CHATGPT

Move the nightly billing job to a Cloudflare Worker

September 22, 2026 · 4 messages · chatgpt.com/c/sample

YOU

We run a nightly billing script with node-cron on a small VPS. It retries failed invoices three times with a growing delay. I want to move it to a Cloudflare Worker cron trigger and keep the retry behaviour. What changes, and what does the handler look like?

CHATGPT

What changes when you move to a Worker cron

Three things are different from a long-running server, and each one affects how you write the retry loop:

1. **No process to keep alive.** The Worker starts on the schedule, runs your handler and exits. State between runs has to live in KV, D1 or Durable Objects.
2. **A wall-clock limit per invocation.** Scheduled handlers get 30 seconds of CPU time on the paid plan, so retries with long sleeps must be spread across invocations, not awaited inside one.
3. **Retries are yours.** Cloudflare does not re-run a cron that threw. If you want three attempts, you schedule them.

The handler

```js
export default {
  async scheduled(event, env, ctx) {
    const failed = await runBilling(env);          // returns invoice ids that failed
    if (failed.length) {
      await env.RETRY_QUEUE.send({ ids: failed, attempt: 1 });
    }
  },
};
```

The retry itself moves to a **Queue consumer**, which gives you the growing delay for free:

```js
export async function queue(batch, env) {
  for (const msg of batch.messages) {
    const { ids, attempt } = msg.body;
    const stillFailed = await retryInvoices(ids, env);
    if (stillFailed.length && attempt < 3) {
      msg.retry({ delaySeconds: 60 * 2 ** attempt });   // 2, 4, 8 minutes
    } else {
      msg.ack();
    }
```

```
  }
}
```

Before and after

Concern	VPS + node-cron	Worker cron + Queue
Scheduling	Process must stay up	`[triggers] crons` in wrangler.toml
Retries	`setTimeout` loop in-process	`msg.retry({ delaySeconds })`
State	Local file or Postgres	KV or D1
Cost at this volume	Fixed monthly VPS	Included in Workers Paid

> Keep the first migration boring: same invoice logic, same idempotency key per invoice, new scheduler. Change one thing at a time.

Reference: Cron Triggers and Queues in the Cloudflare docs.

YOU

What about the invoices that fail all three times? Today they end up in a log file nobody reads.

CHATGPT

Give them a **dead-letter queue** and a human. In `wrangler.toml` point the consumer at a second queue with `dead_letter_queue = "billing-dlq"`; after the third failed attempt Cloudflare moves the message there automatically, so you never write that branch yourself.

Then make the dead-letter queue visible instead of quiet:

- A tiny consumer on `billing-dlq` that writes each invoice id to a D1 table `billing_failures` with the last error message.
- A scheduled Worker at 08:00 that emails the list to finance if the table is not empty. One email a day, only on bad days.
- A `/failures` route behind Cloudflare Access so someone can mark an invoice as handled.

That replaces the log file with a queue that cannot be ignored, and it costs nothing extra at your volume.