` excludes principals. This GET does not.

GPG's sibling:

```go
GetGPGKeyForUserByID(ctx, ctx.Doer().ID, id)
```

That is the contrast. Same current-user group. One 404s a foreign id. One does not.

IDs are sequential. Walk `1` , `2` , `3` . A `read:user` token is enough. The web installer now prefers `DISABLE_REGISTRATION=true` . Any existing account, or an instance that still allows signup, is in scope.

The v16.0.5 key-adjacent work was git-apply RCE and OpenID CSRF. It was not this lookup. Same class as gitea-user-keys-idor, independently still open here.

07 Wrong turns, in the order I made them

Copying the Gitea pack. Forgejo has diverged. I re-read the handler.

Treating the attacker's own list as the leak. `GET /api/v1/user/keys` is scoped. GET-by-id is not. The oracle is a foreign id that is absent from the attacker's list.

Treating `DISABLE_REGISTRATION=true` as a close. The installer prefers it. Any existing account is enough. Open registration is a bonus, not a requirement.

Username-scoped `GET /users/{username}/keys` . That list is public-by-username. This leftover is the current-user numeric lookup, including deploy keys that never appear on the username list.

A reverse shell. Theatre. The witness is a key title.

08 What an attacker can do

Register or use any account, mint a `read:user` token (or use a session cookie), walk `/api/v1/user/keys/{id}`. They get titles, fingerprints, and public key material for keys they do not own. They do not get the private key. They do not get a shell. They inventory who has access to what.

PRIVILEGE	METHOD / SINK
Read another user's SSH public key title, fingerprint, and key material	`GET /api/v1/user/keys/{` a foreign owner
Inventory deploy keys the same way	Same handler, `KeyTypeDeploy` row `public_key` (`Owner`

PRIVILEGE	METHOD / SINK
Read SSH certificate principals	Same handler, `KeyTypePrincipal` r `ListPublicKeys` exc this type
See keys that belong to limited or private users	No visibility check or lookup
Extra JSON (`user` , `key_type` , `read_only`)	`appendPrivateInfor` only if admin or owne

Closed on this route: private key material, a shell, `GET /users/{username}/keys` (username-scoped public list), GPG `GET /api/v1/user/gpg_keys/{id}` (404 on a foreign id), `DeletePublicKey` (owner/admin 403). Cross-repo `GET /repos/{owner}/{repo}/keys/{id}` 404s when `RepoID` does not match.

09 The lab

Source of truth is `lab/` . Image `codeberg.org/forgejo/forgejo:16.0.5` . Host bind **127.0.0.1:19310**. Two accounts. Victim key title `FORGEJO-KEYS-IDOR-WITNESS` . Attacker list does not contain that id. `GET /user/keys/1` as the attacker does. GPG with a foreign id 404s.

PLAIN TEXT Copy

```
git clone https://github.com/abraxas/forgejo-keys-idor
cd forgejo-keys-idor
python3 forgejo-keys-idor-Abraxas-Labs.py
```

Last lab run, trimmed:

PLAIN TEXT Copy

```
IOC planted victim-id=1 attacker-id=2
IOC attacker-list status=200 ids=[2] (no witness)
IOC get id=1 status=200 snippet contains FORGEJO-KEYS-IDOR-WITNESS
IOC gpg-neg id=1 status=404
SUCCESS forgejo-keys-idor victim-id=1 get-http=200 list-has-id=no gpg-http=404 FC
```

I am not reprinting the key.

10 The fix

404 unless `key.OwnerID == ctx.Doer().ID` (or site admin). Reject `KeyTypeDeploy` / `KeyTypePrincipal` on this current-user route. Match `GetGPGKeyForUserByID`.

Same tag leftovers: webhook `0.0.0.1` SSRF, OpenID discovery SSRF, restore-repo file open.

11 References

- Proof of concept: abraxas/forgejo-keys-idor · forgejo-keys-idor-Abraxas-Labs.py
- Lab: `lab/` · docker-compose.yml · run.sh · poc.py

- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-639
- Tree: forgejo/forgejo v16.0.5 (`8d24b7d52df327118cd776cf461d34e52e0ec1ce`) · `key.go`
- Same class on Gitea: gitea-user-keys-idor

Abraxas Labs

Independent security research by abraxas.

Analysis of 0-days, n-days, malware, and the leftovers that still matter.

GitHub **@abraxas_null** **abraxas.null@proton.me** **abraxaslabs.tech**

Independent research. Private analysis by arrangement. No ads. abraxas.null@proton.me · For defensive awareness and authorized testing only.

© 2026 Abraxas Labs