

Improper Authentication in snipe/snipe-it

1



Valid

Reported on Aug 28th 2022

Description

There are two permissions not working correctly: The `Licenses -> View and Modify License Files` & the `Self -> Create API Keys` permission.

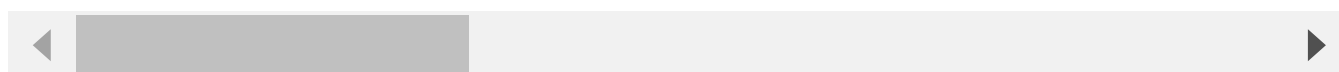
License Files

Files can be uploaded to licenses. There is a permission for users called `View and Modify License Files`. However, this permission is ineffective. A user without this permission is still able to access files uploaded to licenses as long as they have the `View` permission for licenses. Additionally, they can enumerate all uploaded files by simply incrementing the identifier for the file in the URL, since it is a simple counter.

Proof of Concept

Steps to reproduce:

1. Login as admin
2. Go to Licenses and `create` a new license `with` arbitrary `values`
3. Click `on` the License, go `to` File Uploads `and` Upload `any` 2-3 PoC files
4. Observe that the download links `for` the uploaded files are of the form ```
5. Go `to` People `and` `create` a new `user`. Make sure `to` deny `all` permissions `ex`
6. Login `as` the newly created `user`, click `on` Licenses `and` click `on` the lice
7. The URL should now be ``/licenses/<license_id>``
8. Append ``/showfile/3`` `to` the URL `and` observe that the `first` uploaded file
9. `All` files can be enumerated `by` incrementing the ``file_id``



PoC Request from User with only `Licenses -> View` Permission:

```
GET /licenses/1/showfile/3 HTTP/1.1
```

Chat with us

```
Host: 127.0.0.1:8000
Connection: close
```

```
Cookie: snipeit_session=bFxcGzG8fZAfZvPFivACCT7XN9GXdYRBhrvLgZuh
Content-Length: 2
```

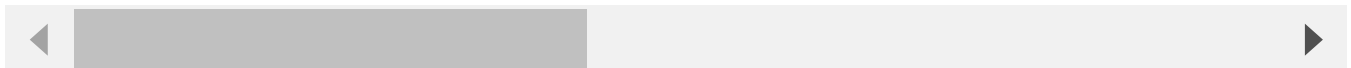
API Keys

A user can create API keys to authenticate to API endpoints. There is a permission called `Create API keys`, however, even users without this permission are able to create API keys, thus rendering the permission ineffective.

Proof of Concept

Steps to reproduce:

1. Login as admin
2. Go to People and `Create` a new `User`. Make Sure to deny all permissions.]
3. Login as the newly created user
4. Take note of the cookie ``snipeit_session`` and the ``csrf-token`` in the H1
5. Make the request to create an API key and observe that an API key is ret



API Creation Request (replace the CSRF Token and session cookie accordingly):

```
POST /oauth/personal-access-tokens HTTP/1.1
Host: 127.0.0.1:8000
X-CSRF-TOKEN: wqLZMfHIXhA8WdsJLmghGHird8AWlDYb8SeEIAIp
Content-Type: application/json; charset=utf-8
Connection: close
Cookie: snipeit_session=CAwxBX0UfnzSh4GD2mIASL0Fp2eoUegXBx0WRN3d
Content-Length: 38
```

```
{"name": "asd", "scopes": [], "errors": []}
```

The created API key is valid although the user does not have the permission

[Chat with us](#)

Impact

Users can access license files & create API keys without the corresponding permissions.

Occurrences

 ProfileController.php L82-L101

The permission check seems to be ineffective.

References

- [CWE-287 - Improper Authentication](#)

CVE

CVE-2022-3173

(Published)

Vulnerability Type

CWE-287: Improper Authentication

Severity

Medium (4.3)

Registry

Other

Affected Version

6.0.10

Visibility

Public

Status

Fixed

Found by



vautia

@vautia

master ▼

Fixed by



snipe

Chat with us



@snipe

maintainer

This report was seen 826 times.

We are processing your report and will contact the **snipe/snipe-it** team within 24 hours.

3 months ago

We have contacted a member of the **snipe/snipe-it** team and are waiting to hear back

3 months ago

We have sent a follow up to the **snipe/snipe-it** team. We will try again in 7 days. 3 months ago

A **snipe/snipe-it** maintainer has acknowledged this report 3 months ago

snipe validated this vulnerability 2 months ago

Hi there - thanks for this :)

Okay, there are two (kind of unrelated) issues here. The API auth stuff *kind of* doesn't matter TBH, since the API checks the individual gates on what a user can do via the API. If they have no permissions, they can't actually do anything with the API anyway, so while this is technically incorrect, the risk here is pretty small. Unfortunately, that `/oauth/personal-access-token` is built into Laravel itself which is why it exists outside of our normal gating system.

I think in order to truly remediate this, we'd need to override the built-in Laravel magic here.

For the file access stuff, you're 100% right - I've got a fix on deck that I'm testing now. (I think these probably should have been two different Huntr issues, since one is effectively exploiting a session replay, versus a missing gate. The enumeration part kinda doesn't matter, since we don't restrict access on a file by file basis, just by the license itself.

You did actually miss one thing tho, which I discovered as I was testing 🤖 🤖 We're not actually checking against full company support - which means:

IF you have full company support enabled AND

IF you have a (non-superadmin) user that can view license files

We are NOT actually checking that the license file belongs to a license that belongs to a company the user is allowed to see. We're skipping the "companyable" step entirely.

So that's fun.

Chat with us

Anyway, we're still working on this. I should have something for you this week.

Best regards,
Snipe

vautia has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

We have sent a fix follow up to the **snipe/snipe-it** team. We will try again in 7 days. 2 months ago

snipe marked this as fixed in **6.0.10** with commit **dcab13** 2 months ago

snipe has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

ProfileController.php#L82-L101 has been validated ✓

Sign in to join this conversation

2022 © 418sec

huntr

home

hacktivity

leaderboard

part of 418sec

company

about

team

Chat with us

[FAQ](#)

[contact us](#)

[terms](#)

[privacy policy](#)

[Chat with us](#)