Worker subrequests

- Why are the origin fields empty on the initial request when a Worker makes a subrequest?
- What the two log entries represent
- How the two entries are linked
- Example
- When there is no second log entry
- Recommended fields to include in Logpush

❮ Back to FAQ

Why are the origin fields empty on the initial request when a Worker makes a subrequest?

When a request hits a zone with a Worker, the initial HTTP request log represents the end-user request to the Worker. If that Worker later makes a `fetch()` request to your origin, Cloudflare writes a second HTTP request log for that Worker subrequest.

Because the initial log entry only covers the request from the client to the Worker, `OriginResponseStatus` is `0` and `OriginIP` is empty on that entry. Those fields are populated on the second log entry, which represents the Worker-to-origin fetch. For more on what `OriginResponseStatus=0` means in different contexts, refer to 504 responses with origin status 0 in Logpush.

What the two log entries represent

Log entry	Typical `ClientRequestSource` value	What it represents	Origin fields
Initial request	`eyeball`	End-user request to the Worker	`OriginResponseStatus` is `0`, `OriginIP` is empty

Log entry	Typical `ClientRequestSource` value	What it represents	Origin fields
Worker subrequest	`edgeWorkerFetch`	Worker `fetch()` request to the origin	Set when the origin is contacted

Refer to ClientRequestSource field for the full list of possible `ClientRequestSource` values.

How the two entries are linked

Each log entry has its own `RayID`. The Worker subrequest also includes `ParentRayID`, which is the `RayID` of the request that triggered it — its immediate parent.

Field	Initial request	Worker subrequest
`RayID`	Unique request ID for the end-user request	Unique request ID for the subrequest
`ParentRayID`	Empty	`RayID` of the request that triggered it

To correlate the two records:

1. Find the initial request and note its `RayID`.
2. Search for log entries where `ParentRayID` equals that `RayID`.
3. Review the matching subrequest log entry for `OriginIP`, `OriginResponseStatus`, and other origin fields.

One end-user request can produce multiple Worker subrequest log entries. Each subrequest gets its own `RayID`, and each of those log entries uses the triggering request's `RayID` as its `ParentRayID`.

`ParentRayID` is single-level — it points to the immediate parent, not the original end-user request. In a single-Worker setup this distinction does not matter because the parent is the end-user request. When Workers are chained (for example, Worker A calls Worker B which calls the origin), Worker B's subrequest log has Worker A's `RayID` as its `ParentRayID`, not the end-user's. To trace the full chain back to the end-user request, follow each `ParentRayID` one level at a time.

Using the initial request together with the matching subrequest entries lets you reconstruct the full request path from client to Worker to origin.

Example

```
# Initial request
ClientRequestSource: eyeball
RayID: 7b52f2c4f9f64c1a
```

```
ParentRayID:
OriginResponseStatus: 0

# Worker subrequest
ClientRequestSource: edgeWorkerFetch
RayID: 7b52f2c4f9f64c1b
ParentRayID: 7b52f2c4f9f64c1a
OriginResponseStatus: 200
OriginIP: 192.0.2.10
```

When there is no second log entry

If the Worker does not make a `fetch()` request to the origin, there is no Worker-to-origin log entry to correlate. For example, the Worker may return a response directly without contacting the origin.

Recommended fields to include in Logpush

To investigate Worker subrequests more easily, include these fields in your HTTP request logs:

- `RayID`
- `ParentRayID`
- `ClientRequestSource`
- `OriginIP`
- `OriginResponseStatus`

□□□ #2
□ □□□ □□□ 2026-11-01 02:52:19 UTC
□ □□□ □□□ 2026-11-01 02:52:21 UTC