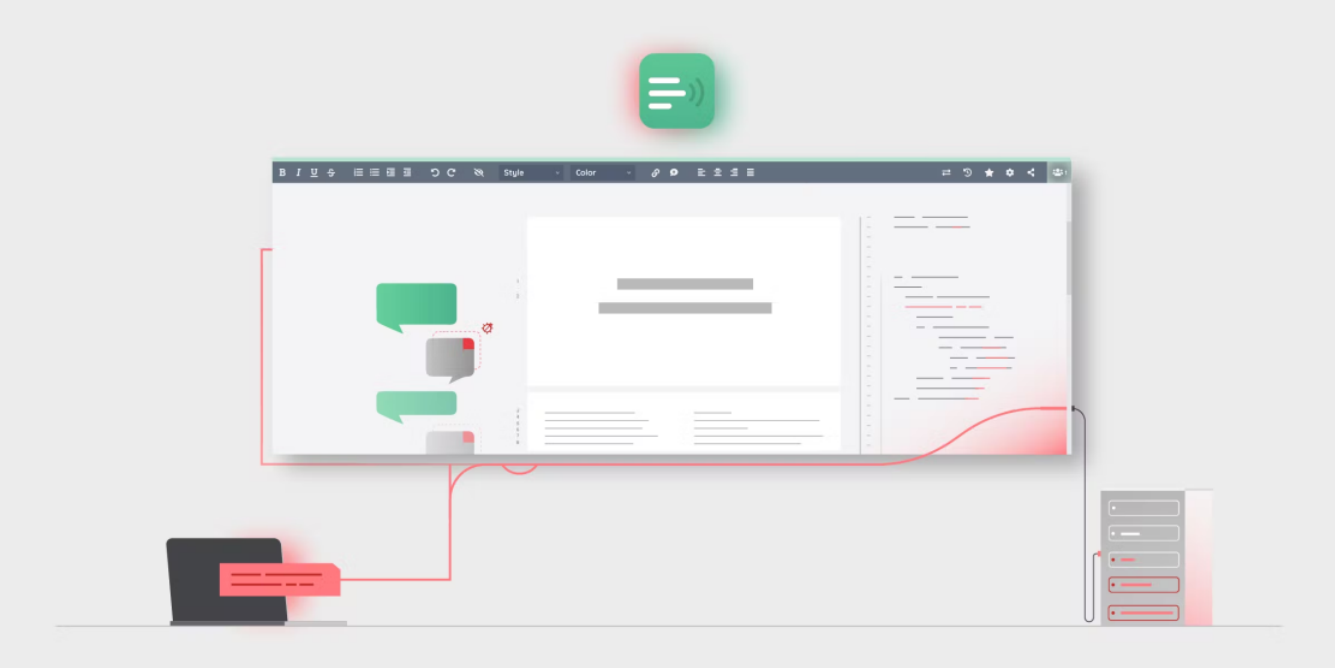
Etherpad 1.8.13 - Code Execution Vulnerabilities

BY PAUL GERSTE | JULY 13, 2021

Security



Etherpad is one of the most popular online text editors that allows collaborating on documents in real-time. It is customizable with more than 250 plugins available and features a version history as well as a chat functionality. There are thousands of instances deployed worldwide with millions of users. The project is very popular within the open-source community as shown by the over 10,000 stars on GitHub. Etherpad instances are often publicly usable and can contain sensitive information.

As part of our security research on open source projects we analyzed Etherpad's code and found two critical vulnerabilities. Both can be combined by an attacker to completely take over an Etherpad instance and its data. In this blog post, we cover the technical details of these code vulnerabilities, show how they were patched, and give advice on how to avoid these types of bugs during development.

Impact

Two injection vulnerabilities were found in Etherpad 1.8.13 that have been present since at least version 1.7.0:

- Cross-Site Scripting (XSS): CVE-2021-34817
- Argument Injection: CVE-2021-34816

The XSS vulnerability allows attackers to take over Etherpad users, including admins. This can be used to steal or manipulate sensitive data. The Argument Injection vulnerability allows attackers to execute arbitrary code on the server, which would allow them to steal, modify or delete all data, or to target other internal systems that are reachable from the server.

Exploiting the XSS vulnerability is possible on any Etherpad instance with a default configuration. The Argument Injection vulnerability requires an admin account to exist, which is not a default setting. Both vulnerabilities can be combined by an attacker to first compromise an admin and then to use these privileges to execute arbitrary code on the server.

A fix for the XSS vulnerability is implemented in Etherpad version 1.8.14. The Argument Injection vulnerability is still unpatched, but it is significantly harder to exploit on its own.

Here is a short demonstration of exploiting both vulnerabilities to get a shell on the server:

Technical Details

In the following technical analysis, we analyze the root cause of the two code vulnerabilities. We first take a look at the XSS vulnerability, then we explain the Argument Injection vulnerability.

