

Webhooks i Relatel

Relatel · august 2026

Med webhooks får dine egne systemer besked med det samme, når der sker noget i Relatel. Et opkald starter, en besked kommer ind, en kontakt bliver rettet. I stedet for at spørge os med faste mellemrum sender vi til dig, så snart hændelsen sker.

Du opretter og styrer dem selv under **Firmaadministration** → **Webhooks**. Der skal ikke skrives en udvikleraftale, og der skal ikke sættes noget op hos Relatel.

Indhold

1. Kort fortalt
2. Oversigten
3. Opret et webhook
4. Signeringsnøglen
5. Sådan tjekker du, at det virker
6. Følg dine leveringer
7. Når et webhook stopper af sig selv
8. Rotér nøglen
9. Pause
10. Nuværende begrænsninger

Kort fortalt

1. Du har en HTTPS-adresse, der kan modtage `POST` med JSON.

2. Du opretter et webhook, vælger hvilken ressource det skal lytte på, og hvilke hændelser du vil have.
 3. Du gemmer signeringsnøglen. Den vises **kun én gang**.
 4. Du sender en testlevering og tjekker, at dit system svarer `2xx`.
 5. Derefter kan du følge hver enkelt levering, svartid og status i listen.
-

Oversigten

Skærbillederne i denne guide er fra en opdigtet eksempel klinik, Tandklinikken Eksempel. De har oprettet ét webhook per system, der skal have besked: **Booking**, **Journalssystem**, **Patient-SMS**, **Patientkartotek**, **Regnskab**, **Vagttelefon** og **Ledelsesrapport**.

Webhooks

Send hændelser fra Relatel til dine egne systemer i realtid.

NAVN	OMRÅDE	HÆNDELSER	STATUS
Booking https://hooks.tandklinik.example/relatel/1	Hovednummer	1	Aktiv
Journalsystem https://hooks.tandklinik.example/relatel/2	Hovednummer	1	Aktiv
Ledelsesrapport https://hooks.tandklinik.example/relatel/4	Hovednummer	1	Aktiv
Patient-SMS https://hooks.tandklinik.example/relatel/7	Hovednummer	2	Aktiv
Patientkartotek https://hooks.tandklinik.example/relatel/5	Hovednummer	2	Aktiv
Regnskab https://hooks.tandklinik.example/relatel/6	Hovednummer	2	På pause
Vagttelefon https://hooks.tandklinik.example/relatel/3	Hovednummer	1	Stoppet

Hver linje er ét webhook: navn, den adresse vi sender til, hvilken ressource det lytter på, status og hvornår vi sidst leverede.

I eksemplet er **Regnskab** sat på pause af klinikken selv, og **Vagttelefon** er stoppet automatisk, fordi dens endpoint blev ved med at fejle. De øvrige leverer som de skal.

Status betyder:

Status	Betydning
Aktiv	Vi sender hændelser.
På pause	Du har selv sat det på pause. Vi sender ikke, og vi gemmer ikke hændelserne.
Stoppet	Vi har stoppet det automatisk, fordi dit endpoint blev ved med at fejle.

Opret et webhook

Grundlæggende

Navn

Kort navn så du kan skelne dine webhooks.

Beskrivelse

Endpoint-URL

Skal begynde med https://. Vi sender hændelser til denne adresse.

Navn er kun til dig, så du kan skelne dine webhooks fra hinanden.

Endpoint-URL skal begynde med `https://`. Vi sender ikke til `http://`, og heller ikke til adresser på private netværk. Det tjekker vi hver gang vi sender, ikke kun når du gemmer.

Ressource er det vigtigste valg. Et webhook lytter på præcis én ressource, og du får kun hændelser fra den:

Ressource	Hændelser
Et hovednummer	<code>call.created</code> , <code>call.ended</code> , <code>incoming_message.created</code>
En chat	<code>chat.new</code> , <code>chat.answered</code> , <code>chat.ended</code>
Kontakter (hele firmaet)	<code>contact.created</code> , <code>contact.updated</code> , <code>contact.destroyed</code>

Vil du have opkald fra to hovednumre, opretter du to webhooks. Der findes ikke et “alle hovednumre”-valg. Det er bevidst, så et webhook aldrig kommer til at sende mere, end du har valgt.

Listen med hændelser skifter, når du vælger ressource, så du kun ser de hændelser, der findes for netop den.

Ressource & hændelser

Ressource

Hovednummer ▾

Webhooken modtager kun hændelser for den ressource, du vælger.

Hændelser

Hovednummer

☐ `call.created`

☐ `call.ended`

☐ `incoming_message.created`

AVANCERET

Avanceret

Timeout (sekunder)

Maks. antal forsøg

Ekstra headers

```
{"Authorization": "Bearer abc123"}
```

Timeout er hvor længe vi venter på dit svar. Standard og højeste værdi er 30 sekunder. Svarer du ikke inden da, tæller det som en fejl.

