OPENCLAW DI MACOS

Memasang, mengamankan, dan menjalankan agen yang melayani pelanggan sungguhan dari Mac Anda

GATEWAY · IMESSAGE · WHATSAPP · SKILLS · PLUGIN · USE CASE

DISUSUN OLEH ROMI NUR ISMANTO
HELLO@ROMINUR.COM

v2026.9.4
WWW.ROMINUR.COM
DITINJAU TERHADAP DOKUMENTASI RESMI, 17 SEPTEMBER 2026

OPENCLAW

DI MACOS

PANDUAN LENGKAP v2026.9.4

Instalasi • Gateway • Kanal • Skills • Plugin • Keamanan • Use Case Bisnis

YANG DIJANJIKAN BUKU INI

— 66 bab, sepuluh bagian, disusun untuk dibaca berurutan
— Setiap perintah dan kunci konfigurasi dirujuk ke dokumentasi resmi
— Enam use case bisnis lengkap: CS, aplikasi bisnis, riset saham, dan lainnya
— Ditulis khusus untuk Mac — launchd, TCC, iMessage, dan node Mac

Disusun oleh Romi Nur Ismanto
hello@rominur.com | www.rominur.com
Edisi Bahasa Indonesia | 17 September 2026

PENGANTAR

Buku untuk satu mesin yang sangat spesifik

OpenClaw berjalan di banyak sistem operasi. Buku ini sengaja hanya membahas satu: macOS. Alasannya bukan selera. Mac adalah satu-satunya tempat sebagian kemampuan OpenClaw benar-benar hidup — iMessage hanya bisa dijembatani dari Mac, izin TCC punya perilaku yang tidak ada padanannya di Linux, layanan latar dikelola launchd, dan aplikasi menu bar membawa kapabilitas perangkat yang tidak tersedia lewat CLI.

Buku yang mencoba melayani semua sistem operasi sekaligus akan menghabiskan separuh halamannya untuk menulis “kecuali di macOS”. Buku ini memilih arah sebaliknya: satu platform, dibahas sampai ke sudutnya.

Siapa yang akan tertolong

- Insinyur IT yang diminta menjalankan asisten AI di mesin perusahaan, bukan di layanan pihak ketiga
- Pemilik bisnis yang ingin agen menjawab pelanggan di WhatsApp atau iMessage tanpa menyerahkan data ke luar
- Operator yang sudah memasang OpenClaw, lalu menemukan bahwa bagian tersulit justru datang setelah instalasi
- Siapa pun yang perlu menghitung apakah sebuah use case layak dijalankan sebelum menulis satu baris konfigurasi

Cara buku ini menjaga dirinya tetap benar

Setiap perintah, nama berkas, dan kunci konfigurasi di buku ini berasal dari dokumentasi resmi OpenClaw yang ditinjau pada tanggal terbit. Tidak ada perintah yang ditulis dari ingatan. Ketika sebuah angka berasal dari pemodelan bisnis dan bukan dari dokumentasi, angka itu ditandai sebagai sintetis di tempatnya.

> **ATURAN PERTAMA**
> OpenClaw bergerak cepat. Kanal rilis stable saja sudah berganti beberapa kali dalam satu bulan. Perlakukan setiap perintah di buku ini sebagai titik awal yang harus Anda cocokkan dengan openclaw --version dan halaman dokumentasi yang relevan sebelum dijalankan pada mesin produksi.

PENGANTAR / STATUS

Apa yang sudah terverifikasi dan apa yang belum

Kejujuran soal sumber lebih berguna daripada kesan lengkap. Tabel berikut menyatakan dari mana isi tiap bagian berasal.

Bagian	Sumber	Status
I dan II — Mac	docs.openclaw.ai/platforms/macos dan halaman turunannya	Ditinjau 17 September 2026
III–VIII	Indeks dokumentasi resmi dan halaman rujukan per topik	Kerangka terverifikasi; detail diperluas bertahap
IX — Use case	Pemodelan bisnis penulis	Angka sintetis; arsitektur mengikuti kemampuan nyata
X — Biaya	Model perhitungan yang dapat Anda jalankan sendiri	Tarif harus diganti dengan tarif penyedia Anda

Versi yang dipakai sebagai acuan sepanjang buku adalah v2026.9.4, rilis stabil terbaru pada saat penyusunan. Bab 2 menjelaskan mengapa OpenClaw tidak pernah punya nomor versi seperti “2.2” dan bagaimana membaca penomorannya.

PENGANTAR / NAVIGASI

Peta enam puluh enam bab

Sepuluh bagian, disusun dari mesin kosong sampai sistem yang melayani pelanggan nyata. Bagian I sampai IV membangun fondasi; V sampai VIII memperluas kemampuan dan mengunci keamanannya; IX dan X menjawab pertanyaan yang selalu datang terakhir tetapi menentukan segalanya: apakah ini layak dijalankan.

Bagian I — Fondasi dan instalasi di Mac

Bab	Judul	Hal.
1	Apa itu OpenClaw dan apa yang sebenarnya Anda pasang	9
2	Penomoran versi, kanal rilis, dan arti v2026.9.4	17
3	Prasyarat Mac dan dua keputusan arsitektur awal	24
4	Memasang aplikasi macOS dan menjalankan first run	31
5	Jendela Connection, profil Gateway, dan mode sambungan	38
6	Pembaruan terkoordinasi aplikasi dan Gateway	45
7	Workspace agen: berkas yang menjadi ingatan kerja	52
8	Runtime agen tertanam, sesi, dan penyimpanan SQLite	59

Bagian II — Operasi harian di macOS

Bab	Judul	Hal.
9	Multi-agen dan pengikatan rute kanal	66
10	Quick Chat, WebChat, dan Mac tab	73
11	Impor login browser dan sinkronisasi cookie	79
12	Node Mac: kamera, layar, lokasi, dan kontrol komputer	86
13	Sikap aman sejak hari pertama	94
14	Membaca dokumentasi resmi dan memverifikasi sendiri	101

Bagian III — Gateway dan konfigurasi

Bab	Judul	Hal.
15	Arsitektur Gateway dan protokol WebSocket	108
16	Berkas openclaw.json dan hot reload	115
17	Autentikasi Gateway, token, dan peran operator	122
18	Model provider, kunci API, dan failover	128

Bab	Judul	Hal.
19	Logging, diagnostics, dan doctor	134
20	Remote access: SSH, Tailscale, dan Cloudflare	140
21	Backup, restore, dan pemindahan mesin	147

Bagian IV — Kanal percakapan

Bab	Judul	Hal.
22	Peta kanal dan kemampuan per platform	154
23	iMessage lewat imsg: pemasangan di Mac	160
24	iMessage: private API, tapback, dan polling	168
25	iMessage: kontrol akses, grup, dan pola penempatan	175
26	WhatsApp: akses, pengiriman, dan operasi	181
27	WhatsApp: grup, allowlist, dan sesi	188
28	Telegram: pemasangan dan transport	195
29	Telegram: topik forum, sesi, dan pesan kaya	200
30	Slack, Discord, Signal, dan kanal kerja lain	207

Bagian V — Skills dan ClawHub

Bab	Judul	Hal.
31	Anatomi skill dan format folder	214
32	Skills CLI: cari, pasang, verifikasi, perbarui	220
33	ClawHub: penemuan, pemindaian, dan audit keamanan	225
34	Menerbitkan skill dan mengelola namespace	231
35	Workshop, library, dan skill yang dipelajari agen	237
36	Konfigurasi visibilitas skill per agen	243

Bagian VI — Plugin dan ekstensi

Bab	Judul	Hal.
37	Model kapabilitas dan siklus muat plugin	249
38	Manifest plugin dan validasi skema ketat	256
39	Membangun plugin pertama	263
40	Hook: agen, tool, pesan, dan siklus hidup	268

Bab	Judul	Hal.
41	Plugin bawaan yang layak dinyalakan	274
42	MCP: server, transport, dan OAuth	280

Bagian VII — Tools, nodes, dan otomasi

Bab	Judul	Hal.
43	Kebijakan tool, profil, dan mode kode	285
44	Exec, proses latar, dan persetujuan perintah	291
45	Nodes: pairing, kapabilitas, dan izin	298
46	Automations: cron, webhook, dan pemicu Gmail	303
47	Heartbeat, standing order, dan Task Flow	310
48	Memori: builtin, pencarian, dan dreaming	316

Bagian VIII — Keamanan, sandboxing, dan produksi

Bab	Judul	Hal.
49	Model kepercayaan dan batas yang didukung	322
50	Kontrol akses, allowlist, dan otorisasi perintah	328
51	Prompt injection dan lapisan yang tetap Anda tegakkan	333
52	Sandboxing: mode, cakupan, dan backend	339
53	Secrets, penyimpanan, dan log	346
54	Audit keamanan dan runbook insiden	352

Bagian IX — Use case bisnis nyata

Bab	Judul	Hal.
55	Kerangka menilai kelayakan sebuah use case	358
56	Use case A — Customer service multi-kanal	363
57	Use case B — Asisten operasi aplikasi bisnis	370
58	Use case C — Riset dan pemantauan pasar saham	377
59	Use case D — Back-office dokumen dan penagihan	384
60	Use case E — Asisten penjualan dan CRM	390
61	Use case F — Operasi internal tim IT	397
62	Pola yang berulang di keenam use case	402

Bagian X — Biaya, skala, dan tata kelola

Bab	Judul	Hal.
63	Model biaya token dan infrastruktur	407
64	Kapasitas, batas, dan titik jenuh	414
65	Checklist go-live dan matriks risiko	420
66	Tata kelola jangka panjang dan serah terima	425

Nomor halaman terisi untuk bab yang sudah ditulis pada edisi ini. Tanda hubung menandai bab yang sedang disusun.

PENGANTAR / JALUR BACA

Tiga jalur, tergantung yang Anda kejar

Gambar 0.1 — Empat jalur baca. Ketiganya bertemu di Bagian VIII; tidak ada jalur yang boleh melewatinya.

Satu peringatan sebelum Anda memilih jalur pintas. OpenClaw adalah perangkat lunak yang sengaja diberi kemampuan menjalankan perintah di mesin Anda, membaca dan menulis berkas, serta mengirim pesan keluar atas nama Anda. Bagian VIII bukan pelengkap. Ia adalah syarat sebelum agen Anda menyentuh data atau orang sungguhan.

FONDASI DAN INSTALASI

Bab 1 Apa itu OpenClaw dan apa yang sebenarnya Anda pasang

OpenClaw adalah gerbang swakelola yang menghubungkan aplikasi percakapan yang sudah Anda pakai setiap hari ke sebuah agen AI. Satu proses berjalan di mesin Anda sendiri, dan dari mana pun Anda mengirim pesan, agen itu yang menjawab. Kalimat itu terdengar sederhana, dan justru di situlah letak kesalahpahaman yang paling mahal.

Tujuan belajar

Membedakan OpenClaw dari layanan chatbot, menyebutkan komponen yang benar-benar terpasang di Mac Anda, dan menjelaskan mengapa keputusan keamanan harus diambil sebelum instalasi, bukan sesudahnya. Pada akhir bab, pembaca dapat menggambarkan jalur satu pesan dari aplikasi percakapan sampai kembali menjadi balasan.

Bukan chatbot, bukan layanan

Chatbot komersial hidup di server penyedianya. Anda mengirim pesan, pesan itu keluar dari jaringan Anda, dijawab di sana, lalu kembali. OpenClaw membalik posisinya: proses Gateway berjalan di perangkat keras Anda, data percakapan tersimpan di disk Anda, dan satu-satunya yang keluar adalah panggilan ke penyedia model — itu pun bisa diarahkan ke model lokal.

Konsekuensinya berpasangan. Anda mendapat kendali penuh atas data, dan Anda juga mewarisi seluruh tanggung jawab operasional: memperbarui, mengamankan, memantau, dan memulihkan. Tidak ada tim dukungan yang akan menyadari gerbang Anda mati pada pukul tiga pagi.

LAYANAN CHATBOT		OPENCLAW
Server penyedia	**Tempat data**	Disk mesin Anda
Yang disediakan	**Pilihan model**	Cloud atau lokal
Tidak ada	**Akses berkas**	Sesuai kebijakan tool
Tidak ada	**Jalankan perintah**	Ada, lewat persetujuan
Penyedia	**Tanggung jawab**	Anda
Langganan tetap	**Biaya**	Token yang dipakai

Gambar 1.1 — Dua model yang bertukar posisi. Kolom kanan memberi kendali, dan menyerahkan seluruh tanggung jawab operasional kepada Anda.

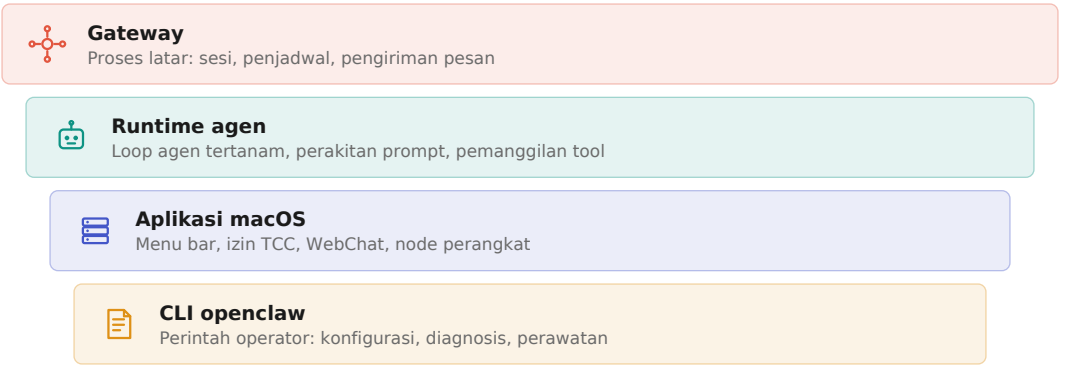
Baris keempat adalah yang paling sering diremehkan. Memberi agen kemampuan

menjalankan perintah di Mac Anda adalah keputusan yang tidak bisa ditarik setengah. Bab 13 membahas sikap awal yang benar, dan Bagian VIII membahasnya sampai tuntas.

BAB 1 / KOMPONEN

Empat benda yang terpasang di Mac Anda

Instalasi lengkap di macOS meninggalkan empat hal yang berbeda sifatnya. Membedakan keempatnya menghemat berjam-jam diagnosis, karena kegagalan pada masing-masing punya gejala yang mirip tetapi penanganan yang sama sekali berbeda.



Gambar 1.2 — Empat lapis yang terpasang. Gateway dan runtime agen adalah inti; aplikasi dan CLI adalah dua cara berbeda untuk menyentuhnya.

Gateway adalah satu-satunya yang wajib hidup terus. Aplikasi macOS boleh ditutup, CLI hanya dipanggil saat diperlukan, tetapi begitu Gateway berhenti, penjadwal ikut berhenti, kanal terputus, dan pekerjaan yang sedang berjalan tergantung pada pemulihan setelah restart.

Yang dimiliki aplikasi macOS

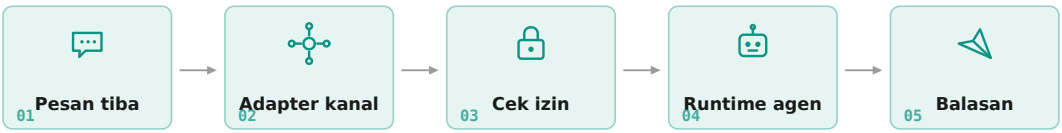
Gambar 1.4 — Lima hal yang dimiliki aplikasi macOS, dan satu kotak terakhir yang menegaskan apa yang bukan urusannya.