Persistent XSS in Chat Messages (CVE-2021-34817)

To allow for better collaboration in a pad, Etherpad offers a chat feature. Here, users can exchange messages in a per-pad group chat. The messages are stored on the server, making the chat history available to everyone.

When a user opens a pad, the chat messages are rendered in the frontend, which involves creating HTML elements from that data. During rendering, the `userId` property of a chat message is inserted into the DOM without properly escaping special characters:

src/static/js/chat.js

```
173    const html =
174        `<p data-authorId='${msg.userId}' …> …` +
175        `<span …`;
176    if (isHistoryAdd) $(html).insertAfter('#chatloadmessagesbutton');
177    else $('#chattext').append(html);
```

In line 174 the `userId` value is used to build a string of HTML markup and in lines 176 and 177 this string is inserted into the DOM. If attackers manage to control a chatter's user ID, then they would be able to insert an XSS payload and perform actions as a victim user. So how can an attacker control a user ID?

Etherpad also features an export/import functionality that handles multiple formats, including a custom JSON-based one. Files in this format can contain a pad's content, its revision history, and all associated chat messages. Such a file can then be used to create a copy of a pad by importing it. An exemplary export file looks like the following:

example.etherpad

```
{
    "pad:1": {
        "chatHead": 0
    },
    "pad:1:chat:0": {
        "text": "Hello World!",
        "userId": "aE45C6209"
    }
}
```

Certain values are validated during the import, but the user IDs of chat messages are used as-is. Since the import feature is enabled by default, an attacker can use this to create a pad that has a chat message with a user ID that consists of arbitrary data.

When that data contains HTML markup, then this markup will be inserted into the DOM, which will execute any inline JavaScript code. As a result, an attacker is able to inject malicious JavaScript code into the chat history which is then executed in an administrator's browser when accessing a pad. This enables an attacker to initiate further attack requests in the browser context of the admin.

Argument Injection in Plugin Management (CVE-2021-34816)

Etherpad also features an admin area that can be used by users that are configured to have the admin role. It allows them to manage plugins, edit settings, and view system information.

When an admin installs a plugin, a message with the name of the plugin is sent to the backend through a WebSocket connection. The backend then installs the NPM package that corresponds to that name:

src/static/js/pluginfw/installer.js

```
49    exports.install = async (pluginName, cb = null) => {
```

In line 56, the plugin name is used as an argument for an `npm install` system command without any validation or sanitization. This allows an attacker to specify a malicious package from the NPM repository or to simply use a URL that points to a package on the attacker's server.

The attacker can either craft a plugin that hooks into Etherpad internals, e.g. creating a backdoor API endpoint, or just use a package with a post-install script which will be executed right after the installation of the package. As a result, the attacker can execute arbitrary code and system commands to fully compromise the Etherpad instance and its data.

To summarize, both vulnerabilities can be chained together in order to first take over an admin's client using the XSS and then gaining access to the server by installing an attacker-controlled plugin.

Patch

In order to mitigate the XSS vulnerability, all values including the userId property should be properly escaped before inserting them into the DOM. This approach was taken by the Etherpad team:

src/static/js/chat.js

```
msg.userId = padutils.escapeHtml(msg.userId);
```

Fixing the Argument Injection is more complex because there is no vetted list of trusted plugins. A first step would be to limit plugin names to valid NPM package names. This prevents the use of URLs and has the benefit that now only packages from the official NPM repository can be installed. However, this is still not a complete fix, because attackers could publish malicious NPM packages.

The optimal solution would be to have a list of trusted plugins and then check a plugin's name against this allowlist before installing it. This is not practical here, as Etherpad is a community project and anybody can contribute plugins.

The Argument Injection vulnerability has not been fixed yet, because it is not really possible as we just saw. However, it is far less likely to be exploited on its own, so fixing the XSS vulnerability also reduced the risk of the Argument Injection. To completely mitigate exploitation, we recommend disabling all admin users and do configuration or plugin management via command-line access.

Timeline

Date	Action
2021-04-06	We report a detailed advisory via email
2021-04-06	Vulnerabilities are confirmed by the vendor
2021-04-08	Vendor fixes the XSS vulnerability
2021-07-04	Vendor releases version 1.8.14 that includes the fix

Summary

In this blog post, we analyzed two code vulnerabilities in Etherpad 1.8.13. The combination of the vulnerabilities can lead to a full compromise of an Etherpad installation. We looked at the vulnerable code snippets and explained how an attacker can exploit them. We also showed how important data validation and sanitization are for avoiding such flaws during development. As we have shown, the smallest coding mistake can be the first stepping stone for an attacker to launch further attacks against the software.

If you are hosting an Etherpad instance and have not updated your installation to version 1.8.14 yet, then we highly recommend that you do so now. Finally, we would like to thank the Etherpad team for their fast response to our initial advisory and their quick fix of the XSS vulnerability.

PAUL GERSTE
Vulnerability Researcher