

## 02 — Authentication (HMAC)

**Audience:** Tenant IT. **Prasyarat:** [01-overview.md](#).

Semua endpoint write yang Anda akan call (`POST /api/v1/tickets/:id/reply`, `POST /api/v1/callbacks/:submission_id`) DAN semua endpoint Anda yang EMBAN call (ticket webhook, action endpoint) diautentikasi pakai **HMAC-SHA256 signature**. Tidak ada OAuth, tidak ada bearer token. Pola ini mengikuti konvensi Stripe/Slack — sederhana, stateless, dan tahan terhadap replay selama jam NTP-sinkron.

---

### 1. Dua jenis secret

Anda akan memegang dua kategori secret. Masing-masing hanya dipakai untuk satu golongan endpoint.

Nama logis	Scope	Dari mana didapat	Dipakai di
<b>Ticket webhook secret</b>	per-tenant	disepakati saat tenant CS onboard; EMBAN menyimpan nama env var di <code>tenants.ticket_webhook_secret_env</code>	outbound <code>ticket.created</code> webhook + inbound <code>POST /api/v1/tickets/:id/reply</code>
<b>Action callback secret</b>	per-action, per-tenant	dikonfigurasi saat Anda mendaftarkan custom action ( <code>endpoint.auth.secret_env</code> )	outbound action endpoint + inbound <code>POST /api/v1/callbacks/:submission_id</code>

Catatan implementasi: EMBAN menyimpan hanya **nama env var** di DB — secret aslinya hidup di `process.env` di runtime EMBAN. Anda cukup tahu nilai secret-nya (yang dipakai untuk sign), tidak perlu tahu nama env var di sisi EMBAN.

---

## 2. Dua skema tanda tangan

Berbeda dengan banyak API yang pakai satu skema HMAC saja, EMBAN pakai **dua skema** tergantung kategori endpoint. Penting: pilih skema yang benar untuk endpoint yang Anda implement, kalau salah signature 100% tidak match.

### 2.1 Ts-prefixed scheme (default)

Dipakai oleh: ticket webhook (#4), ticket reply, callback async action (#3), file delivery (#6).

Canonical signing input:

<timestamp\_ms> "." <raw\_request\_body>

- `timestamp_ms` — current time sebagai **Unix millis** (string angka desimal, bukan float, bukan detik).
- `raw_request_body` — **string body persis seperti yang dikirim di wire**. Kalau Content-Type `application/json`, artinya exact bytes dari `JSON.stringify(...)` yang akan dikirim. **Jangan whitespace-tweak, jangan re-stringify.**
- Untuk request tanpa body, `raw_request_body` = string kosong (`" "`).

Signature:

`signature = "sha256=" + hex( HMAC_SHA256(secret, signing_input) )`

`hex(...)` = lowercase hex, **tanpa pemisah**, tanpa uppercase. Prefix literal `sha256=` wajib.

### 2.2 Body-only scheme (Action Engine outbound)

Dipakai oleh: outbound action endpoint (#5 §3) — saja.

Canonical signing input:

<raw\_request\_body>

(Tidak ada timestamp prefix.)

Signature dihitung dengan rumus yang sama:

`signature = "sha256=" + hex( HMAC_SHA256(secret, raw_request_body) )`

**Kenapa skip timestamp?** Engine kontrol retry sendiri dengan **Idempotency-Key** (= **submission\_id**). Tidak ada risk replay dari pihak ketiga karena tenant set secret-nya di env, EMBAN yang outbound. Untuk inbound (callback Anda balik ke EMBAN), tetap pakai ts-prefixed (§2.1).

---

### 3. Header wajib

Header yang Anda kirim saat outbound (callback / reply ke EMBAN):

Header	Nilai	Contoh
X-ASPRI-Timestamp	timestamp dalam Unix millis (string) — <b>ts-prefixed scheme saja</b>	1745145600123
X-ASPRI-Signature	sha256=<hex-digest>	sha256=9f3a1c...
Content-Type	application/json	wajib untuk POST berbody
Idempotency-Key	opsional tapi <b>sangat direkomendasikan</b> untuk semua POST	ULID / UUID / key unik lain, < 128 char

Header yang EMBAN kirim saat outbound (ticket webhook / action endpoint Anda):