Yang tidak dimiliki aplikasi: konfigurasi Gateway, penyedia model, plugin, kanal, dan keamanan. Semua itu punya tempatnya sendiri, dan buku ini membahasnya di Bagian III sampai VIII.

BAB 1 / ALIRAN PESAN

Perjalanan satu pesan

Menelusuri satu pesan dari ujung ke ujung adalah cara tercepat memahami di mana sebuah kegagalan mungkin terjadi. Setiap anak panah pada gambar berikut adalah tempat yang pernah menjadi sumber masalah nyata bagi seseorang.

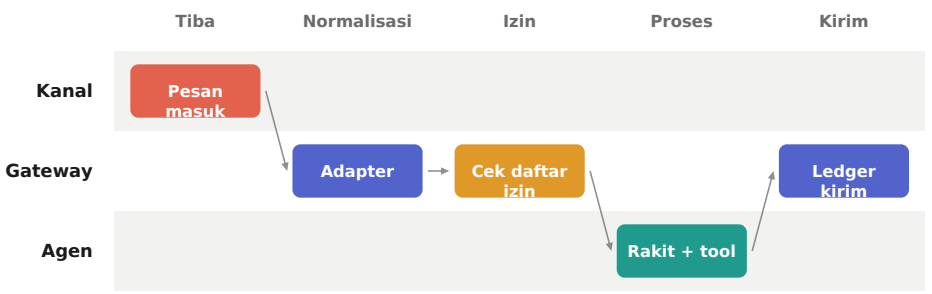


Gambar 1.3 — Lima tahap. Tahap ketiga adalah satu-satunya yang menolak, dan itulah alasan ia ada.

Pada tahap kedua, adapter kanal menormalkan bentuk pesan: WhatsApp, iMessage, dan Telegram punya format yang sangat berbeda, dan runtime agen tidak perlu mengetahui perbedaannya. Pada tahap ketiga, pengirim dicocokkan dengan daftar izin. Pengirim yang tidak dikenal tidak pernah mencapai model.

Tahap keempat adalah tempat biaya muncul. Di sanalah prompt dirakit, konteks dimuat, tool dipanggil, dan token dibakar. Sebagian besar upaya penghematan di buku ini menyasar tahap ini.

Di mana kegagalan biasanya jatuh



Gambar 1.5 — Pesan yang sama dilihat dari sisi pelaku. Agen hanya menyentuh satu kolom; sisanya pekerjaan gerbang.

Gejala	Tahap	Pemeriksaan pertama
Pesan tidak pernah sampai	1–2	Status kanal dan kredensialnya
Agen diam untuk orang tertentu	3	Daftar izin dan kebijakan DM
Balasan lambat sekali	4	Ukuran konteks dan rantai tool
Balasan dibuat tapi tidak terkirim	5	Ledger pengiriman dan pemutus arus
Agen menjawab hal yang salah	4	Berkas workspace dan prompt sistem

BAB 1 / PRAKTIKUM

Memetakan pemasangan Anda sendiri

Latihan ini tidak menuntut OpenClaw sudah terpasang. Tujuannya menuliskan keputusan sebelum Anda mengunduh apa pun, karena keputusan yang tidak tertulis akan berubah diam-diam saat instalasi berjalan.

Langkah 1 — tuliskan satu kalimat tujuan

Tulis dalam satu kalimat apa yang Anda ingin agen kerjakan, dengan kata kerja yang konkret. “Membantu pekerjaan” bukan kalimat tujuan. “Menjawab pertanyaan status pesanan dari reseller di WhatsApp” adalah kalimat tujuan, karena ia menyebut siapa, apa, dan di mana.

Langkah 2 — daftar kemampuan yang dibutuhkan kalimat itu

Pertanyaan	Jawaban Anda menentukan
Apakah agen perlu membaca berkas?	Kebijakan tool baca dan akses workspace
Apakah agen perlu menjalankan perintah?	Apakah exec dibuka sama sekali
Apakah agen perlu mengirim pesan keluar?	Kanal mana yang disambungkan
Apakah agen perlu mengingat antar hari?	Pengaturan sesi dan memori
Siapa yang boleh berbicara dengannya?	Kebijakan DM dan daftar izin

Sebagian besar pembaca menemukan bahwa kalimat tujuannya hanya memerlukan dua dari lima baris itu. Itu kabar baik: setiap baris yang tidak Anda butuhkan adalah permukaan risiko yang tidak perlu Anda buka.

Langkah 3 — tuliskan satu hal yang tidak boleh dilakukan agen

Ini langkah yang paling sering dilewati dan paling berguna. Tuliskan satu tindakan yang, bila dilakukan agen, akan Anda anggap sebagai insiden. Kalimat itu akan menjadi dasar kebijakan tool Anda di Bab 43, dan menuliskannya sekarang mencegah Anda merasionalisasinya nanti.

```
# Contoh isian
Tujuan   : Menjawab pertanyaan status pesanan reseller di WhatsApp
Perlu    : membaca data pesanan; mengirim pesan keluar
Tidak    : mengubah data pesanan, memberi diskon, menjanjikan pengiriman
Insiden  : agen menyebutkan tanggal kirim yang tidak ada di sistem
```

Empat baris ini cukup untuk memandu seluruh keputusan konfigurasi Anda.

BAB 1 / STUDI KASUS

Ketika agen mati dan tidak ada yang tahu

Sebuah distributor memasang agen pada MacBook seorang staf. Selama tiga minggu semuanya berjalan. Pada minggu keempat, staf itu cuti dan membawa laptopnya pulang dalam keadaan tertutup.

Agen berhenti menjawab. Tidak ada peringatan, karena peringatan itu sendiri dikirim oleh proses yang ikut mati. Reseller mengira pesan mereka terkirim dan menunggu. Baru pada hari ketiga seorang reseller menelepon dan bertanya mengapa tidak ada yang membalas.

Akar masalah	Pelajaran
Gateway berada di mesin yang tidur	Gateway menuntut host yang hidup terus; ini keputusan Bab 3
Tidak ada pemantauan eksternal	Sistem tidak dapat melaporkan kematiannya sendiri
Tidak ada pemilik bernama	Tiga hari berlalu tanpa seorang pun merasa bertanggung jawab
Pelanggan tidak tahu harus berharap apa	Tidak ada jalur cadangan yang diberitahukan

Yang menarik dari kasus ini adalah tidak ada satu pun kesalahan teknis. Konfigurasi benar, kanal benar, agen benar. Yang salah adalah penempatan dan kepemilikan — dua hal yang tidak muncul di layar mana pun.

> **PELAJARAN YANG BERLAKU DI SELURUH BUKU**
> Pemantauan yang berjalan di dalam sistem yang dipantau tidak dapat melaporkan kematian sistem itu. Untuk agen yang melayani orang lain, Anda memerlukan pemeriksaan dari luar — sekecil apa pun, bahkan sekadar seseorang yang mengirim pesan uji setiap pagi.

BAB 1 / LATIHAN

Latihan mandiri

1. Isi empat baris pada kotak kode di praktikum untuk kasus Anda sendiri. Simpan; Anda akan memakainya lagi di Bab 13, 43, dan 55.
2. Gambarkan perjalanan satu pesan pada kasus Anda memakai lima tahap Gambar 1.3. Tandai tahap mana yang paling mungkin gagal lebih dulu, dan tuliskan alasannya.
3. Untuk setiap dari empat komponen pada Gambar 1.2, tuliskan apa yang terjadi pada pengguna Anda bila komponen itu berhenti selama satu jam.
4. Rancang satu pemeriksaan dari luar yang akan memberi tahu Anda dalam lima belas menit bila agen berhenti menjawab. Ia tidak boleh berjalan di host yang sama.

BAB 1 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Menutup aplikasi Mac membuat agen mati	Gateway berjalan di dalam aplikasi, bukan sebagai layanan launchd. Lihat Bab 3 dan 5.
Agen menjawab di satu kanal, diam di kanal lain	Masalah kanal, bukan masalah agen. Periksa kredensial dan daftar izin kanal itu saja.
Biaya token naik tanpa penambahan pengguna	Konteks membengkak. Periksa berkas workspace dan riwayat sesi; lihat Bab 7 dan 8.
Perintah shell ditolak padahal sudah disetujui	Persetujuan, kebijakan tool, dan sandbox adalah tiga gerbang berbeda. Lihat Bab 52.
Tidak yakin versi mana yang terpasang	openclaw --version adalah sumber kebenaran, bukan ingatan atau artikel blog.

Tiga pertanyaan review

1. Sebutkan satu kemampuan OpenClaw yang tidak mungkin dimiliki chatbot berbasis layanan, dan satu tanggung jawab yang ikut datang bersamanya.
2. Dari empat komponen yang terpasang, mana yang boleh mati tanpa menghentikan agen, dan mana yang tidak? Jelaskan alasannya.
3. Sebuah pesan dari nomor asing tidak pernah dijawab. Pada tahap ke berapa ia berhenti, dan mengapa perilaku itu benar?

Kunci pembahasan

Kemampuan yang khas adalah menjalankan perintah dan membaca berkas di mesin Anda sendiri; tanggung jawab yang datang bersamanya adalah menegakkan batas kewenangan, karena tidak ada penyedia yang akan melakukannya untuk Anda. Aplikasi macOS dan CLI boleh mati; Gateway tidak, karena penjadwal dan adapter kanal hidup di dalamnya. Pesan dari nomor asing berhenti pada tahap tiga, dan itu benar karena gerbang tolak-dulu adalah satu-satunya perilaku aman untuk sistem yang bisa menjalankan perintah.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos dan gateway, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 2 Penomoran versi, kanal rilis, dan arti v2026.9.4

Banyak orang mencari “OpenClaw versi 2” atau “versi 2.2” dan tidak menemukan apa pun. Penyebabnya sederhana: OpenClaw tidak pernah memakai penomoran semantik. Ia memakai penomoran berbasis kalender, dan memahami polanya mengubah cara Anda merencanakan pembaruan.

Tujuan belajar

Membaca nomor versi OpenClaw, memilih kanal rilis yang sesuai dengan toleransi risiko Anda, dan mengenali rilis yang membawa perubahan pemutus. Pada akhir bab, pembaca dapat memutuskan kanal mana yang pantas untuk mesin produksinya.

Membaca nomornya

```
v2026.9.4
 │    │ │
 │    │ └── rilis keempat pada bulan itu
 │    └──── bulan (September)
 └─────────── tahun
```

Nomor versi memberi tahu kapan, bukan seberapa besar perubahannya.

Karena nomor tidak menyatakan besarnya perubahan, nomor juga tidak bisa dipakai untuk menebak apakah sebuah pembaruan aman. Rilis keempat dalam sebulan bisa saja membawa perubahan yang jauh lebih besar daripada rilis pertama. Satu-satunya cara mengetahuinya adalah membaca catatan rilis, dan buku ini menyarankan Anda menjadikannya kebiasaan sebelum setiap pembaruan.

> **JANGAN PANIK KARENA NOMOR**
> Perbedaan nomor rilis antara aplikasi Mac dan Gateway tidak dengan sendirinya menandakan masalah. Yang menandakan masalah adalah penolakan versi protokol saat handshake — dan aplikasi akan mengatakannya secara eksplisit, termasuk sisi mana yang perlu diperbarui.

BAB 2 / KANAL

Empat kanal dan siapa yang pantas memakainya

Kanal rilis menentukan seberapa cepat pembaruan sampai ke mesin Anda. Memilih kanal adalah keputusan risiko, bukan keputusan selera.

Gambar 2.1 — Empat kanal pada dua sumbu. Tidak ada kanal yang benar untuk semua orang; yang ada hanyalah kanal yang cocok dengan toleransi risiko Anda.

Kanal	Sifat	Pantas untuk
stable	Rilis yang sudah lulus validasi penuh	Mesin produksi dan siapa pun yang tidak ingin terkejut
extended-stable	Tertahan lebih lama sebelum naik	Lingkungan yang perubahannya harus dijadwalkan
beta	Fitur baru sebelum stabil	Mesin uji, bukan mesin yang melayani pelanggan
dev	Mengikuti pengembangan harian	Kontributor dan orang yang sanggup memperbaiki sendiri

Nilai yang hilang atau tidak dikenali diperlakukan sebagai stable. Itu default yang aman, dan buku ini menyarankan Anda membiarkannya begitu sampai ada alasan konkret untuk berpindah.

Kanal juga mengatur aplikasi Mac

Pembaruan aplikasi macOS mengikuti setelan kanal pada Gateway. Kanal beta dan dev ikut menarik build beta aplikasi. Kanal extended-stable hanya menerima rilis aplikasi extended-stable, sehingga ia akan diam ketika tidak ada rilis aplikasi yang cocok — dan diamnya itu normal, bukan kerusakan.

BAB 2 / PERUBAHAN PEMUTUS

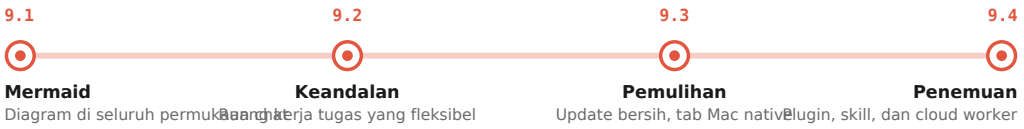
Yang berubah di v2026.9.4

Rilis ini membawa satu perubahan pemutus yang menyentuh hampir semua pemasangan berbasis Node, dan mengabaikannya menghasilkan kerusakan data yang sunyi.

WAJIB DIBACA SEBELUM UPDATE

Runtime Node sekarang menuntut Node 24.16.0 atau lebih baru pada jalur 24.x, atau Node 26.1.0 atau lebih baru. Node 26 adalah yang disarankan. Perbarui Node sebelum memperbarui OpenClaw, untuk mencegah pemotongan teks pada SQLite. Node 22, Node 25, dan build 24.x atau 26.x yang lebih lama tidak lagi didukung.

Frasa “pemotongan teks pada SQLite” layak dibaca dua kali. Kegagalannya tidak berisik: tidak ada pesan galat mencolok, hanya isi percakapan yang terpotong di penyimpanan. Karena itu urutannya penting — Node dulu, OpenClaw kemudian.



Gambar 2.2 — Empat rilis terakhir. Temanya konsisten: membuat pembaruan dan pemulihan berhenti menjadi titik lemah.

Rilis	Yang menonjol
v2026.9.4	Penemuan plugin dan skill, pembelajaran skill yang terlihat, kontrol cloud worker, dan pertanyaan terminal
v2026.9.3	Pemulihan pembaruan yang bersih, sambungan sesi lebih cepat, otomasi browser langsung, tautan chat yang bisa dicabut, dan tab Mac native
v2026.9.2	Perbaikan keandalan dan pemulihan, serta ruang kerja tugas yang fleksibel
v2026.9.1	Diagram Mermaid di seluruh permukaan chat, pengalaman Android yang lebih utuh, dan pemulihan pembaruan yang lebih aman

Pola yang terlihat dari empat rilis terakhir: sebagian besar pekerjaan tertuju pada keandalan pembaruan dan pemulihan. Itu tanda proyek yang sedang mendewasakan diri, dan juga tanda bahwa pembaruan pernah menjadi titik lemahnya. Bab 6 membahas cara memperbarui tanpa kehilangan apa pun.

