New issue

Jump to bottom

# vulnerability that editor user can change admin user's password #327

Closed ufo009e opened this issue on Jan 12, 2020 · 7 comments

Assignees

Labels bug high

Projects Development

**ufo009e** commented on Jan 12, 2020

editor role user can change admin user's properties including password hash, salt and token.

let's say you have 2 users in db
id name isadmin hash
1111 admin true aaaaaa
2222 editor false bbbbbb

editor user can use the postuser function to change his password. Attacker can use this function to change the id and other parameters. If editor changed the id to 1111 (post body)which belong to admin, then he can send another postuser request (set to 1111 in both post body and url) to overwrite admin's properties (since findone by id 1111 matches first row now) including hash, token and salt, also changed isadmin to false,

```
static postUser(req, res) {
    let id = req.params.id;
    let properties = req.body;

    UserController.authenticate(req.cookies.token, req.params.project)
    .then((user) => {
        let hasScope = user.hasScope(req.params.project, 'users');

        if(user.id == id || hasScope) {
            // If the current user does not have the "users" scope, revert any sensitive properties
            if(!hasScope) {
                properties.scopes = user.scopes;
                properties.isAdmin = false;
            }

            return Promise.resolve();
        }

        return Promise.reject(new Error('User "' + user.name + '" does not have scope "user"'));
    })
    .then(() => {
        HashBrown.Service.UserService.updateUserById(id, properties);  <<<<<< accept id, and password value
```

**mrzapp** commented on Jan 12, 2020 • edited Member

I was really hoping for someone like you to show up. Thank you so much for your inspection of the codebase.

I will prevent any changes to the user id. Anything else you think would be appropriate to do here while I'm at it?

**mrzapp** self-assigned this on Jan 12, 2020

**mrzapp** added bug high labels on Jan 12, 2020

**mrzapp** added this to **Backlog** in **Development** via automation on Jan 12, 2020

**mrzapp** moved this from **Backlog** to **To do** in **Development** on Jan 12, 2020

**ufo009e** commented on Jan 12, 2020 Author

the username is dangerous as well. If the editor change the name to admin, the logoff. The logoff update by username function will copy editor user setting to overwirite admin. So there are 2 duplicate lines of editor in db now. I have concern about password part as well, but I can't find a way to use it to exploit without changing username and id.

**mrzapp** commented on Jan 12, 2020 • edited Member

Then maybe let's work backwards from the ideal solution instead of patching an insecure paradigm.

What would be your ideal approach to modifying user data that would prevent these exploits? These are the requirements:

Any user should be able to change their own username, full name, email and password

Admins should be able to change any field of any user

No one should be able to change the id of any user

If we just imagine one API endpoint handling all user data changes, what would that look like to you? Just pseudo code is fine

**ufo009e** commented on Jan 12, 2020 Author

no one can change id. before write new name to db make a check the new name is unique. You have this check when create new users but you did not use it in postuser.

**mrzapp** pushed a commit that referenced this issue on Jan 13, 2020

`Addresses #327 and various related issues` `34daf33`

**mrzapp** pushed a commit that referenced this issue on Jan 13, 2020

`Addresses #327 and various related issues` `01a08c9`

**mrzapp** commented on Jan 13, 2020 Member

**@ufo009e** the latest change implements both a duplicate username check and pruning of potentially disruptive field changes as discussed above.

Are you still able to perform the exploits you found with the latest code in `develop` ?

**ufo009e** commented on Jan 13, 2020 Author

looks like fixed this.

**mrzapp** commented on Jan 13, 2020 Member

Great, thanks!

**mrzapp** closed this as completed on Jan 13, 2020

**Development** automation moved this from **To do** to **Awaiting release** on Jan 13, 2020

Assignees

**mrzapp**

Labels

bug **high**

Projects

**Development**
Awaiting release

Milestone

No milestone

Development

No branches or pull requests

2 participants