

When an external service goes down

Timeout, retry and circuit breaker: how three lines of defense stop a slow external service from locking up your whole application.

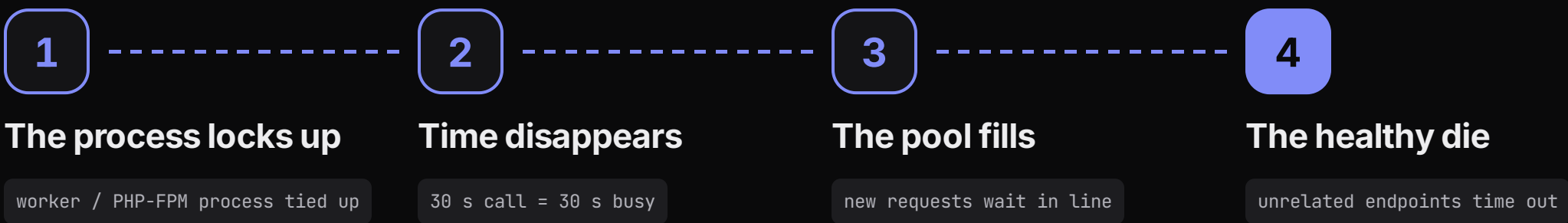
01 NOT DOWN, JUST SLOW

An SMS provider slowed down. The whole application stopped.

Response time climbed to 30 seconds; within half an hour even endpoints with nothing to do with SMS were timing out.

02 CASCADING FAILURE

How slowness spreads



03 FIRST DEFENSE: TIMEOUT

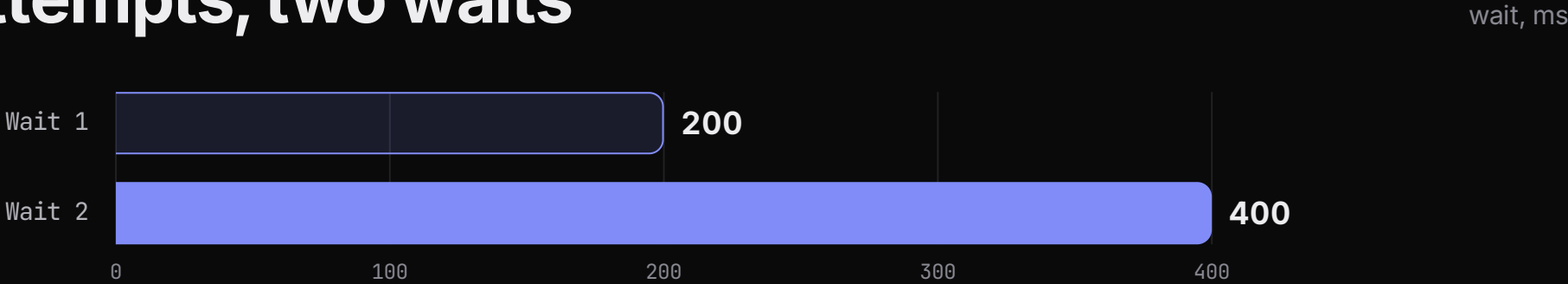
5 s

total timeout

Not the default 30 seconds, but 5: `Http::timeout(5)->connectTimeout(2)`. The values are the source's suggested example; this number should be a decision.

04 SECOND DEFENSE: BOUNDED RETRY

Three attempts, two waits



Example values: $200 \times 2^{n-1}$ ms, plus `random_int(0, 100)` ms of jitter. Three attempts have two waits between them; if errors persist, the problem isn't transient.

05 PUT A CEILING ON BACKOFF

~3.4 min

12th wait without a cap

$\text{delay} = \min(\text{cap}, \text{base} \times \text{factor}^{n-1})$. Without a cap, base 100 ms $\times 2$ reaches about 3.4 minutes by the 12th attempt.

06 THUNDERING HERD

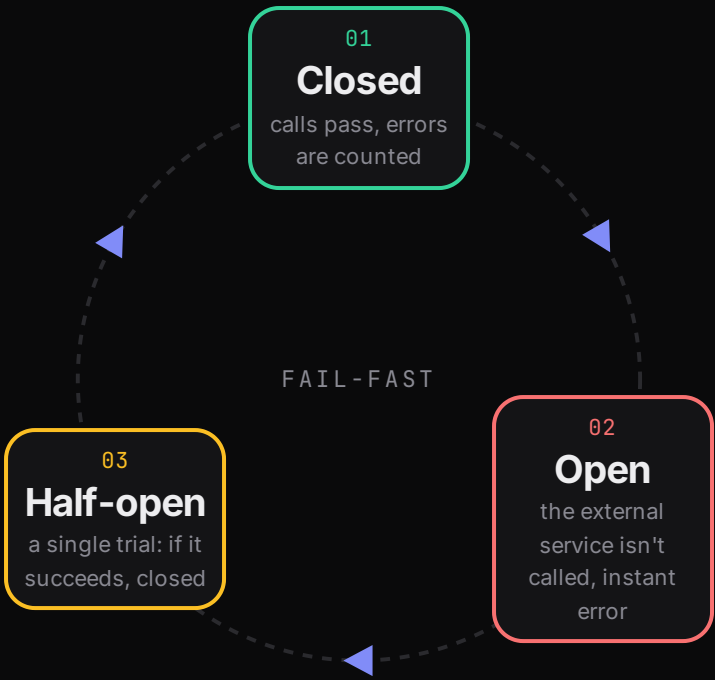
Jitter types

TYPE	RANGE	BEHAVIOR
Full	[0, delay]	Widest spread; one client may retry very early
Equal	[delay/2, delay]	Keeps a floor, still spreads the herd
Decorrelated	[base, previous × 3]	Capped; follows the last wait, a strong default
None	no jitter	Fine for a single client, dangerous for a fleet

07 THIRD DEFENSE: CIRCUIT BREAKER

Stop making the call

Even a 5-second timeout is 5 seconds per process; in the open state that drops to 0.



08 FAIL-FAST BEATS SLOW-FAIL

What you do while the circuit is open

↓

Queue it
If the SMS doesn't have to go out right away, leave it to be processed once the circuit closes.

ⓘ

Serve stale data
If the exchange-rate service is down, show the last known rate.

⚠

Degrade gracefully
Saying "Recommendations can't load right now" beats crashing the page.