BAB 2 / PRAKTIKUM

Menyiapkan disiplin pembaruan

Praktikum ini menghasilkan satu berkas yang akan Anda pakai setiap kali memperbarui. Menuliskannya sekarang, saat tidak ada tekanan, jauh lebih baik daripada menyusunnya di tengah pembaruan yang bermasalah.

Langkah 1 — catat keadaan sekarang

```
openclaw --version          > catatan-versi.txt
node --version             >> catatan-versi.txt
sw_vers                    >> catatan-versi.txt
openclaw config get update.channel >> catatan-versi.txt
```

Empat angka yang menentukan apakah sebuah rilis aman untuk mesin Anda.

Langkah 2 — tetapkan kanal dengan sengaja

Jangan biarkan kanal menjadi nilai bawaan yang tidak pernah Anda putuskan. Tuliskan alasan pemilihannya di catatan yang sama, karena penerus Anda akan bertanya.

```
openclaw config get update.channel
openclaw config set update.channel stable
```

Nilai yang hilang atau tidak dikenali diperlakukan sebagai stable.

Langkah 3 — susun prosedur pembaruan Anda

Urutan	Tindakan	Bukti yang dicatat
1	Baca catatan rilis, cari kata Breaking	Tautan rilis dan ringkasannya
2	Naikkan Node bila dituntut	Keluaran node --version sesudahnya
3	Pastikan cadangan dapat dipulihkan	Tanggal uji pemulihan terakhir
4	Perbarui pada jam sepi	Jam mulai dan selesai
5	Kirim satu pesan uji lewat kanal utama	Tangkapan balasannya

Kolom ketiga adalah yang membedakan prosedur dari niat baik. Tanpa bukti yang dicatat, tidak ada cara membedakan langkah yang dikerjakan dari langkah yang diasumsikan sudah dikerjakan.

BAB 2 / STUDI KASUS

Teks yang hilang tanpa pesan galat

Sebuah tim memperbarui OpenClaw pada Jumat sore. Pembaruan berhasil, agen menjawab normal, dan semua orang pulang. Node di mesin itu masih versi lama karena tidak ada yang membaca catatan rilis.

Selama dua minggu berikutnya, sebagian isi percakapan tersimpan terpotong di penyimpanan. Tidak ada galat. Agen tetap menjawab, hanya saja ia kehilangan sebagian riwayat yang lebih panjang. Masalah baru terlihat ketika seorang pelanggan merujuk percakapan lama dan agen tidak mengenalinya.

YANG TERLIHAT		YANG SEBENARNYA
Berhasil	**Pembaruan**	Berhasil
Menjawab normal	**Agen**	Menjawab normal
Tidak ada	**Galat**	Tidak ada
Terlihat utuh	**Riwayat**	Terpotong di penyimpanan
Hari ke-14	**Terdeteksi**	Sejak hari pertama

Gambar 2.3 — Dua minggu selisih antara kejadian dan deteksi. Kerusakan yang sunyi selalu lebih mahal daripada kerusakan yang berisik.

Pemulihannya mahal: dua minggu riwayat tidak dapat dikembalikan, dan tim harus menjelaskan kepada pelanggan mengapa agen tampak melupakan percakapan. Biaya sebenarnya bukan pada datanya, melainkan pada kepercayaan.

MENGAPA URUTANNYA TIDAK BOLEH DIBALIK

Perubahan pemutus yang menghasilkan kerusakan sunyi adalah alasan mengapa langkah pertama prosedur pembaruan adalah membaca catatan rilis, bukan menjalankan perintah. Lima menit membaca menghemat dua minggu kerusakan.

BAB 2 / LATIHAN

Latihan mandiri

1. Jalankan empat perintah pada praktikum langkah satu dan simpan hasilnya. Tetapkan tanggal untuk menjalankannya ulang setiap bulan.
2. Tuliskan alasan pemilihan kanal rilis Anda dalam dua kalimat, seolah menjelaskannya kepada orang yang akan menggantikan Anda tahun depan.
3. Cari catatan rilis dua versi terakhir OpenClaw dan tandai setiap perubahan yang menyentuh konfigurasi yang Anda pakai.
4. Rancang cara mendeteksi kerusakan sunyi seperti pada studi kasus dalam waktu kurang dari dua puluh empat jam, bukan dua minggu.

BAB 2 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Mencari rilis “2.2” dan tidak ketemu	Tidak ada penomoran semantik. Cocokkan dengan format tahun.bulan.rilis.
Aplikasi Mac menolak menyambung setelah update	Penolakan versi protokol. Baca pesannya; ia menyebut sisi mana yang harus naik.
extended-stable tidak pernah menawarkan update aplikasi	Normal bila belum ada rilis aplikasi extended-stable yang cocok.
Teks percakapan terpotong setelah update	Versi Node di bawah batas minimum. Naikkan Node, lalu jalankan pemeriksaan ulang.
Ingin tahu apa yang berubah sebelum menekan update	Catatan rilis tersedia dalam dua bentuk: untuk manusia dan Markdown untuk alat.

Tiga pertanyaan review

1. Mengapa nomor versi OpenClaw tidak bisa dipakai untuk memperkirakan besarnya risiko sebuah pembaruan?
2. Organisasi Anda mewajibkan setiap perubahan dijadwalkan lewat komite. Kanal mana yang paling sesuai, dan apa konsekuensinya?
3. Anda memperbarui OpenClaw lebih dulu, baru Node. Kerusakan seperti apa yang mungkin sudah terjadi, dan mengapa sulit terdeteksi?

Kunci pembahasan

Nomor kalender hanya menyatakan kapan rilis dibuat, sehingga besarnya perubahan harus dibaca dari catatan rilis, bukan dari selisih angka. Organisasi dengan komite perubahan paling cocok memakai extended-stable, dengan konsekuensi menerima fitur lebih lambat dan sesekali tidak mendapat pembaruan aplikasi sama sekali. Memperbarui OpenClaw sebelum Node berisiko menghasilkan teks terpotong di SQLite; sulit terdeteksi karena tidak memunculkan galat, hanya isi yang hilang diam-diam.

Rujukan: catatan rilis OpenClaw v2026.9.1 sampai v2026.9.4 dan halaman install/development-channels, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 3 Prasyarat Mac dan dua keputusan arsitektur awal

Dua keputusan diambil pada menit pertama instalasi, dan keduanya sulit diubah belakangan: di mana Gateway berjalan, dan siapa yang memilikinya. Bab ini membahas keduanya sebelum Anda mengunduh apa pun.

Tujuan belajar

Memeriksa apakah Mac Anda memenuhi syarat, memilih antara mode lokal dan remote, dan memahami perbedaan antara Gateway yang dimiliki aplikasi dan Gateway yang dikelola sendiri. Pada akhir bab, pembaca dapat menuliskan keputusan arsitekturnya dalam satu kalimat.

Syarat perangkat

Komponen	Syarat minimum
OpenClaw.app	macOS 15.0 (Sequoia) atau lebih baru
Helper openclaw-mac	Sama dengan aplikasi: macOS 15.0 ke atas
Voice Wake dan push-to-talk	macOS 26 atau lebih baru
CLI dan Gateway berbasis Node	Versi Node yang didukung; biner Node 24 dan 26 resmi untuk macOS menuntut macOS 13.5 ke atas
Membangun dari sumber	Toolchain pada panduan macOS dev setup

JEBAKAN YANG SERING TERJADI

Menjalankan CLI pada Mac yang lebih tua tidak membuat aplikasi native menjadi kompatibel dengan versi macOS tersebut. Keduanya punya batas sendiri, dan batas aplikasi lebih tinggi.

Bagi banyak organisasi, baris terakhir tabel itulah yang menentukan. Mac yang masih menjalankan macOS 13 atau 14 bisa menjadi host Gateway lewat CLI, tetapi tidak bisa menjalankan aplikasi menu bar — dan karena itu kehilangan node Mac, izin TCC, Quick Chat, serta kemampuan perangkat yang dibahas di Bab 12.

BAB 3 / KEPUTUSAN PERTAMA

Lokal atau remote

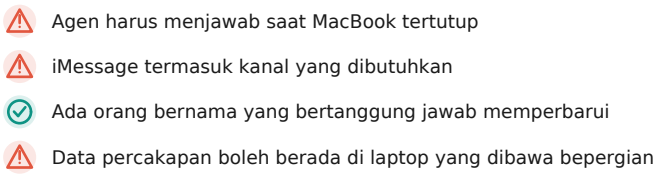
Keputusan pertama: apakah Mac ini yang menjalankan Gateway, atau ia hanya mengendalikan Gateway yang berjalan di tempat lain.

MODE LOCAL		MODE REMOTE
Mac ini	**Gateway di**	Host lain
launchd	**Dijaga oleh**	Host itu sendiri
Ya	**Perlu CLI**	Tidak
Agen ikut mati	**Saat Mac tidur**	Agen tetap hidup
Mac mini yang menetap	**Cocok untuk**	MacBook yang dibawa
Tersedia	**Sinkronisasi cookie**	Perlu CLI eksternal

Gambar 3.1 — Dua mode penempatan. Baris keempat biasanya yang memutuskan, dan ia jarang disadari sebelum agen diam semalaman.

Node Mac pada aplikasi memakai runtime privat bawaannya pada kedua mode. Hanya penyiapan dan pengelolaan Gateway lokal milik aplikasi yang menuntut pemasangan CLI terpisah. Mode remote dan penyambungan ke Gateway lokal yang dikelola sendiri melewati pemasangan itu. Sinkronisasi cookie tetap menuntut CLI eksternal di Mac ini, dan layanan node terpisah yang sudah ada mempertahankan siklus hidup CLI-nya sendiri.

Pertanyaan yang memutuskan



Gambar 3.3 — Empat pertanyaan yang memutuskan penempatan. Tanda seru berarti jawabannya mengubah arsitektur Anda.

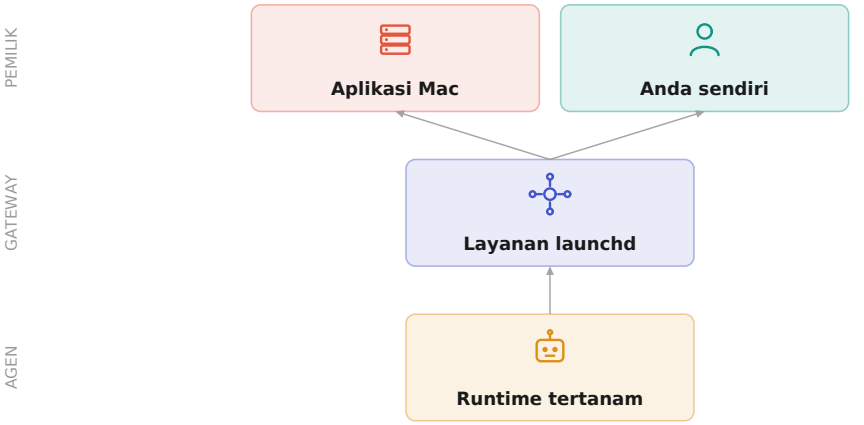
- Apakah agen harus menjawab saat MacBook Anda tertutup? Bila ya, Gateway tidak boleh di MacBook itu.
- Apakah iMessage termasuk kanal yang dibutuhkan? Bila ya, harus ada Mac yang hidup terus di suatu tempat.
- Siapa yang akan memperbarui mesin itu? Mesin tanpa pemilik bernama akan tertinggal versi dalam tiga bulan.
- Apakah data percakapan boleh berada di laptop yang dibawa bepergian? Ini pertanyaan

kepatuhan, bukan teknis.

BAB 3 / KEPUTUSAN KEDUA

Siapa yang memiliki Gateway

Keputusan kedua lebih halus dan lebih sering salah. Gateway lokal bisa dimiliki aplikasi, atau dikelola sendiri oleh Anda. Perbedaannya menentukan siapa yang melakukan pembaruan dan bagaimana pemulihan bekerja.



Gambar 3.2 — Dua jalur kepemilikan menuju satu layanan yang sama. Jalur menentukan siapa yang memperbarui dan siapa yang memperbaiki.

Bila aplikasi yang memiliki Gateway launchd lokal, kartu pembaruan pada dasbor akan menawarkan pembaruan terkoordinasi: aplikasi diperbarui dulu, dan setelah relaunch aplikasi memperbarui lalu menjalankan ulang Gateway-nya pada versi yang cocok, kemudian memverifikasi sambungannya.

Bila Anda yang mengelola Gateway sendiri — atau Gateway berada di mesin lain — tombol yang muncul menjalankan alur pembaruan normal Gateway tersebut, bukan mengubah aplikasi Mac. Perbaikan otomatis tidak akan menurunkan versi Gateway yang lebih baru, dan tidak akan menimpa pin kanal extended-stable.

Ketika Gateway milik aplikasi rusak

Tab Connection menawarkan tiga tindakan ketika Gateway lokal yang dikelola aplikasi hilang, usang, atau rusak: Install Gateway, Update Gateway, dan Repair Gateway. Ketiganya membuka prompt penyiapan yang sudah ada, menampilkan progres, lalu memeriksa Gateway sesudahnya. Anda dapat mencoba ulang di sana setelah sebelumnya membatalkan prompt.

BAB 3 / PRAKTIKUM

Memilih mesin dengan bukti

Praktikum ini mengubah keputusan penempatan dari perdebatan menjadi tabel. Isi kolomnya untuk setiap mesin yang Anda pertimbangkan, lalu keputusannya akan terlihat sendiri.

Langkah 1 — periksa syarat tiap kandidat

```
sw_vers                     # versi macOS
uname -m                    # arsitektur: arm64 atau x86_64
node --version              # bila Node sudah ada
pmset -g                    # perilaku tidur
sudo pmset -c sleep 0       # contoh: cegah tidur saat tersambung daya
```

Perintah keempat dan kelima sering menjadi penentu; mesin yang tidur bukan kandidat host.

Langkah 2 — isi tabel keputusan

Kriteria	Mesin A	Mesin B
macOS 15 ke atas		
Hidup dua puluh empat jam		
Punya pemilik bernama		
Boleh memuat data pelanggan		
Boleh dimatikan SIP-nya bila perlu		
Dapat dijangkau untuk perawatan fisik		

Tabel ini sengaja dibiarkan kosong. Mengisinya memaksa Anda menanyakan hal yang biasanya tidak ditanyakan sampai terlambat — terutama baris keempat dan kelima, yang sering menyentuh kebijakan perusahaan dan bukan spesifikasi perangkat.

Langkah 3 — tuliskan keputusan dalam satu kalimat

```
# Contoh isian
Gateway berjalan di Mac mini kantor (macOS 15.4, tidak pernah tidur),
dimiliki oleh tim IT, diakses dari MacBook staf lewat mode remote
melalui Tailscale. iMessage tidak dipakai, sehingga SIP tetap menyala.
```

Kalimat ini akan menjawab sebagian besar pertanyaan arsitektur berikutnya.

BAB 3 / STUDI KASUS

Gateway yang dimiliki dua pihak

Sebuah perusahaan memasang OpenClaw lewat aplikasi macOS di satu Mac mini. Beberapa bulan kemudian, seorang insinyur memasang CLI secara manual di mesin yang sama untuk menguji sesuatu, lalu menjalankan Gateway dari terminal.

Sejak itu, kadang ada dua proses yang bersaing, kadang aplikasi melaporkan Gateway yang bukan miliknya, dan pembaruan menghasilkan hasil yang berbeda tergantung siapa yang menjalankannya. Tidak ada yang rusak secara mencolok; semuanya hanya menjadi tidak dapat diprediksi.

