

Configuration and Deployment

Configuration and Deployment

Anonymous · 10/10/2026

Configuration and Deployment

Environment variables

Lambda function configuration

Set these on the Lambda function's configuration, for example:

```
$ aws lambda update-function-configuration \
  --function-name email2webhook \
  --environment '{
    "Variables": {
      "SES_S3_BUCKET": "email-inbox-123456789012-us-east-2-an",
      "SES_S3_KEY_PREFIX": "inbound/",
      "WEBHOOK_URL": "https://example.com/webhooks/email2webhook",
      "WEBHOOK_SECRET": "replace-with-a-real-secret",
      "WEBHOOK_FORMAT": "postal",
      "WEBHOOK_SIGNATURE": "hmac"
    }
  }'
```

- `SES_S3_BUCKET` — name of the S3 bucket where the receipt rule stores raw emails.
- `SES_S3_KEY_PREFIX` — optional object key prefix, only needed if the S3 action's "Object key prefix" field was set to a non-empty value; it must match that value exactly, **including the trailing slash** if you want it to appear as a folder in the S3 console (e.g. `inbound/`), since S3 has no real folders — a `/` in the key is just a display convention (default: none).
- `WEBHOOK_URL` — URL the envelope is POSTed to. Required unless `WEBHOOK_DEBUG_URL` is set (debug-only mode).
- `WEBHOOK_DEBUG_URL` — optional second URL that gets an identical, best-effort copy of the same signed request, sent before `WEBHOOK_URL` — for temporarily watching traffic (e.g. a debug endpoint) without touching `WEBHOOK_URL`. Failures here (bad status, timeout, connection error) are logged and never raised, so a broken or slow debug endpoint can never block or fail the real `WEBHOOK_URL` request. If `WEBHOOK_URL` isn't set, the function runs in debug-only mode and sends only here.
- `WEBHOOK_SECRET` — shared secret used to sign the request body when `WEBHOOK_SIGNATURE=hmac` (the default).

- `WEBHOOK_FORMAT` — `postal` (default) or `ses`. See “Output formats” above.
- `WEBHOOK_SIGNATURE` — `hmac` (default) or `none`. See “Signing” above.
- `WEBHOOK_SIGNATURE_PREFIX` — literal string prepended to the hex digest in `X-Webhook-Signature-256` when `WEBHOOK_SIGNATURE=hmac` (default: `none`, just the hex digest).
- `ATTACHMENT_REQUIRE_EXTENSIONS` — comma-separated list of attachment extensions (with or without the leading dot, case-insensitive, e.g. `xml` or `.xml`, `.txt`). If set, an email is only processed (and sent to the webhook at all) when it has at least one attachment matching one of these extensions; otherwise it’s discarded silently. Applies to both `WEBHOOK_FORMAT` values. Default: empty, no filtering.
- `ATTACHMENT_ALLOW_EXTENSIONS` — same syntax as above. If set, only matching attachments are included in the `postal` format’s `attachments` list (non-matching ones are dropped, but the email itself is still sent); has no effect on the `ses` format, which embeds the raw email as-is. Default: empty, all attachments included.

Local testing only

Never set these in the deployed Lambda function; they exist only to run `lambda_function.py` directly on a workstation, without AWS credentials or a live webhook:

- `SES_LOCAL_TEST_EML_FILE` — path to a local `.eml` file; when set, `get_raw_email` reads from this file instead of calling S3.
- `SES_LOCAL_TEST_MESSAGE_ID` — `messageId` value used to build the mock SES event in the `__main__` block (default: `local-test-message-id`).

Deploying

`make all` installs the runtime dependencies listed in `pyproject.toml`’s `[project.dependencies]` into `function/` (`clean` runs first, so no stray files from a previous build or a local test run end up bundled), then zips `function/` into `aws-lambda-email2webhook.zip`. `make clean` removes the zip and everything under `function/` except `lambda_function.py`, restoring it to its checked-in state. `make dev` creates a local `.venv` with the dev extras (`ruff`, `mypy`, `boto3`, `requests`, plus type stubs) for linting, type-checking and running the function locally — none of that ever gets bundled into the zip.

`boto3` is deliberately not in `[project.dependencies]` — the Lambda Python runtime already provides it, adding it would only bloat the deployment package. It’s only listed under the dev extra, to have it available locally for type-checking and for running `lambda_function.py` directly (see “Local testing only” above).

The Makefile pins `python3.14` to build the package, matching the Lambda runtime’s version, so there’s no risk of a compiled extension (`requests`’s `charset_normalizer` is one) being built against a mismatched Python/OS ABI.

Once you have the zip, upload it — the first time, create the function; afterwards, update its code:

```
$ aws lambda update-function-code \  
  --function-name email2webhook \  
  --zip-file fileb://aws-lambda-email2webhook.zip
```

Verifying the webhook signature

The receiving end must recompute `hmac_sha256(WEBHOOK_SECRET, raw_request_body)` and compare it (constant-time) against the value in `X-Webhook-Signature-256`. By default that value is just the hex digest, nothing to strip; if `WEBHOOK_SIGNATURE_PREFIX` is set, strip that literal string from the front before comparing.

Last updated on 10/10/2026