Header	Nilai	Catatan
X-ASPRI-Signature	sha256=<hex-digest>	selalu ada
X-ASPRI-Timestamp	Unix millis	hanya pada ts-prefixed scheme (ticket webhook)
X-ASPRI-Tenant-Id	tenant ULID terkait	identifikasi tenant
X-ASPRI-Submission-Id	ULID	hanya untuk Action Engine outbound
X-ASPRI-Schema-Version	mis. "1.0"	hanya untuk Action Engine, sesuai <b>schema_version</b> action def

Header	Nilai	Catatan
Idempotency-Key	ULID (= <code>ticket.id / submission_id</code> )	pakai untuk dedup di sisi Anda

---

## 4. Toleransi skew & replay

Berlaku hanya untuk ts-prefixed scheme:

- Default window: **5 menit** (`API_HMAC_TIMESTAMP_SKEW_MS=300000`).
- Jika `|now - X-ASPRI-Timestamp| > window` → 401 `timestamp_skew`.
- Implikasi: **jam server Anda wajib tersinkron NTP**. Drift > 5 menit = semua request ditolak.
- Tidak ada nonce/anti-replay kedua. Window timestamp sudah cukup kecil untuk mencegah replay; idempotency-key mencegah efek samping duplikat di layer aplikasi.

Body-only scheme tidak melakukan skew check — engine kontrol retry sendiri.

---

## 5. Kode contoh — Node.js

### 5.1 Sign outbound (callback / reply ke EMBAN — ts-prefixed)

```
import { createHmac, randomUUID } from "node:crypto";

function signRequest(body, secret) {

  const ts = Date.now();

  const raw = typeof body === "string" ? body : JSON.stringify(body);

  const sig =

    "sha256=" +

    createHmac("sha256", secret).update(`${ts}.${raw}`).digest("hex");

  return { ts, raw, sig };
```

```

}

async function replyToTicket(baseUrl, ticketId, secret, payload) {

  const { ts, raw, sig } = signRequest(payload, secret);

  const res = await fetch(`${baseUrl}/api/v1/tickets/${ticketId}/reply`, {

    method: "POST",

    headers: {

      "Content-Type": "application/json",

      "X-ASPRI-Timestamp": String(ts),

      "X-ASPRI-Signature": sig,

      "Idempotency-Key": randomUUID(),

    },

    body: raw,          // kirim string yang SAMA dengan yang di-sign

  });

  return res.json();

}

```

**Pitfall:** jangan lakukan `body: payload` (object) — framework fetch/axios akan re-stringify-kan secara internal dengan urutan key yang berbeda, membuat signature tidak cocok. Sign dulu, kirim string yang sama persis.

## 5.2 Verify inbound (ticket webhook / action endpoint dari EMBAN — ts-prefixed)

```

import express from "express";

import { createHmac, timingSafeEqual } from "node:crypto";

const SECRET = process.env.ASPRI_TICKET_WEBHOOK_SECRET;

```

```

const SKEW_MS = 5 * 60 * 1000;

app.use("/aspri/tickets", express.raw({ type: "application/json" }));

app.post("/aspri/tickets", (req, res) => {

  const ts = Number(req.header("X-ASPRI-Timestamp"));

  const sig = req.header("X-ASPRI-Signature") ?? "";

  if (!Number.isFinite(ts) || Math.abs(Date.now() - ts) > SKEW_MS) {

    return res.status(401).send("timestamp skew");

  }

  const raw = req.body.toString("utf8");

  const expected =

    "sha256=" +

    createHmac("sha256", SECRET).update(`${ts}.${raw}`).digest("hex");

  const a = Buffer.from(sig);

  const b = Buffer.from(expected);

  if (a.length !== b.length || !timingSafeEqual(a, b)) {

    return res.status(401).send("bad signature");

  }

  // ... process payload

});

```

### 5.3 Verify inbound body-only (action endpoint outbound dari EMBAN)

Lebih sederhana — tidak ada timestamp / skew check.

```

app.post("/aspri/action", (req, res) => {

```

```

const sig = req.header("X-ASPRI-Signature") ?? "";

const raw = req.body.toString("utf8");

const expected =

    "sha256=" +

        createHmac("sha256", SECRET).update(raw).digest("hex");

const a = Buffer.from(sig);

const b = Buffer.from(expected);

if (a.length !== b.length || !timingSafeEqual(a, b)) {

    return res.status(401).send("bad signature");

}

// ... process action payload

});