Gejala yang muncul	Penjelasannya
Kartu pembaruan berubah-ubah bunyinya	Kepemilikan Gateway berbeda antar sesi
Konfigurasi kadang tidak berlaku	Dua proses membaca berkas yang sama pada waktu berbeda
Restart lewat aplikasi tidak berpengaruh	Yang berjalan adalah proses manual, bukan layanan launchd
Log tidak lengkap	Sebagian keluaran pergi ke terminal yang sudah ditutup

Perbaikannya sederhana setelah penyebabnya dipahami: hentikan proses manual, pastikan hanya satu Gateway yang berjalan, dan tetapkan secara tertulis siapa pemilik layanan itu. Yang mahal adalah minggu-minggu diagnosis sebelum seseorang menyadari ada dua.

PERIKSA INI LEBIH DULU

Hanya satu Gateway yang sebaiknya berjalan per host, kecuali Anda memang sengaja menjalankan profil yang terisolasi. Bila Anda mencurigai ada dua, periksa proses yang berjalan dan layanan launchd yang terpasang sebelum mendiagnosis hal lain.

BAB 3 / LATIHAN

Latihan mandiri

1. Isi tabel keputusan pada praktikum untuk dua mesin nyata di organisasi Anda, lalu tuliskan kalimat keputusan penempatannya.
2. Untuk mesin yang Anda pilih, tuliskan siapa yang akan memperbaruinya, seberapa sering, dan apa yang terjadi bila orang itu tidak ada.
3. Bila Anda memilih mode remote, gambarkan jalur jaringannya dari MacBook sampai Gateway dan tandai setiap titik yang menuntut autentikasi.
4. Tuliskan satu skenario di mana keputusan penempatan Anda akan menjadi salah, dan apa yang akan memicu Anda meninjaunya ulang.

BAB 3 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Aplikasi menolak dipasang di Mac kantor	Kemungkinan macOS di bawah 15.0. CLI masih bisa, aplikasi tidak.
Voice Wake tidak muncul di pengaturan	Menuntut macOS 26 ke atas; ini batas terpisah dari batas aplikasi.
Agen mati setiap malam	Gateway berada di laptop yang tidur. Pindahkan ke host yang hidup terus.
Pembaruan aplikasi tidak menyentuh Gateway	Gateway tidak dimiliki aplikasi. Jalankan alur pembaruan Gateway itu sendiri.
Perbaikan otomatis tidak menurunkan versi	Perilaku yang disengaja; ia juga tidak menimpa pin extended-stable.

Tiga pertanyaan review

1. Tim Anda memakai MacBook yang dibawa pulang setiap hari dan membutuhkan agen yang menjawab sepanjang malam. Rancang penempatan Gateway-nya dan sebutkan satu konsekuensi biayanya.
2. Apa perbedaan praktis antara Gateway yang dimiliki aplikasi dan yang dikelola sendiri, dilihat dari sudut orang yang harus memperbarui?
3. Mac dengan macOS 14 hendak dipakai sebagai host. Apa yang masih bisa dijalankan, dan kemampuan apa yang hilang?

Kunci pembahasan

Gateway harus berada di mesin yang tidak tidur — Mac mini, server, atau host remote — sementara MacBook cukup menjadi klien mode remote; konsekuensi biayanya adalah satu perangkat tambahan yang harus dirawat dan diperbarui. Gateway milik aplikasi memberi

pembaruan terkoordinasi satu tombol, sedangkan Gateway kelolaan sendiri menuntut Anda menjalankan alurnya sendiri, dengan imbalan kendali penuh atas waktu pembaruan. Mac dengan macOS 14 masih dapat menjalankan CLI dan Gateway, tetapi kehilangan aplikasi menu bar beserta node Mac, izin TCC, Quick Chat, dan kemampuan perangkat yang menyertainya.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos dan platforms/mac/bundled-gateway, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 4 Memasang aplikasi macOS dan menjalankan first run

Instalasi OpenClaw di Mac tidak sulit. Yang sulit adalah menahan diri untuk tidak menekan tombol berikutnya sebelum memahami apa yang baru saja Anda setujui. Bab ini berjalan melewati alur first run langkah demi langkah, dengan catatan pada setiap titik yang menentukan.

Tujuan belajar

Mengunduh build yang benar, melewati lima langkah first run dengan sadar, dan mengetahui ke mana harus kembali ketika satu langkah gagal. Pada akhir bab, pembaca memiliki agen yang menjawab di Mac-nya sendiri.

Mengambil berkasnya

Build aplikasi macOS diambil dari halaman rilis GitHub OpenClaw. Ketika sebuah rilis mengirimkan aset aplikasi macOS, carilah salah satu dari dua berkas ini:

```
OpenClaw-<versi>.dmg     # disarankan
OpenClaw-<versi>.zip
```

Pola nama berkas rilis aplikasi macOS.

Sebagian rilis hanya mengirimkan aset CLI, evidence, atau Windows. Bila rilis terbaru tidak punya aset aplikasi macOS, pakai rilis terbaru yang punya, atau bangun dari sumber. Ini bukan kerusakan; ini konsekuensi dari kadens rilis yang cepat.

BAB 4 / FIRST RUN

Lima langkah pertama

Gambar 4.1 — Alur first run. Langkah kedua adalah keputusan arsitektur dari Bab 3, dan di sinilah ia diterapkan.

1. Pasang dan jalankan OpenClaw.app.
2. Pilih This Mac untuk Gateway lokal, atau Connect to an existing Gateway untuk memasukkan alamatnya dan masuk. Gateway yang tersimpan akan membuka dasbornya setelah tersambung dan menyelesaikan penyiapan awal tanpa mengubah Gateway utama Mac ini.
3. Untuk Gateway lokal baru, tunggu sementara aplikasi memasang runtime CLI eksternalnya dan menjalankan Gateway. Menyambung ke Gateway remote atau Gateway lokal yang dikelola terpisah tidak menuntut pemasangan CLI di Mac ini.
4. Pilih koneksi AI yang Anda inginkan. Deteksi hanya menampilkan koneksi yang tersedia; memilih salah satunya memulai pemeriksaan model langsung. Rute yang sudah terkonfigurasi muncul sebagai Current model.
5. Selesai. Aplikasi membuka dasbor, dan OpenClaw memandu sisa penyiapan — impor memori, kanal, izin — dalam satu percakapan.

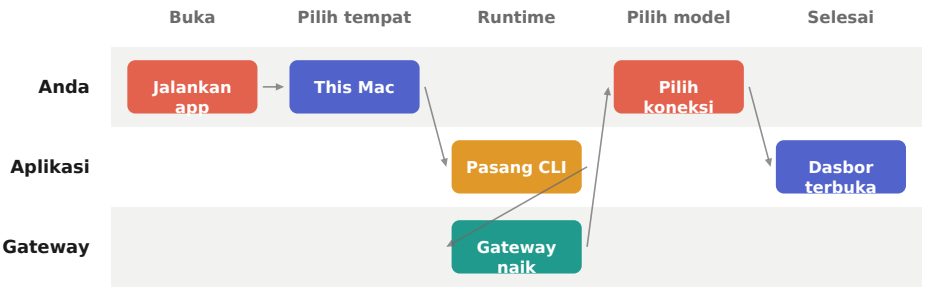
Langkah kelima layak diperhatikan. Sisa penyiapan tidak dilakukan lewat serangkaian formulir, melainkan lewat percakapan dengan agen itu sendiri. Ini nyaman, dan juga berarti Anda sedang menyetujui hal-hal sambil mengobrol. Baca setiap permintaan izin sebagaimana Anda membaca dialog instalasi.

PERSETUJUAN TIDAK PERNAH DIAM-DIAM
Menyalakan kapabilitas sensitif membuka konfirmasi native dengan Cancel sebagai pilihan default. Menutup atau mengganti halaman dasbor membatalkan persetujuan yang tertunda; minta ulang perubahan itu dari halaman yang sedang aktif.

BAB 4 / SETELAH TERPASANG

Tempat-tempat yang perlu Anda kenali

Setelah dasbor terbuka, ada beberapa lokasi yang akan Anda kunjungi berulang kali sepanjang sisa buku ini. Mengenalinya sekarang menghemat waktu nanti.



Gambar 4.2 — Siapa mengerjakan apa selama first run. Anda hanya mengambil tiga keputusan; sisanya berjalan sendiri.

Yang dicari	Tempatnya
Preferensi aplikasi, kapabilitas lokal, izin	Dashboard → Settings → This Mac
Kontrol suara perangkat	Settings → Talk → This Mac
Preferensi pembaruan aplikasi	Settings → Updates → This Mac
Status Gateway lokal, remote/SSH, Tailscale, discovery	Connection… → tab Connection
Profil Gateway tersimpan	Connection… → tab Gateways
Versi aplikasi dan informasi build	About OpenClaw

Kontrol perangkat pada baris pertama sampai ketiga hanya muncul di dasbor tertanam aplikasi macOS, bukan di peramban biasa. Bila Anda membuka dasbor lewat Chrome dan tidak menemukan This Mac, itu bukan kesalahan Anda.

Membuka pengaturan dengan cepat

```
Cmd-,               # buka Dashboard settings
Option-Space        # Quick Chat, komposer gaya Spotlight
openclaw dashboard  # buka Control UI dari terminal
```

Tiga jalan pintas yang akan paling sering dipakai.

BAB 4 / PRAKTIKUM

First run yang terdokumentasi

Instalasi pertama akan Anda ulangi — pada mesin lain, setelah kerusakan, atau oleh orang lain. Praktikum ini meminta Anda mendokumentasikannya sambil mengerjakan, ketika seluruh detailnya masih segar.

Langkah 1 — catat sebelum memulai

```
# Tuliskan di catatan instalasi
Tanggal        :
Mesin          :
Versi macOS    :
Berkas rilis   : OpenClaw-<versi>.dmg
Mode Gateway   : This Mac / Connect to existing
Penyedia model :
```

Enam baris ini menjawab sebagian besar pertanyaan saat pemasangan diulang.

Langkah 2 — verifikasi tiap langkah selesai

Setelah langkah	Verifikasi	Bila gagal
Aplikasi terpasang	Ikon muncul di menu bar	Periksa syarat macOS 15
Gateway terpasang	Status di tab Connection sehat	Ulangi lewat Install Gateway
Model terpilih	Pemeriksaan model langsung lulus	Periksa kredensial penyedia
Dasbor terbuka	Kartu akun menunjukkan Gateway yang benar	Periksa profil yang aktif
Agen menjawab	Kirim satu pesan lewat Quick Chat	Baca log lewat openclaw logs

Kolom verifikasi mencegah kesalahan yang paling umum pada instalasi: melanjutkan ke langkah berikutnya sambil berasumsi langkah sebelumnya berhasil. Ketika sesuatu gagal di langkah kelima, Anda ingin tahu bahwa langkah kedua benar-benar sudah terbukti.

Langkah 3 — periksa keadaan sesudahnya

```
openclaw --version
openclaw status
openclaw health
openclaw doctor
ls -la ~/.openclaw/
```

Perintah terakhir memperlihatkan direktori yang akan Anda cadangkan di Bab 21.

BAB 4 / STUDI KASUS

Persetujuan yang tidak pernah dibaca

Seorang operator menyelesaikan first run dalam sepuluh menit. Alur onboarding memandu penyiapan lewat percakapan, dan ia menjawab setiap permintaan dengan menekan tombol yang tersedia — termasuk beberapa permintaan izin macOS yang muncul di sela percakapan.

Dua bulan kemudian, tim keamanan menanyakan mengapa aplikasi memiliki izin Accessibility dan Screen Recording. Tidak ada yang dapat menjawab mengapa izin itu diberikan, untuk apa dipakai, atau apakah masih diperlukan.

Yang terjadi	Yang seharusnya
Izin diberikan sambil mengobrol	Izin diberikan sebagai keputusan terpisah yang dicatat
Tidak ada catatan alasannya	Setiap izin dicatat beserta kemampuan yang membutuhkannya
Tidak ada peninjauan berkala	Izin ditinjau saat audit kuartalan
Tidak ada yang tahu cara mencabutnya	Prosedur pencabutan dicatat bersama pemberiannya

Alur onboarding yang berbentuk percakapan memang nyaman, dan kenyamanan itulah risikonya: menyetujui sesuatu terasa seperti menjawab pertanyaan, bukan seperti memberi wewenang. Perlakukan setiap permintaan izin dengan kesungguhan yang sama seperti Anda membaca dialog instalasi perangkat lunak asing.

MENGAPA DEFAULT-NYA CANCEL

Konfirmasi kapabilitas sensitif memakai Cancel sebagai pilihan bawaan justru karena alasan ini. Bila Anda menekan setuju, lakukan dengan sadar dan catat alasannya di tempat yang dapat ditemukan orang lain.

BAB 4 / LATIHAN

Latihan mandiri

1. Jalankan first run sambil mengisi catatan instalasi pada praktikum. Simpan catatan itu di tempat yang dapat diakses rekan kerja Anda.
2. Buka Dashboard → Settings → This Mac → Permissions dan daftarkan setiap izin yang aktif beserta alasan Anda memberikannya.
3. Cabut satu izin yang tidak Anda perlukan, lalu amati kemampuan apa yang berhenti bekerja. Catat hasilnya; ini pemetaan yang berguna nanti.
4. Tuliskan prosedur pemasangan ulang dalam sepuluh baris atau kurang, lalu minta rekan kerja mengikutinya tanpa bertanya kepada Anda.

BAB 4 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Rilis terbaru tidak punya berkas .dmg	Normal. Pakai rilis terbaru yang membawa aset macOS, atau bangun dari sumber.
Tidak ada koneksi AI yang terdeteksi	Deteksi hanya menampilkan yang tersedia. Siapkan kredensial penyedia lebih dulu.
Menu This Mac tidak ada di dasbor	Dasbor dibuka di peramban biasa. Kontrol perangkat hanya ada di dasbor tertanam.
Persetujuan hilang sebelum sempat ditekan	Halaman dasbor berganti. Minta ulang perubahan dari halaman yang sedang aktif.
Gateway lokal gagal dipasang saat first run	Ulangi dari tab Connection lewat Install Gateway; prompt yang sama akan terbuka.

Tiga pertanyaan review

1. Pada langkah keempat, mengapa daftar koneksi AI bisa tampak kosong padahal Anda punya langganan yang aktif?
2. Anda menyambung ke Gateway yang sudah ada milik tim. Bagian mana dari alur first run yang dilewati, dan mengapa?
3. Mengapa default tombol pada konfirmasi kapabilitas sensitif adalah Cancel, dan apa yang akan terjadi bila defaultnya sebaliknya?