Maks. antal forsøg er hvor mange gange vi prøver, før vi giver op. Standard og højeste værdi er 8.

Ekstra headers kan du bruge, hvis din modtager kræver f.eks. en `Authorization`-header. Skriv dem som JSON:

```
{"Authorization": "Bearer abc123"}
```

De bliver krypteret hos os, og de bliver ikke skrevet i vores logfiler.

Signeringsnøglen

Herunder er **Patientkartotek** netop oprettet, så nøglen vises for første og eneste gang.

Patientkartotek

https://hooks.tandklinik.example/relatel/5 Aktiv

Succesrate (24t) —	Leveringer (24t) 0	Typisk svartid —	Langsomste 5% —	Fejl i træk 0
-----------------------	-----------------------	---------------------	--------------------	------------------

Send testlevering Pause Rediger Slet

Seneste leveringer

Opdatér

Ingen leveringer endnu. Når hændelser sker på dette område, dukker de op her.

Signeringsnøgle

Kopiér din signeringsnøgle

Det er eneste gang, du ser den. Gem den hos din webhook-modtager, inden du forlader siden.

59vu4JCdJb6NUPNUKytKKD7oP6ouqZeHivRrWdFXEJFx2maJtRp8wsyWHbeN3tTH

Kopiér

Mister du den, skal du rotere nøglen for at få en ny.

Signeringsnøgle 22c4f217

Rotér nøgle

Når du opretter et webhook, får du en signeringsnøgle. **Den vises kun denne ene gang.** Kopiér den ind i dit modtagersystem, inden du forlader siden.

Bagefter viser vi kun et *fingeraftryk* af nøglen, altså de første otte tegn af en SHA-256 af den, f.eks. `22c4f217`. Det er nok til at se, hvilken nøgle der er i brug, men fingeraftrykket kan ikke bruges til at regne nøglen ud. Derfor kan det roligt stå i en mail eller en supportsag.

Har du mistet nøglen, kan du ikke få den igen. Så roterer du i stedet og får en ny.

Sådan tjekker du, at det virker

Hver levering fra os har disse headers:

```
Content-Type: application/json
Relatel-Delivery-Id: 8f2e1c4a-6b03-4a7d-9e11-5c2f0a3b8d47
Relatel-Event-Type: call.created
Relatel-Timestamp: 1785062400
Relatel-Signature: t=1785062400,v1=3b8f2c91d4...
```

`Relatel-Timestamp` er Unix-tid i hele sekunder, altså tallet `1785062400`, ikke en dato på ISO-form.

`Relatel-Delivery-Id` er den samme på alle forsøg af samme hændelse. Brug den til at genkende noget, du allerede har behandlet.

Bodyen ser sådan ud:

```
{
  "delivery_id": "8f2e1c4a-6b03-4a7d-9e11-5c2f0a3b8d47",
  "event_type": "call.created",
  "occurred_at": "2026-08-07T09:20:00Z",
  "payload": { }
}
```

VERIFICÉR SIGNATUREN

`v1` er en HMAC-SHA256 over en streng, du selv sætter sammen: tidsstemplet, et punktum, og den rå body præcis som du modtog den. Med tallene ovenfor er strengen, der signeres, altså:

```
1785062400>{"delivery_id":"8f2e1c4a-6b03-4a7d-9e11-5c2f0a3b8d47","event_type":"call.created", ...}
```

Fremgangsmåden er den samme i alle sprog: læs den rå body, byg strengen, lav en HMAC-SHA256 med din signeringsnøgle, og sammenlign med `v1`. Herunder er den vist i Ruby og i JavaScript.

I **Ruby**, hvor `openssl` ligger i standardbiblioteket. I en Rails-controller er den rå body `request.raw_post`, og headeren er `request.headers['Relatel-Signature']`:

```
require 'openssl'

def verificer(raw_body, header, signeringsnoegle)
  timestamp = header[/t=(\d+)/, 1]
  return false if timestamp.nil?

  # Afvis for gamle leveringer. Vi anbefaler højst 5 minutter.
  return false if (Time.now.to_i - timestamp.to_i).abs > 300

  forventet = OpenSSL::HMAC.hexdigest(
    'sha256', signeringsnoegle, "#{timestamp}.#{raw_body}"
  )

  signaturer = header.split(',')
    .select { |del| del.start_with?('v1=') }
    .map { |del| del.delete_prefix('v1=') }

  # Der kan være flere v1=. Accepter, hvis én af dem passer.
  signaturer.any? do |kandidat|
    kandidat.bytesize == forventet.bytesize &&
      OpenSSL.fixed_length_secure_compare(kandidat, forventet)
  end
end
```