```

---

## 6. Kode contoh — Python

### 6.1 Sign outbound (ts-prefixed)

```
import hmac, hashlib, json, time, uuid, requests
```

```
def sign_request(body, secret: str):
```

```
    ts = int(time.time()) * 1000)
```

```
    raw = body if isinstance(body, str) else json.dumps(body, separators=(",", ":"))
```

```
    sig = "sha256=" + hmac.new(
```

```
        secret.encode(),
```

```

        f'{ts}.{raw}'.encode(),

        hashlib.sha256,

    ).hexdigest()

    return ts, raw, sig

def reply_to_ticket(base_url: str, ticket_id: str, secret: str, payload: dict):

    ts, raw, sig = sign_request(payload, secret)

    resp = requests.post(

        f'{base_url}/api/v1/tickets/{ticket_id}/reply',

        data=raw,          # kirim string, bukan json=

        headers={

            "Content-Type": "application/json",

            "X-ASPRI-Timestamp": str(ts),

            "X-ASPRI-Signature": sig,

            "Idempotency-Key": str(uuid.uuid4()),

        },

        timeout=10,

    )

    return resp.json()

```

**Pitfall:** `requests.post(url, json=payload)` me-reserialize dengan whitespace default `" , "` — signature tidak akan cocok. Pakai `data=raw` dengan string yang SAMA dengan yang di-sign.



## 6.2 Verify inbound (Flask — ts-prefixed)

```
import hmac, hashlib, time, json, os

from flask import Flask, request, abort

SECRET = os.environ["ASPRI_TICKET_WEBHOOK_SECRET"].encode()

SKEW_MS = 5 * 60 * 1000

app = Flask(__name__)

@app.post("/aspri/tickets")

def handle_ticket():

    ts_hdr = request.headers.get("X-ASPRI-Timestamp", "")

    sig_hdr = request.headers.get("X-ASPRI-Signature", "")

    try:

        ts = int(ts_hdr)

    except ValueError:

        abort(401)

    if abs(int(time.time() * 1000) - ts) > SKEW_MS:

        abort(401)

    raw = request.get_data()                                # bytes, bukan parsed JSON

    expected = "sha256=" + hmac.new(

        SECRET, f'{ts}'.encode() + raw, hashlib.sha256

    ).hexdigest()

    if not hmac.compare_digest(sig_hdr, expected):
```

```
    abort(401)

    payload = json.loads(raw)

    # ... process

    return "ok", 200
```

## 6.3 Verify inbound body-only (action endpoint)

```
@app.post("/aspri/action")
```

```
def handle_action():

    sig_hdr = request.headers.get("X-ASPRI-Signature", "")

    raw = request.get_data()

    expected = "sha256=" + hmac.new(

        ACTION_SECRET, raw, hashlib.sha256

    ).hexdigest()

    if not hmac.compare_digest(sig_hdr, expected):

        abort(401)

    # ... process

    return "ok", 200
```

---

## 7. Kode contoh — curl + openssl (reproduksi manual)

Untuk debugging — generate signature secara manual lalu compare dengan output kode Anda.

### 7.1 Ts-prefixed (callback ke EMBAN)

```
#!/usr/bin/env bash
```

```
SECRET="your-action-callback-secret"
```

```
BASE="https://api.rentalai.id"
```

```
SUBMISSION_ID="01J1ZXK7ABCDE..."
```

```
BODY_FILE="/tmp/body.json"
```

```
cat > "$BODY_FILE" <<'JSON'
```

```
{"status": "success", "template_key": "booking_confirmed", "data": {"booking_id": "BK-2026-0412"}}
```

```
JSON
```

```
TS=$(date +%s%3N)
```

```
BODY=$(cat "$BODY_FILE")
```

```
SIG="sha256=$(printf '%s.%s' "$TS" "$BODY" | openssl dgst -sha256 -hmac "$SECRET" -hex |  
awk '{print $2}')
```

```
curl -sS -X POST "$BASE/api/v1/callbacks/$SUBMISSION_ID" \
```

```
-H "Content-Type: application/json" \
```

```
-H "X-ASPRI-Timestamp: $TS" \
```

```
-H "X-ASPRI-Signature: $SIG" \
```

```
-H "Idempotency-Key: $SUBMISSION_ID" \
```

```
--data-binary "@$BODY_FILE"
```

**Pitfall:** `-d` vs `--data-binary`. `-d` akan strip newline/whitespace; gunakan `--data-binary` supaya body di-kirim byte-for-byte sama dengan yang di-hash.