Kunci pembahasan

Daftar koneksi hanya menampilkan koneksi yang benar-benar terdeteksi di mesin itu, sehingga langganan yang kredensialnya belum tersedia secara lokal tidak akan muncul. Menyambung ke Gateway yang sudah ada melewati pemasangan runtime CLI, karena Gateway itu sudah berjalan dan dikelola di tempat lain; penyiapan awal pun selesai tanpa

mengubah Gateway utama Mac ini. Default Cancel ada karena persetujuan kapabilitas sensitif harus merupakan tindakan sadar; bila defaultnya menyetujui, satu penekanan Enter yang tidak sengaja dapat memberi agen akses yang tidak pernah Anda maksudkan.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos bagian Download, First run, dan Connection, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

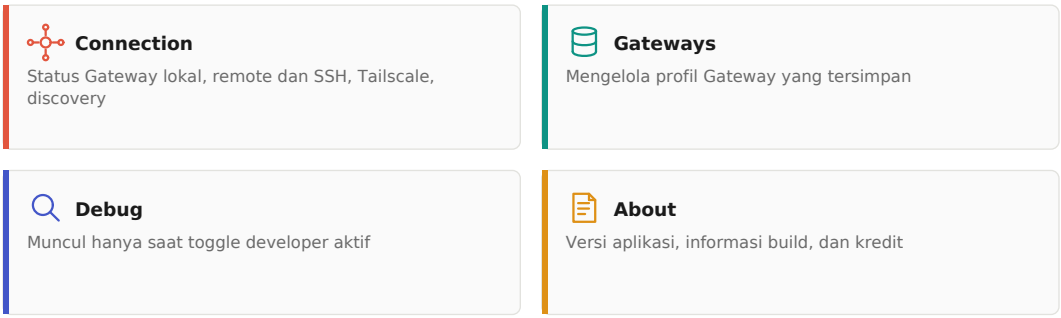
Bab 5 Jendela Connection, profil Gateway, dan mode sambungan

Satu Mac bisa berbicara dengan beberapa Gateway. Kemampuan ini nyaman ketika Anda punya Gateway pribadi di rumah dan Gateway tim di kantor, dan berbahaya ketika Anda lupa sedang menyambung ke yang mana. Bab ini membahas jendela yang mengatur semuanya.

Tujuan belajar

Membuka jendela Connection ketika Gateway tidak dapat dihubungi, menambah dan berpindah antar profil Gateway, dan membaca kartu akun untuk memastikan Anda sedang berbicara dengan mesin yang benar. Pada akhir bab, pembaca dapat memulihkan sambungan yang putus tanpa memasang ulang apa pun.

Empat tab



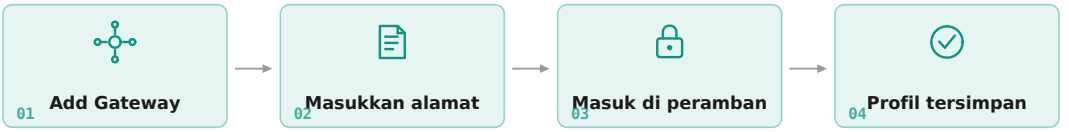
Gambar 5.1 — Empat tab jendela Connection. Tab Debug tersembunyi sampai Anda menyalakannya di This Mac → Developer.

PINTU YANG SELALU TERBUKA
Jendela Connection dapat dibuka meskipun Gateway sedang tidak dapat dihubungi. Ini disengaja: justru pada saat itulah Anda paling membutuhkannya. Jangan mencari jalan masuk lewat dasbor ketika dasbor sendiri tidak bisa dimuat.

BAB 5 / MENAMBAH GATEWAY

Menambah dan berpindah antar Gateway

Untuk menambah Gateway lain demi akses dasbor dan chat, buka Connection… → Gateways → Add Gateway, lalu masukkan hostname atau alamat HTTPS-nya. Gateway yang memakai Cloudflare Access memungkinkan Anda masuk dengan akun pribadi di peramban default. Anda juga dapat memulainya dari Get the apps → Open in Mac app pada situs web Gateway tersebut.



Gambar 5.2 — Menambah Gateway dengan sign-in peramban. Sesi tersimpan di Keychain, sehingga tidak perlu masuk dua kali.

Untuk Gateway tersimpan yang ditambahkan lewat sign-in peramban, dasbor memakai sesi pribadi berbasis Keychain milik profil itu. Anda tidak perlu masuk untuk kedua kalinya di dalam peramban tertanam. Tombol Reconnect di Connection… → Gateways memperbarui sesi yang kedaluwarsa.

Kartu akun: cara memastikan Anda di mesin yang benar

Kartu akun di kiri bawah dasbor menampilkan nama Anda dan Gateway yang sedang aktif, termasuk kesehatannya dan status primary. Saat terputus, ia menampilkan Reconnecting…. Buka bagian Gateway pada kartu itu untuk berpindah Gateway, Command-klik atau Control-klik sebuah Gateway untuk membukanya di jendela lain, atau pilih Gateway settings…. Pilihan Set as primary… muncul ketika Gateway saat ini dapat dipromosikan.

Kontrol ini tersedia bahkan ketika hanya ada satu Gateway tersimpan. Biasakan melirik kartu itu sebelum mengetik perintah yang mengubah keadaan.

BAB 5 / PEMULIHAN

Ketika Gateway lokal bermasalah

Bila Gateway lokal yang dikelola aplikasi hilang, usang, atau rusak, tab Connection menawarkan tiga tindakan. Ketiganya membuka prompt penyiapan yang sudah ada, menampilkan progres pemasangan, lalu memeriksa Gateway sesudahnya.

Tindakan	Dipakai ketika
Install Gateway	Gateway lokal belum ada sama sekali
Update Gateway	Gateway ada tetapi versinya tertinggal
Repair Gateway	Gateway ada tetapi tidak berjalan benar
Set Up Gateway	Gateway lebih baru daripada aplikasi dan tidak kompatibel

Anda dapat mencoba lagi di sini setelah sebelumnya membatalkan sebuah prompt. Untuk Gateway yang tidak kompatibel karena lebih baru daripada aplikasi, Set Up Gateway memberi kesempatan meninjau pilihan penyiapan sebelum melanjutkan. Gateway yang dikelola secara terpisah mempertahankan alur pembaruannya sendiri.

Penolakan versi protokol

Bila sambungan Gateway utama menolak versi protokol aplikasi, aplikasi menampilkan peringatan pembaruan dan menyimpan penjelasannya pada status sambungan. Penyiapan remote dan probe sambungan menampilkan panduan yang sama secara inline. Pesannya menyebut rilis aplikasi dan kedua versi protokol, serta memberi tahu sisi mana yang perlu diperbarui.

```
# bila Gateway yang tertinggal
openclaw update

# bila aplikasi yang tertinggal
# pasang build Mac yang lebih baru dari halaman rilis
```

Dua arah perbaikan; pesan galat menyebut arah mana yang benar.

MEMBACA PESANNYA DENGAN BENAR

Handshake yang ditolak mungkin tidak melaporkan versi rilis Gateway. Aplikasi menandai informasi itu sebagai tidak tersedia. Perbedaan nomor rilis saja tidak memicu peringatan ini — hanya ketidakcocokan protokol.

BAB 5 / PRAKTIKUM

Menyiapkan dua profil Gateway

Praktikum ini membangun keadaan yang akan Anda pakai sepanjang buku: satu profil lokal untuk percobaan, satu profil jarak jauh untuk pekerjaan nyata. Memisahkan keduanya sejak awal mencegah percobaan Anda menyentuh mesin yang melayani orang.

Langkah 1 — pastikan profil lokal sehat

```
openclaw health
openclaw gateway status
```

Jalankan dari Mac yang menjalankan Gateway lokal.

Langkah 2 — tambahkan profil kedua

Buka Connection… → Gateways → Add Gateway, lalu masukkan hostname atau alamat HTTPS Gateway kedua. Setelah tersimpan, profil itu muncul pada kartu akun di kiri bawah dasbor.

Langkah 3 — biasakan membaca kartu akun

Yang ditampilkan kartu	Yang harus Anda periksa
Nama Gateway	Apakah ini mesin yang Anda maksud
Status kesehatan	Apakah ia benar-benar tersambung
Penanda primary	Apakah ia Gateway utama Mac ini
Reconnecting…	Berarti sambungan sedang putus

Latih kebiasaan ini sekarang, ketika taruhannya rendah. Pada Bab 61, perintah yang Anda ketik akan merestart layanan produksi, dan kebiasaan melirik kartu akun sebelum menekan enter adalah yang memisahkan operator berpengalaman dari yang belum.

Langkah 4 — simulasikan Gateway yang tidak terjangkau

```
openclaw gateway stop      # pada mesin Gateway lokal
```

Lalu buka Connection… dari aplikasi dan amati bahwa jendela tetap terbuka.

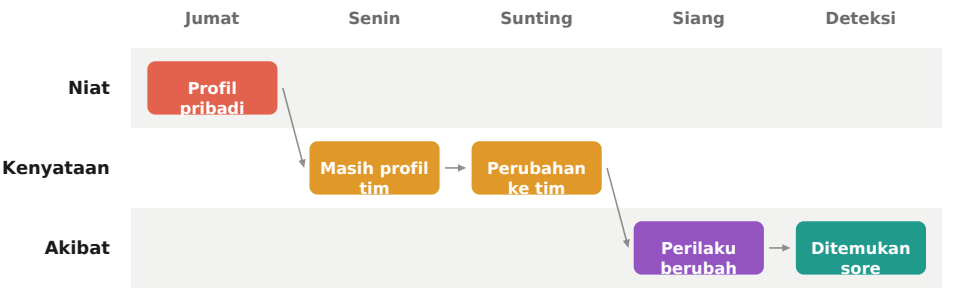
Amati apa yang masih dapat Anda lakukan dalam keadaan itu: jendela Connection terbuka, profil dapat diganti, dan tindakan Install, Update, atau Repair Gateway tersedia. Mengetahui ini sebelum terjadi insiden menghemat kepanikan.

BAB 5 / STUDI KASUS

Perintah yang mendarat di mesin yang salah

Seorang insinyur memiliki dua profil tersimpan: Gateway pribadi di rumah dan Gateway tim di kantor. Ia menjalankan percobaan konfigurasi pada Jumat malam, lalu menutup laptop tanpa berpindah profil.

Senin pagi ia membuka dasbor dan melanjutkan pekerjaannya, mengira ia berada pada profil pribadi. Ia menyunting kebijakan tool dan menyalakan satu plugin untuk diuji. Kedua perubahan itu mendarat pada Gateway tim.



Gambar 5.3 — Selisih antara niat dan kenyataan bertahan selama hampir satu hari kerja penuh.

Tidak ada kerusakan permanen, tetapi tim menghabiskan setengah hari mencari mengapa agen berperilaku berbeda. Penyebabnya baru ketahuan ketika seseorang memeriksa riwayat perubahan konfigurasi dan menemukan suntingan yang tidak ada yang mengaku membuatnya.

Pencegahan	Biaya penerapannya
Membaca kartu akun sebelum tiap perubahan	Dua detik
Memberi nama profil yang sangat berbeda	Sekali setel
Memakai profil terpisah untuk percobaan	Satu mesin tambahan
Menjadikan konfigurasi sebagai repositori git	Sekali setel; lihat Bab 16

MENGAPA RIWAYAT KONFIGURASI BERHARGA
Baris terakhir adalah yang menyelamatkan tim ini. Karena konfigurasi berada di bawah kendali versi, mereka dapat melihat persis apa yang berubah dan kapan — dan mengembalikannya dengan satu perintah.

BAB 5 / LATIHAN

Latihan mandiri

1. Tambahkan satu profil Gateway kedua, lalu berpindah bolak-balik lima kali sambil memperhatikan kartu akun setiap kali.
2. Hentikan Gateway lokal Anda, lalu daftarkan setiap tindakan yang masih dapat dilakukan dari jendela Connection.
3. Beri nama kedua profil Anda sedemikian rupa sehingga tidak mungkin tertukar sekilas. Hindari nama yang berbeda satu huruf.
4. Tuliskan prosedur singkat yang Anda ikuti sebelum menjalankan perubahan yang mengubah keadaan pada Gateway mana pun.

BAB 5 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Dasbor tidak bisa dibuka sama sekali	Pakai jendela Connection; ia terbuka meski Gateway tidak terjangkau.
Diminta masuk dua kali untuk Gateway yang sama	Profil belum memakai sesi peramban tersimpan. Tambahkan lewat sign-in peramban.
Sesi Gateway kedaluwarsa	Reconnect di Connection… → Gateways memperbarui sesinya.
Tab Debug tidak terlihat	Nyalakan toggle developer di This Mac → Developer.
Salah mengirim perintah ke Gateway yang keliru	Biasakan membaca kartu akun kiri bawah sebelum menjalankan perubahan.

Tiga pertanyaan review

1. Mengapa jendela Connection sengaja dibuat dapat dibuka tanpa sambungan Gateway yang sehat?
2. Anda melihat peringatan pembaruan setelah memperbarui aplikasi Mac. Bagaimana menentukan sisi mana yang harus dinaikkan?
3. Apa risiko praktis dari menyimpan beberapa profil Gateway, dan kebiasaan apa yang menekan risiko itu?

Kunci pembahasan

Jendela Connection dibuat selalu dapat dibuka karena justru saat Gateway tidak terjangkau-lah Anda membutuhkan alat untuk memperbaikinya; menaruhnya di dalam dasbor akan membuatnya ikut hilang bersama masalahnya. Sisi yang harus dinaikkan disebut langsung pada pesan peringatan, yang memuat rilis aplikasi dan kedua versi protokol. Risiko beberapa profil adalah menjalankan perubahan pada mesin yang salah; kebiasaan yang menekannya adalah membaca kartu akun di kiri bawah sebelum setiap tindakan yang mengubah keadaan.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos bagian Connection, Updates, dan Open dashboard links, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 6 Pembaruan terkoordinasi aplikasi dan Gateway

Empat rilis terakhir OpenClaw menghabiskan sebagian besar tenaganya pada satu tema: membuat pembaruan tidak merusak apa pun. Itu memberi tahu dua hal sekaligus — bahwa pembaruan pernah menjadi titik lemah, dan bahwa sekarang ada mekanisme yang layak dipercaya bila Anda memahaminya.

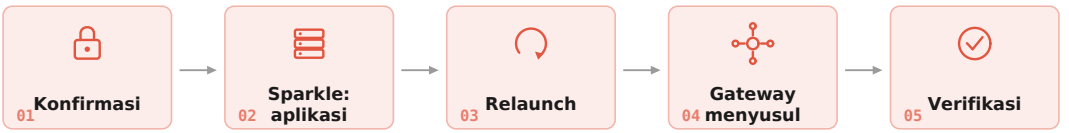
Tujuan belajar

Membedakan dua jenis pembaruan yang ditawarkan kartu dasbor, menjalankan pembaruan terkoordinasi dengan aman, dan memulihkan keadaan ketika pembaruan gagal. Pada akhir bab, pembaca dapat memperbarui mesin produksi tanpa jendela pemeliharaan yang panjang.

Dua tombol yang berbeda arti

Yang tertulis	Artinya
Update Mac app + Gateway	Aplikasi bertanda tangan memiliki Gateway launchd lokal. Sparkle memperbarui aplikasi lebih dulu; setelah relaunch, aplikasi otomatis memperbarui dan menjalankan ulang Gateway-nya pada versi yang cocok, lalu memverifikasi sambungan.
Update Gateway	Aplikasi tersambung ke Gateway remote, Gateway lokal yang dikelola manual, atau pemasangan lain yang bukan miliknya. Tombol menjalankan alur pembaruan normal Gateway tersebut, bukan mengubah aplikasi Mac.

Kedua tombol meminta konfirmasi lebih dulu. Kartu menyerahkan pembaruan kepada aplikasi hanya setelah Anda memilih Update Mac app and restart, sehingga salah klik tidak pernah memulai Sparkle.



Gambar 6.1 — Urutan pembaruan terkoordinasi. Aplikasi selalu naik lebih dulu; Gateway menyesuaikan ke versi yang cocok setelahnya.

BAB 6 / KEGAGALAN

Ketika pembaruan gagal

Pembaruan terkoordinasi yang gagal tetap berada di jendela bergaya penyiapan, lengkap dengan tindakan retry, tautan panduan pembaruan, dan tautan Discord. Ia tidak menghilang dan meninggalkan Anda menebak.

