Best practices for using webhooks

In this article

Follow these best practices to improve security and performance when using webhooks.

Subscribe to the minimum number of events

You should only subscribe to the webhook events that you need. This will reduce the amount of work your server needs to do. For more information about subscribing to events, see "Creating webhooks" and "Editing webhooks."

Use a webhook secret

You should set a webhook secret for your webhook and verify that the signature of each webhook delivery matches the secret. This helps to ensure that the webhook delivery is from GitHub. For more information, see "Validating webhook deliveries."

The webhook secret should be a random string of text with high entropy. You should securely store your webhook secret in a way that your server can access.

Use HTTPS and SSL verification

You should ensure that your server uses an HTTPS connection. By default, GitHub will verify SSL certificates when delivering webhooks. GitHub recommends that you leave SSL verification enabled.

Allow GitHub's IP addresses

You can set up an IP allow list for your server, and add the IP addresses that GitHub uses for webhook deliveries. This can block spoofed requests to your server.

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.

For more information, see "About GitHub's IP addresses."

Respond within 10 seconds

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 (Java).

Check the event type and action before processing the event

There are multiple webhook event types, and many events can have multiple action types. GitHub continues to add new event types and new actions to existing event types. Your application should check the event type and action of a webhook payload before processing the payload. To determine the event type, you can use the `X-GitHub-Event` request header. To determine the action type, you can use the top-level `action` key in the event payload.

Redeliver missed deliveries

If your server goes down, you should redeliver missed webhooks once your server is back up. For more information, see "Redelivering webhooks."

Use the `X-GitHub-Delivery` header

In a replay attack, a bad actor intercepts a webhook delivery and re-sends the delivery. To protect against replay attacks, you can use the `X-GitHub-Delivery` header to ensure that each delivery is unique.

Note: If you request a redelivery, the `X-GitHub-Delivery` header will be the same as in the original delivery.

Further reading

- "Best practices for using the REST API"
- "Best practices for creating a GitHub App"