Det samme i **JavaScript til Node.js**, hvor `crypto` er indbygget:

```
// Node.js. rawBody er request-bodyen som tekst, ikke som parset objekt.
// header er værdien af Relatel-Signature, f.eks.
// "t=1785062400,v1=3b8f2c91d4...,v1=91acdd0e77..."
const crypto = require('crypto')

function verificer(rawBody, header, signeringsnoegle) {
  const timestamp = header.match(/t=(\d+)/)[1]

  // Afvis for gamle leveringer. Vi anbefaler højst 5 minutter.
  if (Math.abs(Date.now() / 1000 - Number(timestamp)) > 300) {
    return false
  }

  const forventet = crypto
    .createHmac('sha256', signeringsnoegle)
    .update(timestamp + '.' + rawBody)
    .digest('hex')

  const forventetBuffer = Buffer.from(forventet, 'hex')

  // Der kan være flere v1=. Accepter, hvis én af dem passer.
  return header
    .split(',')
    .filter((del) => del.startsWith('v1='))
    .some((del) => {
      const modtaget = Buffer.from(del.slice(3), 'hex')
      // timingSafeEqual kaster, hvis de to buffere ikke er lige lange.
      return (
        modtaget.length === forventetBuffer.length &&
        crypto.timingSafeEqual(modtaget, forventetBuffer)
      )
    })
}
```

Tre ting det er værd at gøre rigtigt:

- **Brug den rå body.** Parser du JSON'en først og laver den til tekst igen, ændrer tegnene sig, og signaturen passer ikke længere.
- **Sammenlign i konstant tid.** Brug `OpenSSL.fixed_length_secure_compare` eller `timingSafeEqual`, ikke `==` eller `===`.
- **Accepter flere v1=.** Under en nøglerotation sender vi én signatur per gyldig nøgle. Passer bare én af dem, er leveringen ægte.

SEND EN TESTLEVERING

Knappen **Send testlevering** sender en `webhook.test`-hændelse gennem præcis samme kø, signering og genforsøg som rigtige hændelser. Kan din modtager verificere en testlevering, kan den også verificere en rigtig. Testleveringen indeholder ingen kundedata.

Følg dine leveringer

Listen herunder er fra **Patient-SMS**, som har kørt et døgn. Øverst ligger et forsøg, der lige er fejlet, og en hændelse vi har opgivet. Resten er leveret.

TID	HÆNDELSE	SVAR	STATUS	SVARTID	FORSØG	
31 Aug 14:07	call.created	502	Prøver igen om 13 minutter	—	3	Send igen
31 Aug 13:12	call.ended	500	Opgivet	—	8	Send igen
31 Aug 12:32	call.created	200	Leveret	164ms	1	
31 Aug 11:52	call.ended	200	Leveret	201ms	1	
31 Aug 11:12	call.created	200	Leveret	238ms	1	
31 Aug 10:32	call.ended	200	Leveret	275ms	1	
31 Aug 09:52	call.created	200	Leveret	312ms	1	
31 Aug 09:12	call.ended	200	Leveret	349ms	1	

Øverst på siden står de vigtigste tal for de sidste 24 timer: succesrate, antal leveringer, typisk svartid, de langsomste 5 % og hvor mange fejl der er i træk lige nu.

I listen betyder status:

Status	Betydning
Leveret	Du svarede <code>2xx</code> .
Prøver igen om ...	Forsøget fejlede, og vi prøver igen på det viste tidspunkt.
Opgivet	Vi har brugt alle forsøg. Vi prøver ikke mere af os selv.

Læg mærke til, at **Prøver igen** og **Opgivet** godt kan vise samme svarkode. Det er statussen, ikke koden, der siger, om der kommer flere forsøg.

En levering, der er opgivet eller mislykket, kan sendes igen med **Send igen**.

Vi gemmer leveringer i **30 dage**, og listen viser de seneste 50.

HVORNÅR PRØVER VI IGEN?

Vi venter længere for hvert forsøg: 1 minut, 5 minutter, 25 minutter, cirka 2 timer, cirka 10 timer, og derefter højst 24 timer mellem forsøg. Med standardindstillingen på 8 forsøg har du omkring **3½ døgn** til at få dit endpoint på benene igen, før vi opgiver en hændelse.

Når et webhook stopper af sig selv

Her er det **Vagttelefon**, vi har stoppet, og beskeden øverst siger hvorfor.

Vagttelefon

https://hooks.tandklinik.example/relatel/3

Stoppet

Stoppet automatisk, fordi 10 leveringer blev opgivet inden for en time. Ret dit endpoint, og vælg Genoptag.