APA YANG TIDAK AKAN DILAKUKANNYA

Perbaikan otomatis tidak pernah menurunkan versi Gateway yang lebih baru, dan tidak menimpa pin kanal extended-stable. Bila Anda sengaja menahan sebuah mesin di versi tertentu, mekanisme perbaikan menghormati keputusan itu.

Setelah pembaruan berhasil

Setelah pembaruan sukses, aplikasi mencari sesi langsung tingkat atas yang paling terakhir dipakai manusia, lalu memberi agen tersebut satu kali peristiwa pembaruan. Aktivitas heartbeat dan cron tidak memengaruhi pilihan itu. Agen kemudian dapat menyambut Anda kembali dari percakapan yang kemungkinan besar sedang Anda pakai.

Pada mode remote, layanan node yang dipasang terpisah dan dikelola aplikasi mempertahankan alur pembaruan dan pemulihannya sendiri. Aplikasi melewati notifikasi ketika Gateway remote lebih tua daripada aplikasi. Node worker privat milik aplikasi ikut diperbarui bersama bundel aplikasi itu sendiri.

BAB 6 / KEBIASAAN

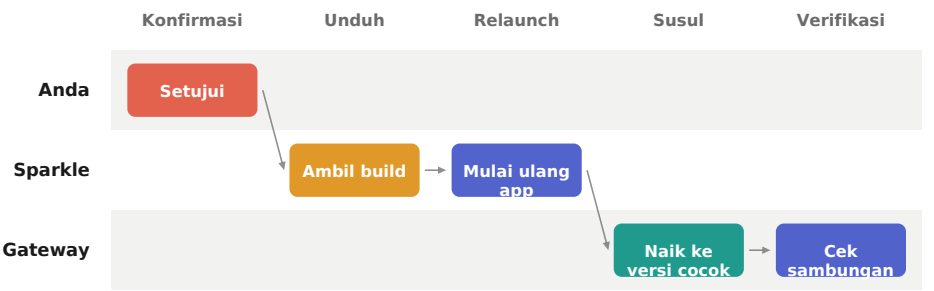
Urutan yang menyelamatkan

Bab 2 sudah menyebut perubahan pemutus pada runtime Node. Di sini urutannya dirangkum menjadi prosedur yang dapat Anda tempel di dinding.

- Catatan rilis dibaca, kata Breaking dicari lebih dulu
- Node dinaikkan bila rilis menuntutnya — Node dulu, selalu
- Cadangan sudah diuji pulih, bukan sekadar ada
- Dijalankan pada jam sepi, bukan jam puncak pesan
- Satu pesan uji dikirim lewat kanal utama sesudahnya

Gambar 6.3 — Lima langkah yang layak ditempel di dinding ruang operasi. Urutannya bukan saran; urutannya adalah pengamannya.

1. Baca catatan rilis. Cari kata Breaking sebelum apa pun yang lain.
2. Naikkan Node bila rilis menuntutnya. Node dulu, selalu.
3. Pastikan ada cadangan yang benar-benar dapat dipulihkan, bukan sekadar ada.
4. Perbarui pada jam sepi, bukan pada jam pelanggan paling banyak mengirim pesan.
5. Setelah selesai, verifikasi sambungan dan kirim satu pesan uji lewat kanal utama.



Gambar 6.2 — Pembaruan terkoordinasi dilihat per pelaku. Gateway tidak pernah bergerak lebih dulu.

Kebiasaan buruk	Akibat yang sering muncul
Memperbarui tanpa membaca catatan rilis	Perubahan pemutus ditemukan setelah pelanggan mengeluh
Memperbarui OpenClaw sebelum Node	Teks percakapan terpotong di SQLite tanpa pesan galat
Menganggap cadangan ada tanpa pernah mengujinya	Cadangan ternyata tidak dapat dipulihkan justru saat dibutuhkan

Kebiasaan buruk	Akibat yang sering muncul
Memperbarui beberapa mesin sekaligus	Tidak ada mesin pembanding ketika sesuatu berperilaku aneh
Melewatkan pesan uji setelah pembaruan	Kerusakan kanal baru diketahui berjam-jam kemudian

BAB 6 / PRAKTIKUM

Latihan pembaruan yang aman

Praktikum ini menjalankan pembaruan pada mesin yang tidak melayani siapa pun, supaya Anda mengenal bentuk prosesnya sebelum menjalankannya di tempat yang penting.

Langkah 1 — catat titik kembali

```
openclaw --version
node --version
openclaw backup
cd ~/.openclaw/workspace && git log --oneline -1
```

Empat perintah ini adalah titik kembali Anda bila sesuatu berjalan salah.

Langkah 2 — baca sebelum menekan

Buka catatan rilis untuk versi yang ditawarkan. Cari kata Breaking. Bila ada, baca seluruh paragrafnya, bukan hanya judulnya. Perubahan runtime Node pada v2026.9.4 adalah contoh yang judulnya terdengar teknis tetapi akibatnya menyentuh data.

Langkah 3 — jalankan dan amati

Tahap	Yang Anda amati	Berapa lama wajar
Konfirmasi	Kartu menyebut apa yang akan diperbarui	Seketika
Unduh dan pasang aplikasi	Progres Sparkle	Beberapa menit
Relaunch	Aplikasi tertutup lalu terbuka lagi	Di bawah satu menit
Gateway menyusul	Status berubah lalu pulih	Beberapa menit
Verifikasi	Sambungan dikonfirmasi	Seketika

Langkah 4 — buktikan agen masih bekerja

```
openclaw --version         # apakah benar naik
openclaw health
openclaw doctor
# lalu kirim satu pesan uji lewat kanal utama
```

Perintah terakhir adalah yang sesungguhnya membuktikan; tiga di atasnya hanya pendukung.

BAB 6 / STUDI KASUS

Sepuluh mesin yang diperbarui bersamaan

Sebuah organisasi menjalankan sepuluh Gateway untuk sepuluh tim. Tim platform memutuskan memperbarui semuanya pada malam yang sama untuk efisiensi — satu jendela pemeliharaan, satu pengumuman, selesai.

Pada pagi harinya, tiga tim melaporkan perilaku yang aneh pada kanal mereka. Tidak ada yang tahu apakah penyebabnya pembaruan, konfigurasi tim itu sendiri, atau kebetulan. Tidak ada mesin pembanding yang masih berjalan pada versi lama.

PERBARUI SEKALIGUS		PERBARUI BERTAHAP
Satu malam	**Jendela pemeliharaan**	Beberapa hari
Rendah	**Usaha koordinasi**	Lebih tinggi
Tidak ada	**Mesin pembanding**	Selalu ada
Menebak	**Diagnosis saat bermasalah**	Membandingkan
Sepuluh tim	**Radius kerusakan**	Satu tim

Gambar 6.4 — Efisiensi kolom kiri dibayar pada hari sesuatu tidak berjalan sesuai rencana.

Yang mereka lakukan berikutnya menjadi prosedur tetap: satu mesin perintis diperbarui lebih dulu dan diamati selama satu siklus kerja penuh, lalu sisanya menyusul bertahap. Jendela pemeliharaannya lebih panjang, dan tidak ada lagi pagi yang dihabiskan untuk menebak.

> **MEMILIH MESIN PERINTIS**
> Mesin perintis sebaiknya mesin yang benar-benar dipakai, bukan mesin uji yang menganggur. Masalah yang muncul pada pembaruan hampir selalu berkaitan dengan pemakaian nyata — beban, kanal aktif, konfigurasi yang tumbuh alami.

BAB 6 / LATIHAN

Latihan mandiri

1. Jalankan satu siklus pembaruan lengkap pada mesin percobaan, dengan mencatat waktu tiap tahap. Simpan angkanya sebagai acuan.
2. Tuliskan prosedur pembaruan organisasi Anda dalam bentuk yang dapat diikuti orang lain tanpa bertanya, termasuk urutan mesin.
3. Tetapkan siapa mesin perintis Anda dan mengapa, lalu tuliskan berapa lama ia diamati sebelum mesin berikutnya menyusul.
4. Rancang cara mengembalikan keadaan bila pembaruan gagal di tengah, dan ujilah rencana itu sekali pada mesin percobaan.

BAB 6 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Kartu hanya menawarkan Update Gateway	Aplikasi tidak memiliki Gateway itu. Normal untuk Gateway remote atau manual.
Sparkle tidak pernah menawarkan build baru	Periksa update.channel; extended-stable diam bila tidak ada rilis yang cocok.
Pembaruan gagal dan jendela masih terbuka	Itu perilaku yang benar. Pakai retry di jendela itu sebelum tindakan manual.
Agen menyapa di percakapan yang tidak diduga	Peristiwa pembaruan dikirim ke sesi manusia terakhir, bukan ke cron atau heartbeat.
Versi Gateway turun setelah perbaikan	Tidak seharusnya terjadi; perbaikan otomatis tidak menurunkan versi. Periksa ulang.

Tiga pertanyaan review

1. Mengapa aplikasi diperbarui lebih dulu dan Gateway menyusul, bukan sebaliknya?
2. Sebuah mesin sengaja dipin pada extended-stable. Apa yang akan dan tidak akan dilakukan perbaikan otomatis terhadapnya?
3. Anda mengelola sepuluh Mac. Rancang urutan pembaruan yang menyisakan kemampuan membandingkan ketika terjadi masalah.

Kunci pembahasan

Aplikasi naik lebih dulu karena aplikasilah yang memiliki dan menjalankan Gateway launchd; memperbarui Gateway lebih dulu akan membuat aplikasi lama berbicara dengan protokol yang lebih baru. Pada mesin ber-pin extended-stable, perbaikan otomatis akan memperbaiki kerusakan tetapi tidak akan menurunkan versi yang lebih baru maupun menimpa pin kanalnya. Untuk sepuluh Mac, perbarui satu mesin lebih dulu sebagai perintis, tunggu satu siklus kerja penuh, lalu naikkan sisanya secara bertahap sehingga selalu ada mesin pada versi lama sebagai pembanding.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos bagian Updates, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 7 Workspace agen: berkas yang menjadi ingatan kerja

OpenClaw membaca instruksi operasi dan ingatan dari direktori workspace-nya. Berkas di dalamnya bukan dokumentasi untuk manusia; ia adalah bahan yang benar-benar masuk ke dalam prompt. Menyunting berkas ini adalah cara paling langsung mengubah perilaku agen, dan juga cara paling cepat merusaknya.

Tujuan belajar

Menyebutkan setiap berkas bootstrap dan kapan ia dimuat, membedakan berkas yang dibuat otomatis dari yang tidak, dan menyiapkan strategi cadangan yang benar. Pada akhir bab, pembaca dapat menjelaskan mengapa sebuah agen berperilaku seperti yang dilakukannya.

Lokasi default

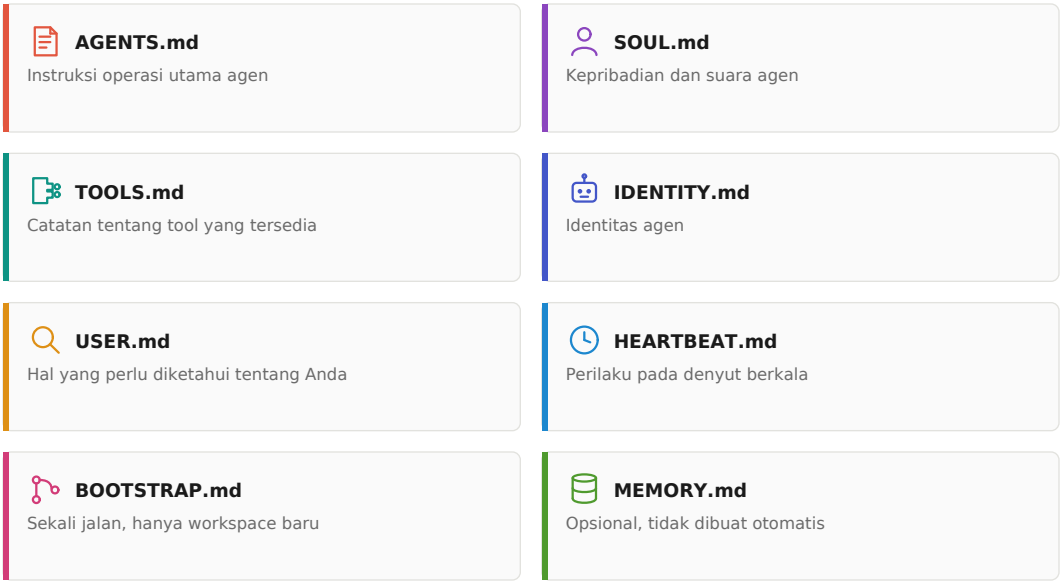
```
~/.openclaw/workspace          # workspace agen default
~/.openclaw/openclaw.json      # konfigurasi Gateway
~/.openclaw/workspace-work     # contoh workspace agen tambahan
```

Secara default OpenClaw memakai ~/.openclaw/workspace sebagai workspace agen.

Direktori itu dibuat otomatis pada penyiapan atau pada jalannya agen pertama kali, lengkap dengan berkas awalnya. Setiap agen memakai satu direktori workspace sebagai satu-satunya direktori kerja — cwd — untuk tool dan konteksnya.

BAB 7 / BERKAS

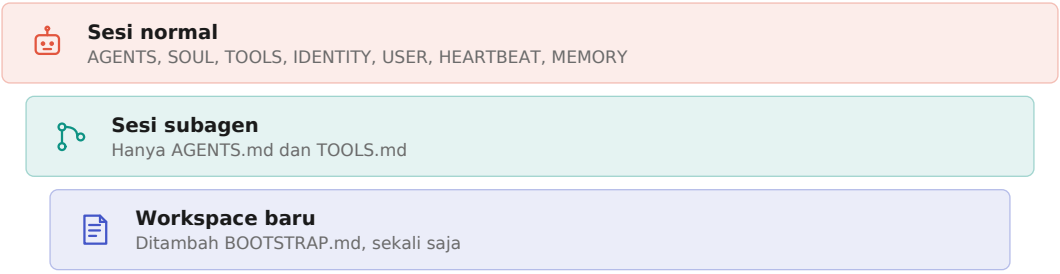
Tujuh berkas dan satu yang opsional



Gambar 7.1 — Delapan berkas workspace. Dua kotak terakhir adalah pengecualian yang paling sering menimbulkan kebingungan.

DUA PENGECUALIAN YANG PENTING
BOOTSTRAP.md hanya dibuat ketika workspace benar-benar baru, dan ia tidak akan kembali setelah Anda menghapusnya. MEMORY.md tidak dibuat otomatis; ketika ada, ia dimuat untuk sesi normal.

Sesi subagen hanya menyuntikkan AGENTS.md dan TOOLS.md. Ini keputusan desain yang punya akibat praktis: kepribadian dan ingatan pribadi tidak ikut ke subagen, sehingga subagen berperilaku lebih netral dan lebih murah. Bila subagen Anda terasa “lupa siapa dirinya”, itu bukan kerusakan.



Gambar 7.2 — Tiga konteks memuat berkas yang berbeda. Perbedaan inilah yang menjelaskan mengapa perilaku agen berubah antar jenis sesi.

