Troubleshooting webhooks

In this article

Learn how to diagnose and resolve common errors for webhooks.

Missing webhook deliveries

If you are not receiving the webhook deliveries that you expect, you should identify the point at which the delivery is missing.

1. Trigger an event that you expect to result in a webhook delivery. For example, if your webhook is a repository webhook that is subscribed to the `issues` event, you can open an issue on that repository.
2. Look at the recent deliveries log for your webhook. For information about how to do this for each webhook type, see "Viewing webhook deliveries."

   If the recent deliveries log does not include a delivery that corresponds to the webhook event that you triggered in the previous step, then GitHub did not attempt a delivery. To identify the cause:

   a. Wait a few minutes, and then check again. Webhook deliveries can take a few minutes to appear.

   b. Make sure that you triggered an event in the location where your webhook is configured. For example, if your webhook is a repository webhook, make sure that you triggered the event in the same repository where your webhook is configured.

   c. Make sure that your webhook is subscribed to the event that you triggered. For example, if you expect a webhook delivery when you open an issue, make sure your webhook is subscribed to the `issues` event.

   d. Make sure that your webhook is active. For more information, see "Disabling webhooks."

e. Make sure that your webhook is not impacted by OAuth app access restrictions. If your webhook was created by an OAuth app on behalf of a user who authorized the OAuth app, the webhook will be automatically disabled if it is an organization or repository webhook for an organization that has restricted access by the OAuth app. For more information, see "About OAuth app access restrictions."

f. Check whether your event may have hit a documented limit. For example, if you push more than three tags at once, the `push` event will not be triggered for that push. For more information about documented limits for each event, see "Webhook events and payloads."

g. Check the status of webhooks at githubstatus.com.

If the recent deliveries log indicates that there was an error with the delivery, then GitHub attempted delivery but the delivery was unsuccessful. This is typically due to a problem with your server. You can refer to the sections below to help resolve the specific error.

3. Look at the logs for your server. The information in the logs depends on the code that your server runs to handle webhook deliveries. To help you diagnose problems on your server, you may want to add additional log statements to your code.

Cannot have more than 20 webhooks

You can create up to 20 repository, organization, or global webhooks for each event type. If you attempt to create more, you will receive an error stating that you cannot have more than 20 webhooks.

If you require more than 20 webhooks, you can run a proxy that receives webhooks from GitHub and forwards them to an unlimited number of destination URLs.

URL host localhost is not supported

You cannot use `localhost` or `127.0.0.1` as a webhook URL.

To deliver webhooks to your local server for testing, you can use a webhook forwarding service. For more information, see "Testing webhooks" or visit https://smee.io/.

Failed to connect to host

The `failed to connect to host` error occurs when GitHub attempts a webhook delivery but could not resolve the webhook's URL to an IP address.

To check whether a host name resolves to an IP address, you can use `nslookup`. For example, if your payload URL is `https://octodex.github.com/webhooks`, you can run `nslookup octodex.github.com`. If the host name could not be resolved to an IP address, the nslookup command will indicate that the server can't find the host name.

Failed to connect to network

The `failed to connect to network` error indicates that your server refused the connection when GitHub attempted to deliver a webhook.

You should make sure that your server allows connections from GitHub's IP addresses.

You can use the `GET /meta` endpoint to find the current list of GitHub's IP addresses. For more information, see "Meta." GitHub occasionally makes changes to its IP addresses, so you should update your IP allow list periodically.

Timed out

The `timed out` error indicates that GitHub did not receive a response from your server within 10 seconds of delivering a webhook.

Your server should respond with a 2xx response within 10 seconds of receiving a webhook delivery. If your server takes longer than that to respond, then GitHub terminates the connection and considers the delivery a failure.

In order to respond in a timely manner, you may want to set up a queue to process webhook payloads asynchronously. Your server can respond when it receives the webhook, and then process the payload in the background without blocking future webhook deliveries. For example, you can use services like Hookdeck or libraries like Resque (Ruby), RQ (Python), or RabbitMQ.

Peer certificate cannot be authenticated with given CA certificates

This error indicates that there is a problem related to your server's certificates. The most common problems are:

- Your server is using a self-signed certificate.
- Your server is not sending the full certificate chain when the connection is established.

To help diagnose the problem, you can use the SSL server test from SSL Labs. This service can only work with the default port for HTTPS (port 443) and can only work with servers that are accessible from the Internet.

You can also use `openssl` to help diagnose the problem. To do so, run `openssl s_client -connect HOST:PORT` in a terminal. Replace `HOST` with your server's host name and `PORT` with the port. For example, `openssl s_client -connect example.com:443`. To identify problems, look for `verify error` in the output.

Invalid HTTP response

The `invalid HTTP response` error occurs when your server returns a 4xx or 5xx status in response to a webhook delivery from GitHub.

You should configure your server to return a 2xx status. If your server returns a 4xx or 5xx status, GitHub will record the delivery as a failure.

Webhooks deliveries are out of order

GitHub may deliver webhooks in a different order than the order in which the events took place. If you need to know when the event occurred relative to another event, you should use the timestamps that are included in the delivery payload.

Webhook deliveries are not immediate

Webhook deliveries can take a few minutes to be delivered and to appear in the recent deliveries log. Before concluding that your webhook delivery failed, wait a few minutes

and then check again.

Failed signature verification

You should use a webhook secret and the `X-Hub-Signature-256` header to verify that a webhook delivery is from GitHub. For more information, see "Validating webhook deliveries."

If you are sure that the payload is from GitHub but the signature verification fails:

- Make sure that you have configured a secret for your webhook. The `X-Hub-Signature-256` header will not be present if you have not configured a secret for your webhook. For more information about configuring a secret for your webhook, see "Editing webhooks."
- Make sure you are using the correct header. GitHub recommends that you use the `X-Hub-Signature-256` header, which uses the HMAC-SHA256 algorithm. The `X-Hub-Signature` header uses the HMAC-SHA1 algorithm and is only included for legacy purposes.
- Make sure that you are using the correct algorithm. If you are using the `X-Hub-Signature-256` header, you should use the HMAC-SHA256 algorithm.
- Make sure you are using the correct webhook secret. If you don't know the value of your webhook secret, you can update your webhook's secret. For more information, see "Editing webhooks."
- Make sure that the payload and headers are not modified before verification. For example, if you use a proxy or load balancer, make sure that the proxy or load balancer does not modify the payload or headers.
- If your language and server implementation specifies a character encoding, ensure that you handle the payload as UTF-8. Webhook payloads can contain unicode characters.