Succesrate (24t)

0.0%

Leveringer (24t)

10

Typisk svartid

155ms

Langsomste 5%

—

Fejl i træk

5

Send testlevering

Genoptag

Rediger

Slet

Seneste leveringer

Opdatér

TID	HÆNDELSE	SVAR	STATUS	SVARTID	FORSØG	
31 Aug 11:56	webhook.test	500	Opgivet	120ms	1	Send igen
31 Aug 11:53	webhook.test	500	Opgivet	149ms	1	Send igen
31 Aug 11:50	webhook.test	500	Opgivet	178ms	1	Send igen
31 Aug 11:47	webhook.test	500	Opgivet	207ms	1	Send igen
31 Aug 11:44	webhook.test	500	Opgivet	126ms	1	Send igen
31 Aug 11:41	webhook.test	500	Opgivet	155ms	1	Send igen
31 Aug 11:38	webhook.test	500	Opgivet	184ms	1	Send igen
31 Aug 11:35	webhook.test	500	Opgivet	213ms	1	Send igen

Bliver dit endpoint ved med at fejle, stopper vi webhooket. Det er for at undgå, at vi bliver ved med at sende til noget, der er nede, og for at du kan se, at der er noget i vejen.

Sådan starter du igen:

1. Ret årsagen i dit eget system.
2. Tryk **Send testlevering**, og se at den bliver **Leveret**.
3. Tryk **Genoptag**.

Er dit endpoint stadig i stykker, når du genoptager, stopper vi det igen. Vi nulstiller nemlig ikke fejltællingen, når et webhook har stoppet sig selv. Sådan kan du ikke klikke dig ud af problemet uden at have løst det.

Sætter du selv et webhook på pause, er det en anden sag. Der nulstiller vi tællingen, når du starter igen.

Rotér nøglen

Booking er herunder midt i en rotation: begge nøgler virker, og siden viser hvornår den gamle udløber.

Booking

https://hooks.tandklinik.example/relatel/1 Aktiv

Succesrate (24t)
100.0%

Leveringer (24t)
1

Typisk svartid
197ms

Langsomste 5%
—

Fejl i træk
0

Send testlevering
Pause
Rediger
Slet

Seneste leveringer

Opdater

TID	HÆNDELSE	SVAR	STATUS	SVARTID	FORSØG
31 Aug 13:32	webhook.test	200	Leveret	197ms	1

Signeringsnøgle

Begge nøgler bliver accepteret indtil 1. sep 2026, 13:57, cirka 24 timer fra nu. Skift din modtager til den nye nøgle inden da.

Ny nøgle

Forrige nøgle

a2893d83

8a9f8988

Rotér igen
Annullér rotation

Skal du skifte signeringsnøgle, fordi den er kommet et forkert sted hen, eller fordi I roterer nøgler fast, trykker du **Rotér nøgle**.

Så sker der dette:

- Du får en ny nøgle vist én gang. Gem den.
- **I 24 timer accepterer vi begge nøgler.** Vi sender én `v1=`-signatur per nøgle, så du kan skifte over i dit eget tempo uden at miste leveringer.
- Siden viser fingeraftryk på begge nøgler, og hvornår den gamle udløber.

Har du trykket ved en fejl, kan du bruge **Annullér rotation** inden for de 24 timer. Så bliver den gamle nøgle den gældende igen, og den nye kasseres.

Efter de 24 timer virker kun den nye nøgle.

Pause

Pause stopper alle leveringer med det samme, også de forsøg der allerede var i kø. Brug den, når du skal have din modtager ned til vedligeholdelse.

Vi gemmer **ikke** hændelser under en pause. De hændelser, der sker i pausen, bliver ikke leveret bagefter. Skal du kunne hente det, du gik glip af, skal dit system hente det via vores API.

Nuværende begrænsninger

- Kun de hændelser, der står i tabellen ovenfor. Flere kommer til.
 - Én ressource per webhook. Der findes ikke et “alle hovednumre”-valg.
 - Ingen filtrering på, om et opkald er ind- eller udgående.
 - Timeout højst 30 sekunder, højst 8 forsøg.
 - Leveringer gemmes i 30 dage.
 - Vi garanterer ikke rækkefølgen. Brug `occurred_at` i bodyen, hvis rækkefølge betyder noget for dig.
 - Samme hændelse kan i sjældne tilfælde leveres mere end én gang. Brug `Relatel-Delivery-Id` til at genkende en levering, du allerede har behandlet.
-

Spørgsmål

Skriv til din kontakt hos Relatel. Har du et konkret problem med en levering, hjælper det meget, hvis du sender `Relatel-Delivery-Id` og tidspunktet med. Så kan vi slå præcis den levering op.