## 7.2 Body-only (verify inbound dari EMBAN — manual reproduksi)

Skip TS, just hash body:

```
SIG_EXPECTED="sha256=$(cat "$BODY_FILE" | openssl dgst -sha256 -hmac "$SECRET"  
-hex | awk '{print $2}')
```

```
echo "Expected: $SIG_EXPECTED"
```

```
echo "Got: $(grep X-ASPRI-Signature received-headers.txt)"
```

---

## 8. Daftar error autentikasi

HTTP	error_code	Artinya	Aksi
401	missing_auth_headers	salah satu dari X-ASPRI-Signature / X-ASPRI-Timestamp absen (timestamp untuk ts-prefixed; signature selalu)	pastikan header sesuai skema endpoint
401	invalid_timestamp	X-ASPRI-Timestamp bukan angka	kirim Unix millis sebagai string
401	timestamp_skew	drift > 5 menit	sync NTP; cek response body untuk nilai drift
401	unknown_signer	EMBAN tidak tahu secret untuk request ini	biasanya tenant_id salah, atau tenant belum set ticket_webhook_secret_env di sisi mereka
401	invalid_signature	signature mismatch	cek: (a) body bytes identik dengan yang di-sign, (b) secret benar, (c) hex lowercase, (d) prefix sha256= ada, (e) skema benar (ts-prefixed vs body-only)

---

## 9. Tips debugging signature mismatch

1. **Print-then-send**: log `raw_request_body` yang di-sign sebelum dikirim, log lagi setelah dikirim. Harus identik.
2. **Encoding**: raw body harus UTF-8. Non-ASCII (aksara Arab, emoji) aman selama encoding konsisten di kedua sisi.
3. **Headers dipakai apa?** EMBAN hanya men-sign `timestamp.body` (atau `body` saja untuk body-only) — **tidak** mensign header lain. Jadi perbedaan header tidak mempengaruhi signature.
4. **Window**: jika `timestamp_skew`, response menyertakan nilai drift ms; pakai itu untuk mengukur seberapa jauh clock Anda mundur/maju.
5. **Salah skema**: paling sering. Action Engine outbound = body-only. Semua yang lain = ts-prefixed. Cek ulang di doc endpoint mana.
6. **Test vektor**: ulang dengan body `{"ping":1}` + timestamp tetap + secret tetap, bandingkan digest dengan implementasi lain (mis. CyberChef / openssl) untuk memastikan library HMAC Anda benar.

Contoh test vektor (ts-prefixed):

```
secret = "test-secret"
```

```
timestamp = 1745145600123
```

```
body = {"ping":1}
```

```
input = 1745145600123>{"ping":1}
```

```
sha256 = 6a2f8e0b0f0e3b2f7e0b7a4e8e4f0c9e2d1b8a5f3c6e9d2a4b7e0c1d3f5a8b9c  
(contoh)
```

```
header = X-ASPRI-Signature: sha256=6a2f8e0b...8b9c
```

Angka di atas **bukan** nilai nyata — regenerate dengan HMAC sendiri untuk verifikasi.

---

## 10. Rotasi secret

**Ticket / action secret (per-tenant):**

- Koordinasi per tenant. Secret diubah di file env EMBAN + sisi Anda dalam satu operasi.

- Pakai nama env var baru (`TICKET_WEBHOOK_SECRET_TENANT_X_V2`) dan update `tenants.ticket_webhook_secret_env` via migration/admin tool — ini menjamin tidak ada "ambiguous" secret.

#### Indikator leak:

- 401 `invalid_signature` burst dari IP yang tidak dikenal → audit log, anggap secret compromised, rotate segera.
- Secret jangan sekali-kali di-commit ke git. Kalau terlanjur, rotate dulu baru rewrite history.

**Cadence yang biasa dipakai:** 90 hari untuk webhook secrets, lebih sering kalau Anda punya regulatory requirement.

---

## 11. Ringkasan checklist

- ☐ Punya 1 ticket secret per tenant CS (kalau menerima forwarded ticket).
- ☐ Punya 1 action secret per custom action (kalau host action endpoint).
- ☐ NTP-sync di semua host yang sign/verify (untuk ts-prefixed scheme).
- ☐ Body byte-for-byte identik dengan yang di-sign (di kedua arah: verify inbound + sign outbound).
- ☐ Header `X-ASPRI-Signature` pakai prefix `sha256=` + lowercase hex.
- ☐ Pilih skema yang benar per endpoint (ts-prefixed vs body-only) — lihat doc endpoint masing-masing.
- ☐ Library HMAC-SHA256 standar (crypto/hashlib/openssl) — jangan re-implement.

Lanjut: **#5 Action engine** (lalu #3 Callbacks / #6 File delivery / #4 Ticketing).