Securing your GitHub Pages site with HTTPS

In this article

HTTPS adds a layer of encryption that prevents others from snooping on or tampering with traffic to your site. You can enforce HTTPS for your GitHub Pages site to transparently redirect all HTTP requests to HTTPS.

GitHub Pages is available in public repositories with GitHub Free and GitHub Free for organizations, and in public and private repositories with GitHub Pro, GitHub Team, GitHub Enterprise Cloud, and GitHub Enterprise Server. For more information, see "GitHub's plans."

People with admin permissions for a repository can enforce HTTPS for a GitHub Pages site.

About HTTPS and GitHub Pages

All GitHub Pages sites, including sites that are correctly configured with a custom domain, support HTTPS and HTTPS enforcement. For more information about custom domains, see "About custom domains and GitHub Pages" and "Troubleshooting custom domains and GitHub Pages."

GitHub Pages sites shouldn't be used for sensitive transactions like sending passwords or credit card numbers.

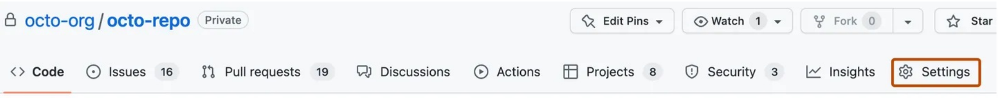
Warning: Unless your enterprise uses Enterprise Managed Users, GitHub Pages sites are publicly available on the internet by default, even if the repository for the site is private or internal. You can publish a site privately by managing access control for the site. Otherwise, if you have sensitive data in your site's repository, you may want to remove the data before publishing. For more information, see "About repositories" and "Changing the visibility of your GitHub Pages site."

Note: RFC3280 states that the maximum length of the common name should be 64 characters. Therefore, the entire domain name of your GitHub Pages site must be less than 64 characters long for a certificate to be successfully created.

Enforcing HTTPS for your GitHub Pages site

1. On GitHub Enterprise Cloud, navigate to your site's repository.

2. Under your repository name, click ⚙ **Settings**. If you cannot see the "Settings" tab, select the ··· dropdown menu, then click **Settings**.



3. In the "Code and automation" section of the sidebar, click **Pages**.
4. Under "GitHub Pages," select **Enforce HTTPS**.

Troubleshooting certificate provisioning ("Certificate not yet created" error)

When you set or change your custom domain in the Pages settings, an automatic DNS check begins. This check determines if your DNS settings are configured to allow GitHub to obtain a certificate automatically. If the check is successful, GitHub queues a job to request a TLS certificate from Let's Encrypt. On receiving a valid certificate, GitHub automatically uploads it to the servers that handle TLS termination for Pages. When this process completes successfully, a check mark is displayed beside your custom domain name.

Please note that your GitHub Pages site must be publicly available for a Let's Encrypt certificate to be issued. Once the certificate has been issued you may revert the site to private.

The process may take some time. If the process has not completed several minutes after you clicked **Save**, try clicking **Remove** next to your custom domain name. Retype the domain name and click **Save** again. This will cancel and restart the provisioning process.

Resolving problems with mixed content

If you enable HTTPS for your GitHub Pages site but your site's HTML still references images, CSS, or JavaScript over HTTP, then your site is serving *mixed content*. Serving mixed content may make your site less secure and cause trouble loading assets.

To remove your site's mixed content, make sure all your assets are served over HTTPS by changing `http://` to `https://` in your site's HTML.

Assets are commonly found in the following locations:

- If your site uses Jekyll, your HTML files will probably be found in the *_layouts* folder.
- CSS is usually found in the `<head>` section of your HTML file.
- JavaScript is usually found in the `<head>` section or just before the closing `</body>` tag.
- Images are often found in the `<body>` section.

Tip: If you can't find your assets in your site's source files, try searching your site's source files for `http` in your text editor or on GitHub Enterprise Cloud.

Examples of assets referenced in an HTML file

Asset type	HTTP	HTTPS
CSS	`<link rel="stylesheet"`	`<link rel="stylesheet"`

CSS	`<link rel="stylesheet" href="http://example.com/css/main.css">`	`<link rel="stylesheet" href="https://example.com/css/main.css">`
JavaScript	`<script type="text/javascript" src="http://example.com/js/main.js"></script>`	`<script type="text/javascript" src="https://example.com/js/main.js"></script>`
Image	`<a href="http://www.somesite.com"><img src="http://www.example.com/logo.jpg" alt="Logo"></a>`	`<a href="https://www.somesite.com"><img src="https://www.example.com/logo.jpg" alt="Logo"></a>`

Legal

Terms Privacy Status Pricing Expert services Blog