BAB 7 / CADANGAN

Perlakukan folder ini seperti ingatan

Dokumentasi resmi memberi satu saran yang layak diikuti tanpa tawar-menawar: perlakukan folder ini seperti ingatan OpenClaw, dan jadikan ia repositori git — idealnya privat — sehingga AGENTS.md dan berkas memori Anda tercadangkan. Bila git terpasang, workspace yang benar-benar baru akan diinisialisasi otomatis.

```
cd ~/.openclaw/workspace
git status                    # periksa apakah sudah menjadi repo
git log --oneline -10         # riwayat perubahan perilaku agen
git diff HEAD~1 AGENTS.md     # apa yang berubah sejak kemarin
```

Riwayat git di workspace berubah menjadi riwayat perilaku agen.

Manfaat kedua yang sering tidak disadari: ketika agen mulai berperilaku aneh, git diff pada workspace sering menjawab pertanyaan lebih cepat daripada membaca log. Perubahan perilaku hampir selalu berasal dari perubahan berkas, dan berkas itu sekarang punya riwayat.

JANGAN LETAKKAN DI FOLDER TERSINKRONISASI
Bila folder ini ikut tersinkronisasi ke iCloud Drive atau Dropbox, Anda akan mendapatkan konflik berkas dan perilaku yang tidak dapat direproduksi. Simpan state di luar folder yang tersinkronisasi awan.

BAB 7 / PRAKTIKUM

Membangun workspace yang dapat ditelusuri

Praktikum ini mengubah workspace dari folder berkas menjadi catatan sejarah perilaku agen Anda. Pekerjaannya lima menit dan manfaatnya berlangsung bertahun-tahun.

Langkah 1 — periksa keadaan workspace

```
cd ~/.openclaw/workspace
ls -la
git status
```

Bila git terpasang, workspace yang benar-benar baru sudah diinisialisasi otomatis.

Langkah 2 — baca tiap berkas dan pahami perannya

Buka satu per satu dan baca isinya. Sebagian besar orang melewatkan langkah ini dan kemudian bertanya-tanya mengapa agen berperilaku tertentu — padahal jawabannya tertulis di berkas yang belum pernah mereka buka.

Berkas	Pertanyaan yang dijawabnya
AGENTS.md	Bagaimana agen harus bekerja
SOUL.md	Bagaimana agen harus terdengar
TOOLS.md	Apa yang perlu diketahui agen tentang tool-nya
IDENTITY.md	Siapa agen ini
USER.md	Siapa Anda, dan apa yang perlu diingat tentang Anda
HEARTBEAT.md	Apa yang diperiksa agen secara berkala
MEMORY.md	Apa yang harus diingat lintas sesi

Langkah 3 — buat komit pertama sebagai garis dasar

```
git add -A
git commit -m 'garis dasar: workspace awal sebelum penyesuaian'
```

Komit ini menjadi titik banding untuk seluruh perubahan perilaku berikutnya.

Langkah 4 — ubah satu hal dan amati

Sunting satu kalimat di SOUL.md, mulai sesi baru, dan perhatikan perbedaan nadanya. Lalu jalankan git diff untuk melihat persis apa yang Anda ubah. Latihan kecil ini membangun intuisi yang akan berguna ketika perilaku agen berubah tanpa Anda sengaja.

```
git diff HEAD -- SOUL.md
git log --oneline --stat -3
```

Riwayat git di workspace berubah menjadi riwayat perilaku agen.

BAB 7 / STUDI KASUS

Empat orang menyunting satu berkas

Sebuah tim berisi empat orang berbagi satu agen. Setiap orang menyunting AGENTS.md ketika menemukan perilaku yang ingin diperbaiki. Tidak ada yang memberi tahu yang lain, karena suntingannya kecil-kecil.

Setelah tiga bulan, AGENTS.md berisi instruksi yang saling bertentangan. Satu bagian meminta jawaban singkat, bagian lain meminta penjelasan rinci. Agen menjadi tidak konsisten, dan setiap orang menyalahkan modelnya.

Gejala	Penyebab sesungguhnya
Jawaban kadang panjang kadang pendek	Dua instruksi yang bertentangan di berkas yang sama
Perilaku berubah tanpa ada yang mengubah konfigurasi	Seseorang menyunting workspace, bukan konfigurasi
Tidak ada yang tahu instruksi mana yang benar	Tidak ada riwayat yang menjelaskan alasan tiap baris
Perbaikan satu orang membatalkan perbaikan yang lain	Tidak ada tinjauan sebelum perubahan berlaku

Perbaikan yang mereka pilih: workspace dijadikan repositori git privat, setiap perubahan pada AGENTS.md menuntut satu peninjau, dan pesan komit wajib menjelaskan perilaku apa yang sedang diperbaiki. Konflik berhenti dalam dua minggu.

> **INKONSISTENSI YANG UMUM**
> Perlakukan berkas workspace seperti kode, karena fungsinya memang seperti kode: ia menentukan perilaku sistem. Perusahaan yang menerima pull request untuk perubahan satu baris kode sering membiarkan siapa pun menyunting AGENTS.md tanpa tinjauan apa pun.

BAB 7 / LATIHAN

Latihan mandiri

1. Jadikan workspace Anda repositori git dan buat komit garis dasar. Bila ia sudah menjadi repositori, periksa apakah riwayatnya bermakna.
2. Baca seluruh berkas bootstrap dan tuliskan satu kalimat ringkasan untuk masing-masing dengan kata-kata Anda sendiri.
3. Ubah satu kalimat di SOUL.md, amati perubahan nadanya, lalu kembalikan dengan git. Catat berapa lama seluruh siklus itu memakan waktu.
4. Rancang aturan penyuntingan workspace untuk tim Anda: siapa yang boleh, apakah perlu tinjauan, dan bagaimana perubahan dikomunikasikan.

BAB 7 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
BOOTSTRAP.md tidak kembali setelah dihapus	Perilaku yang benar. Ia hanya dibuat untuk workspace yang benar-benar baru.
Subagen tidak mengenali kepribadian agen	Sesi subagen hanya memuat AGENTS.md dan TOOLS.md. Bukan kerusakan.
Agen tidak memuat ingatan yang Anda tulis	MEMORY.md tidak dibuat otomatis. Pastikan berkasnya benar-benar ada.
Perilaku agen berubah tanpa ada yang mengubah konfigurasi	Periksa git diff di workspace; hampir selalu ada berkas yang tersunting.
Konflik berkas aneh pada workspace	Folder berada di dalam direktori yang tersinkronisasi awan. Pindahkan.

Tiga pertanyaan review

1. Mengapa sesi subagen sengaja dibatasi hanya pada AGENTS.md dan TOOLS.md? Sebutkan satu manfaat biaya dan satu manfaat keamanan.
2. Agen Anda mulai memakai nada yang tidak pernah Anda minta. Berkas mana yang pertama Anda periksa, dan perintah apa yang paling cepat menjawabnya?
3. Rancang strategi cadangan workspace untuk tim beranggota empat orang yang semuanya menyunting AGENTS.md.

Kunci pembahasan

Subagen dibatasi karena kepribadian dan ingatan pribadi tidak diperlukan untuk tugas sempit; manfaat biayanya adalah prompt yang jauh lebih pendek, dan manfaat keamanannya adalah data pribadi pada USER.md dan MEMORY.md tidak ikut menyebar ke sesi yang mungkin memproses masukan tidak tepercaya. Nada yang berubah paling

mungkin berasal dari SOUL.md, dan git diff pada workspace adalah cara tercepat membuktikannya. Untuk tim empat orang, jadikan workspace repositori git privat dengan pull request wajib pada AGENTS.md, sehingga setiap perubahan perilaku punya peninjau dan dapat dikembalikan dengan satu perintah.

Rujukan: dokumentasi OpenClaw, halaman concepts/agent-workspace dan panduan asisten pribadi, ditinjau 17 September 2026.

FONDASI DAN INSTALASI

Bab 8 Runtime agen tertanam, sesi, dan penyimpanan SQLite

OpenClaw mengirimkan satu runtime agen tertanam: loop agen bawaan, penyambungan tool, dan perakitan prompt. Ini berbeda dari mendelegasikan giliran ke proses harness eksternal. Memahami batas itu menjelaskan mengapa sebagian perilaku tidak dapat diubah dari luar.

Tujuan belajar

Menjelaskan apa yang dimiliki runtime tertanam, menemukan tempat sesi disimpan, dan memahami perilaku steering ketika pesan baru tiba di tengah giliran yang sedang berjalan. Pada akhir bab, pembaca dapat menelusuri sebuah sesi sampai ke barisnya di basis data.

Apa yang dimiliki runtime

Runtime agen tertanam dimiliki OpenClaw sepenuhnya: penemuan model, penyambungan tool, perakitan prompt, manajemen sesi, dan pengiriman ke kanal berbagi satu permukaan runtime yang terpadu. Setiap agen yang terkonfigurasi punya workspace, berkas bootstrap, dan penyimpanan sesinya sendiri.

Gambar 8.1 — Enam tanggung jawab yang berbagi satu permukaan runtime. Keterpaduan inilah yang membuat perilakunya konsisten antar kanal.

BAB 8 / PENYIMPANAN

Di mana sesi benar-benar tersimpan

Baris sesi disimpan pada basis data SQLite per agen. Mengetahui jalurnya berguna untuk cadangan, diagnosis, dan untuk memahami apa yang sebenarnya hilang ketika seseorang menghapus sesuatu.

```
~/.openclaw/agents/<agentId>/agent/openclaw-agent.sqlite
~/.openclaw/agents/<agentId>/sessions/
```

Baris sesi ada di SQLite; folder sessions menampung berkas JSONL warisan.

Berkas transkrip JSONL masih dapat berada di bawah folder sessions sebagai masukan migrasi warisan, arsip yang dihapus atau direset, impor, ekspor, dan artefak dukungan. Riwayat agen yang aktif disimpan di SQLite bersama baris sesinya.

ID sesi bersifat stabil dan dipilih oleh OpenClaw. OpenClaw tidak membaca folder sesi milik alat lain. Bila Anda berharap memindahkan riwayat dari sistem agen lain dengan menyalin folder, harapan itu tidak akan terpenuhi — ada jalur impor tersendiri untuk itu.

SQLITE PER AGEN		FOLDER SESSIONS
Riwayat aktif	**Isinya**	JSONL warisan
Operasi harian	**Dipakai untuk**	Migrasi dan arsip
Wajib	**Disalin saat pindah**	Opsional
Tidak	**Dibaca dari alat lain**	Tidak

Gambar 8.2 — Dua tempat yang sering tertukar. Menyalin folder sessions saja tidak memindahkan riwayat yang aktif.

Perintah	Gunanya
`openclaw sessions`	Daftar sesi percakapan yang tersimpan
`openclaw sessions --agent work`	Satu penyimpanan agen yang terkonfigurasi
`openclaw sessions --all-agents`	Menggabungkan seluruh penyimpanan agen
`openclaw sessions --active 120`	Menyaring berdasarkan keaktifan
`openclaw sessions --json`	Keluaran yang dapat dibaca mesin

```
openclaw sessions export-trajectory \
  --session-key "agent:main:telegram:direct:123" \
  --workspace .
```

Mengekspor bundel trajektori untuk satu sesi tersimpan.

BAB 8 / STEERING

Pesan yang tiba di tengah giliran

Perilaku ini sering mengejutkan operator baru. Prompt masuk yang tiba di tengah sebuah giliran secara default diarahkan ke dalam giliran yang sedang berjalan. Agen tidak mengantre dan menjawab belakangan; ia menyesuaikan arah di tengah jalan.

Runtime memeriksa steering sebelum peluncuran tool yang belum dimulai dan sebelum panggilan model berikutnya. Tool yang sedang berjalan tetap dilanjutkan. Panggilan sekuensial yang belum dimulai akan dilewati, sementara panggilan paralel tetap berlanjut setelah batch-nya melewati titik pemeriksaan peluncuran.

Gambar 8.3 — Steering hanya terjadi pada titik periksa. Tool yang sudah berjalan tidak pernah dipotong di tengah.

PERIKSA SEBELUM MEMAKAI DI KANAL PELANGGAN

Bila Anda ingin perilaku antre, bukan steering, itu adalah setelan tersendiri. Jangan berasumsi default-nya sesuai dengan kebiasaan tim Anda — terutama pada kanal pelanggan, di mana pesan beruntun adalah hal biasa.

BAB 8 / PRAKTIKUM

Menelusuri satu sesi sampai ke barisnya

Praktikum ini mengubah sesi dari sesuatu yang abstrak menjadi baris yang dapat Anda lihat. Kemampuan ini akan terpakai setiap kali Anda mendiagnosis perilaku yang aneh.

Langkah 1 — daftar sesi yang ada

```
openclaw sessions
openclaw sessions --verbose
openclaw sessions --active 120
openclaw sessions --json | head -40
```

Opsi --active menyaring berdasarkan keaktifan dalam menit terakhir.

Langkah 2 — pahami bentuk kunci sesi

Bentuk kunci	Artinya
agent:<id>:main	Sesi utama agen; tidak dapat dikonfigurasi namanya
agent:<id>:telegram:group:<chatId>	Sesi grup Telegram yang terisolasi
...:topic:<threadId>	Akhiran topik forum untuk isolasi per topik
agent:<id>:whatsapp:group:<jid>	Sesi grup WhatsApp

Membaca kunci sesi adalah keterampilan yang terpakai di banyak bab berikutnya. Kunci itu memberi tahu Anda agen mana, kanal mana, dan ruang mana — tiga informasi yang menjawab sebagian besar pertanyaan diagnosis.

Langkah 3 — ekspor satu sesi untuk diperiksa

```
openclaw sessions export-trajectory \
  --session-key "agent:main:telegram:direct:123" \
  --workspace .
```

Bundel trajektori memperlihatkan urutan langkah yang benar-benar diambil agen.

Langkah 4 — amati steering secara langsung

Kirim satu pesan yang memicu pekerjaan agak panjang, lalu kirim pesan kedua sebelum agen selesai. Amati bahwa agen mengubah arah pada titik periksa, bukan mengantre. Lakukan ini sekali sekarang supaya perilakunya tidak mengejutkan Anda di kanal pelanggan nanti.

BAB 8 / STUDI KASUS

Tiga pesan beruntun dari satu pelanggan

Seorang reseller mengirim tiga pesan dalam sepuluh detik: nomor pesanan, lalu pertanyaan tentang stok, lalu keluhan tentang pengiriman terakhir. Kebiasaan yang sangat umum di WhatsApp.

Agen mengarahkan ketiganya ke dalam satu giliran yang sedang berjalan dan menjawab sekali dengan arah gabungan. Jawabannya menyinggung stok dan pengiriman, tetapi pertanyaan pertama tentang nomor pesanan terlewat — tertutup oleh dua pesan berikutnya.

Gambar 8.4 — Tiga pesan menjadi satu giliran. Efisien secara biaya, tetapi pesan pertama seakan terabaikan.

Reseller itu mengirim pesan keempat: “halo, nomor pesanan saya tadi bagaimana?” Dari sisi pelanggan, agen terlihat mengabaikan pertanyaan. Dari sisi sistem, agen bekerja persis seperti yang dirancang.

Pilihan penanganan	Konsekuensinya
Biarkan steering	Hemat token; risiko pertanyaan terlewat
Ubah ke mode antre	Tiap pesan dijawab; biaya lebih tinggi
Instruksi menjawab seluruh poin	Membantu, tetapi bukan jaminan
Gabungkan pesan dengan jeda singkat	Menunggu sebentar sebelum memproses

Tidak ada jawaban yang benar untuk semua kasus. Yang penting adalah memilih dengan sadar, bukan menemukan perilakunya dari keluhan pelanggan. Untuk kanal pelanggan, sebagian besar tim akhirnya memilih kombinasi baris ketiga dan keempat.

