New issue

Jump to bottom

fix: use `context` property for template variables #163

Closed UziTech wants to merge 3 commits into `express-handlebars:master` from `UziTech:options-data`

Conversation 7 Commits 3 Checks 21 Files changed 4

UziTech commented on Apr 29, 2021 • edited — Member

BREAKING CHANGE:

An object passed to template data will need to be passed as an object in the `context` property.
This prevents mixing untrusted data with express-handlebars options.
For more information see https://blog.shoebpatel.com/2021/01/23/The-Secret-Parameter-LFR-and-Potential-RCE-in-NodeJS-Apps/

Thanks **@agustingianni** for bringing this to my attention.

Example:

```
<h1>Hi, {{name}}</h1>
```

<= v5

```
res.render('hi', {name: "Tony", layout: false})
```

v6

```
res.render('hi', {context: {name: "Tony"}, layout: false})
```

`fix: use context property for template variables ...` ✓ 187194b

UziTech force-pushed the `options-data` branch from **db292ce** to **187194b** last year — Compare

UziTech added 2 commits last year

`chore(readme): update readme to reflect context property` ✓ 4f0689b

`chore(examples): update advanced example with context property` ✓ 35b1f78

agustingianni commented on May 4, 2021

Hello, thanks for tagging me and sorry for the slight delay, I've been swamped with work. I will give it a look now.

agustingianni commented on May 4, 2021

So with this change if you force users to pass the template arguments via `{ context: { ... } }` that should stop the attack, but I think this is not whats happening from my understanding of the code. If the user supplies an object that does not have the `context` handlebars just creates a default one and proceeds with the rendering:

```
        async renderView (viewPath, options = {}, callback = null) {
                const context = options.context || {};
    }
```

My concern would be that users that call res.render("name", taintedObject) are still vulnerable because `taintedObject` can be made in a way that it has the `context` key recently introduced **and** they still can override the same set of properties that allow for the bug to happen.

The problem with trying to fix this is that the problem is at three different levels, first the user uses the `express` render method in a way that is dangerous, second express merges the object passed by the user with data that is intended to be used by the underlying template engine, and third the renderer assumes that the configuration data comming from express is trustworthy.

In my opinion the most balanced way to fix this is to warn clients of handlebars to never pass objects whose keys are attacker controlled to the `render` method. This also has the benefit of not breaking the API for your users.

This is what `eta` did to address the same issue here, https://eta.js.org/docs/examples/express while they wait for `express` to implement a better API for the next big release.

Until then, I can't imagine a proper solution to the issue.

Let me know what you think!

UziTech commented on May 4, 2021 — Member Author

Thanks! I added a note to the readme about this.

UziTech commented on May 4, 2021 — Member Author

I'm going to hold off on this type of change until express fixes their API. Express v5 has been in alpha for almost 7 years 🤞 they will release it one of these days 🤣.

UziTech closed this on May 4, 2021

UziTech reopened this on May 4, 2021

UziTech closed this on May 4, 2021

agustingianni commented on May 10, 2021

Fantastic **@UziTech** thank you for your time addressing this issue!

UziTech mentioned this pull request on Jun 7, 2021

Security Vulnerability Issue [CVE-2021-32820] #184

Closed

bavarianbytes commented on Mar 22

Is it still neccessary to put template variables in a `context` object when passing to the `render` method? Can't find anything about that in the current documentation.

If yes, is it right that i need to access these variables in the handlebar template by calling `{{context.myvariable}}` ?

UziTech commented on Mar 22 Member Author

@bavarianbytes no this change was not merged as it didn't fix the underlying vulnerability. The documentation should be correct.

1

Reviewers

No reviews

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

3 participants