BAB 8 / LATIHAN

Latihan mandiri

1. Daftarkan seluruh sesi pada mesin Anda dan uraikan arti tiap kunci sesi yang muncul.
2. Ekspor satu bundel trajektori dan telusuri urutan langkah agen. Tandai titik di mana ia memanggil tool dan titik di mana ia memutuskan menjawab.
3. Picu steering dengan sengaja, lalu tuliskan apakah perilakunya sesuai dengan yang Anda inginkan untuk kanal yang akan Anda pakai.
4. Hitung berapa banyak riwayat yang akan hilang bila Anda hanya menyalin folder sessions tanpa basis data SQLite, dengan memeriksa isi keduanya.

BAB 8 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Riwayat hilang setelah menyalin folder sessions	Riwayat aktif ada di SQLite, bukan di JSONL. Pakai jalur impor resmi.
Agen berubah arah di tengah jawaban	Itu steering, dan ia default. Ubah mode antrean bila tidak diinginkan.
Tool tetap berjalan padahal sudah dibatalkan	Tool yang sudah dimulai tidak dipotong; hanya yang belum dimulai yang dilewati.
Sesi dari alat agen lain tidak terbaca	OpenClaw tidak membaca folder sesi alat lain. Ada perintah migrate tersendiri.
Ingin menelusuri satu sesi secara mendalam	Ekspor trajektorinya dengan sessions export-trajectory.

Tiga pertanyaan review

1. Mengapa tool yang sudah berjalan tidak dipotong ketika steering terjadi? Sebutkan satu risiko bila ia dipotong.
2. Seorang pelanggan mengirim tiga pesan beruntun dalam sepuluh detik. Jelaskan apa yang terjadi pada giliran agen, dan mengapa perilaku itu bisa menjadi masalah untuk layanan pelanggan.
3. Anda perlu memindahkan satu agen beserta riwayatnya ke Mac lain. Apa yang harus ikut, dan apa yang tidak cukup sekadar disalin?

Kunci pembahasan

Tool yang sudah berjalan tidak dipotong karena memotongnya dapat meninggalkan efek samping setengah jadi — berkas tertulis sebagian, panggilan API terkirim tanpa penanganan hasilnya. Tiga pesan beruntun akan diarahkan ke dalam giliran yang sedang berjalan, sehingga agen menjawab satu kali dengan arah gabungan; pada layanan

pelanggan ini bisa menjadi masalah karena pertanyaan pertama seakan terabaikan, dan di sanalah mode antrean layak dipertimbangkan. Untuk pindah mesin, yang harus ikut adalah workspace beserta riwayat git-nya, konfigurasi openclaw.json, dan basis data SQLite per agen; menyalin folder sessions saja tidak cukup karena riwayat aktif tidak berada di sana.

Rujukan: dokumentasi OpenClaw, halaman concepts/agent dan cli/sessions, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

Bab 9 Multi-agen dan pengikatan rute kanal

Satu Gateway dapat menjalankan beberapa agen sekaligus. Masing-masing punya workspace, berkas bootstrap, dan penyimpanan sesinya sendiri. Kemampuan ini menyelesaikan masalah yang muncul cepat di organisasi: agen untuk pekerjaan tidak boleh berbagi ingatan dengan agen untuk urusan pribadi, dan agen operasi tidak boleh berbagi kewenangan dengan agen riset.

Tujuan belajar

Membuat agen baru dengan workspace terpisah, mengikat lalu lintas kanal ke agen tertentu, dan memahami apa yang terjadi ketika sebuah pengikatan tidak cocok. Pada akhir bab, pembaca dapat memisahkan dua konteks kerja tanpa menjalankan dua Gateway.

Membuat dan melihat agen

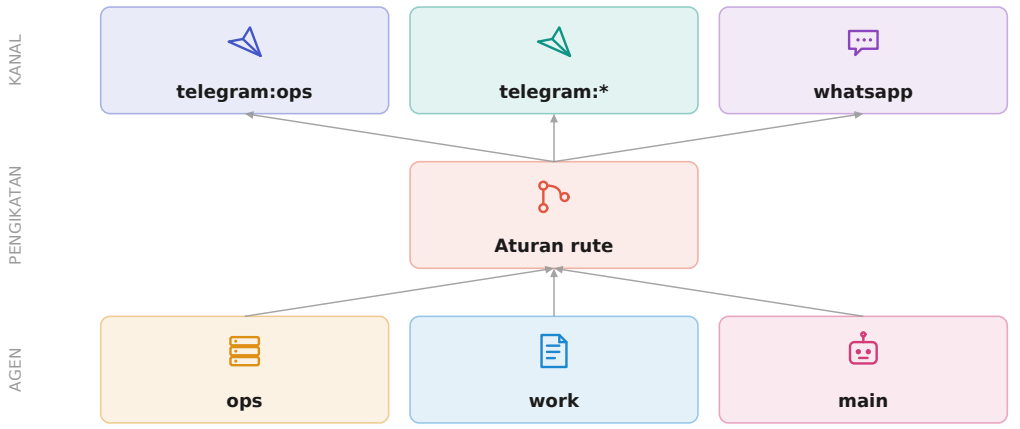
```
openclaw agents list
openclaw agents list --bindings

openclaw agents add work  --workspace ~/.openclaw/workspace-work
openclaw agents add ops   --workspace ~/.openclaw/workspace-ops \
                          --bind telegram:ops --non-interactive
openclaw agents add research --role researcher --non-interactive

openclaw agents delete work
```

Setiap agen memakai satu direktori workspace sebagai cwd untuk tool dan konteksnya.

Opsi --bindings pada perintah list menyertakan aturan rute selengkapnya, bukan hanya ringkasan jumlah per agen. Untuk daftar agen yang eksplisit, tanda default dan medan isDefault pada keluaran JSON mengikuti agents.defaults.systemAgent.agentId.



Gambar 9.1 — Lalu lintas masuk dipetakan ke agen lewat pengikatan. Tanpa pengikatan, semuanya jatuh ke agen default.

BAB 9 / PENGIKATAN

Mengunci lalu lintas ke agen tertentu

Pengikatan rute dipakai untuk mengunci lalu lintas kanal masuk ke agen tertentu. Tanpa itu, pesan mengikuti agen default, dan pemisahan yang Anda rancang tidak pernah benar-benar terjadi.

```
openclaw agents bindings
openclaw agents bindings --agent work
openclaw agents bindings --json

openclaw agents bind   --agent work --bind telegram:ops --bind discord:guild-a
openclaw agents unbind --agent work --bind telegram:ops
```

Menambah dan mencabut pengikatan. Beberapa --bind boleh digabung dalam satu perintah.

DUA PENGISIAN OTOMATIS YANG PERLU DISADARI

Bila Anda menghilangkan accountId dengan menulis --bind <channel> saja, OpenClaw menyelesaikannya dari default kanal dan hook penyiapan plugin bila tersedia. Bila Anda menghilangkan --agent pada bind atau unbind, OpenClaw menyasar agen default yang sedang aktif.

Baris kedua pada catatan itu adalah sumber kesalahan yang sering terjadi: seseorang menjalankan bind tanpa --agent sambil mengira sedang menyunting agen yang baru dibuatnya, padahal pengikatan itu menempel pada agen default.

Skill yang berbeda per agen

Bila Anda juga ingin skill yang terlihat berbeda untuk tiap agen, aturlah

agents.defaults.skills dan agents.list[].skills di dalam openclaw.json. Pemisahan kewenangan yang benar biasanya menggabungkan tiga hal sekaligus: workspace terpisah, pengikatan rute, dan daftar skill yang berbeda.

- Workspace terpisah per agen
- Pengikatan rute kanal ke agen yang benar
- Daftar skill dibatasi per agen
- Menganggap satu agen cukup untuk semua konteks

Gambar 9.2 — Tiga lapis pemisahan yang saling melengkapi. Melewatkan salah satunya membuat dua lainnya kehilangan sebagian artinya.

BAB 9 / IDENTITAS

Identitas dan avatar agen

```
openclaw agents set-identity --workspace ~/.openclaw/workspace --from-identity
openclaw agents set-identity --agent main --avatar avatars/openclaw.png

openclaw agents team create --non-interactive
```

Identitas agen dapat diambil dari berkas IDENTITY.md atau disetel langsung.

Memberi agen nama dan avatar yang berbeda bukan sekadar kosmetik. Pada kanal yang dipakai bersama tim, identitas visual adalah cara tercepat orang mengetahui kewenangan apa yang sedang mereka ajak bicara. Agen operasi yang tampil sama persis dengan agen riset adalah undangan untuk salah kirim perintah.

BAB 9 / PRAKTIKUM

Membangun agen kedua dari nol

Praktikum ini membuat agen terpisah untuk pekerjaan yang berbeda, lalu membuktikan bahwa keduanya benar-benar terpisah. Pembuktian itu bagian yang paling sering dilewati dan paling penting.

Langkah 1 — buat agen dengan workspace sendiri

```
openclaw agents list
openclaw agents add riset --workspace ~/.openclaw/workspace-riset
openclaw agents list --bindings
```

Perintah ketiga memperlihatkan aturan rute selengkapnya, bukan hanya jumlahnya.

Langkah 2 — buktikan workspace-nya terpisah

```
ls ~/.openclaw/workspace
ls ~/.openclaw/workspace-riset
diff <(ls ~/.openclaw/workspace) <(ls ~/.openclaw/workspace-riset)
```

Keduanya punya berkas bootstrap sendiri, sehingga kepribadian dan ingatannya tidak bercampur.

Langkah 3 — beri identitas yang berbeda

Sunting IDENTITY.md dan SOUL.md pada workspace baru supaya agen itu terdengar berbeda. Ini bukan kosmetik: pada kanal bersama, identitas visual adalah cara tercepat orang mengetahui kewenangan apa yang sedang mereka ajak bicara.

```
openclaw agents set-identity --agent riset --from-identity
```

Identitas dapat diambil dari berkas IDENTITY.md pada workspace agen tersebut.

Langkah 4 — uji pemisahannya

Uji	Cara	Hasil yang benar
Ingatan terpisah	Beri tahu agen A satu fakta, tanyakan ke agen B	Agen B tidak mengetahuinya
Skill terpisah	Setel skills berbeda per agen, minta keduanya	Masing-masing hanya tahu miliknya
Rute benar	Kirim pesan lewat kanal yang diikat	Agen yang dituju yang menjawab
Sesi terpisah	Jalankan openclaw sessions --all-agents	Dua penyimpanan berbeda terlihat

Bila salah satu uji gagal, jangan lanjutkan ke bab berikutnya sebelum menemukan penyebabnya. Pemisahan yang setengah jalan lebih berbahaya daripada tidak ada

pemisahan sama sekali, karena ia menciptakan rasa aman yang keliru.

BAB 9 / STUDI KASUS

Pengikatan yang menempel pada agen yang salah

Seorang operator membuat agen ops, lalu menjalankan perintah pengikatan untuk mengarahkan satu grup Telegram kepadanya. Ia lupa menyertakan penunjuk agen pada perintah itu.

Pengikatan menempel pada agen default. Grup operasi itu, yang seharusnya dilayani agen dengan kewenangan runbook, dilayani agen utama yang juga melayani percakapan pribadi operator — lengkap dengan ingatan dan skill-nya.

Yang tampak	Yang sebenarnya terjadi
Agen menjawab di grup	Agen yang salah yang menjawab
Perintah runbook bekerja	Karena agen default punya skill itu juga
Tidak ada galat	Pengikatan memang sah, hanya salah sasaran
Ingatan pribadi ikut terbawa	Workspace agen default yang dipakai

Yang membuat kasus ini sulit: semuanya berfungsi. Tidak ada kegagalan yang terlihat, hanya pemisahan yang tidak pernah benar-benar terjadi. Baru ketika agen menyebut sesuatu dari percakapan pribadi di dalam grup kerja, seseorang menyadarinya.

> **VERIFIKASI SETELAH SETIAP PENGIKATAN**
> Setelah setiap perintah bind atau unbind, jalankan openclaw agents bindings dan baca hasilnya. Dua detik verifikasi mencegah pemisahan semu yang dapat bertahan berbulan-bulan tanpa terdeteksi.

BAB 9 / LATIHAN

Latihan mandiri

1. Buat satu agen kedua dan jalankan keempat uji pemisahan pada praktikum langkah empat. Catat hasilnya.
2. Rancang pemisahan agen untuk organisasi Anda: berapa agen, apa peran masing-masing, dan kanal mana yang diikat ke mana.
3. Jalankan bind tanpa penunjuk agen dengan sengaja pada lingkungan uji, lalu temukan ke mana ia menempel. Ini melatih Anda mengenali gejalanya.
4. Tuliskan satu kalimat untuk setiap agen yang menjelaskan apa yang boleh dan tidak boleh ia lakukan. Simpan bersama catatan Bab 1.

BAB 9 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Pesan jatuh ke agen yang salah	Periksa openclaw agents bindings; kemungkinan tidak ada aturan yang cocok.
bind menempel pada agen yang tidak diinginkan	--agent dihilangkan, sehingga menyasar agen default yang sedang aktif.
Agen baru tidak punya kepribadian	Workspace baru punya berkas bootstrap sendiri. Isi SOUL.md dan IDENTITY.md-nya.
Skill yang sama muncul di semua agen	Atur agents.defaults.skills dan agents.list[].skills di openclaw.json.
Sulit membedakan agen di kanal tim	Beri nama dan avatar berbeda lewat agents set-identity.

Tiga pertanyaan review

1. Kapan memisahkan agen lebih tepat daripada memisahkan sesi? Sebutkan satu kriteria yang jelas.
2. Anda menjalankan bind tanpa --agent. Apa yang terjadi, dan bagaimana mendeteksinya sebelum menimbulkan masalah?
3. Rancang pemisahan untuk perusahaan yang punya satu agen layanan pelanggan dan satu agen operasi internal. Sebutkan tiga lapis pemisahannya.

Kunci pembahasan

Memisahkan agen tepat ketika dua konteks tidak boleh berbagi ingatan atau kewenangan; memisahkan sesi cukup ketika yang berbeda hanya topik pembicaraan. Menjalankan bind tanpa --agent membuat pengikatan menempel pada agen default, dan cara mendeteksinya adalah menjalankan openclaw agents bindings sesudahnya, bukan

mengandalkan ingatan. Untuk perusahaan dengan agen CS dan agen operasi, pisahkan workspace, ikat kanal pelanggan hanya ke agen CS, dan batasi daftar skill agen CS supaya ia tidak memiliki tool operasi sama sekali.

Rujukan: dokumentasi OpenClaw, halaman cli/agents dan concepts/multi-agent, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

Bab 10 Quick Chat, WebChat, dan Mac tab

Aplikasi macOS memberi tiga cara berbicara dengan agen yang sama, masing-masing dengan kelebihannya. Memilih yang tepat untuk tiap situasi terdengar sepele, tetapi ia menentukan apakah agen terasa seperti alat yang selalu siap atau aplikasi yang harus dibuka lebih dulu.

Tujuan belajar

Memanggil Quick Chat tanpa membuka jendela, memakai chat native beserta lampiran gambar, dan mengelola Mac tab di panel Browser. Pada akhir bab, pembaca dapat berpindah antara ketiganya tanpa kehilangan konteks.

Gambar 10.1 — Tiga permukaan, satu agen. Ketiganya berbagi sesi dan ingatan yang sama.

Quick Chat

Quick Chat adalah komposer bergaya Spotlight untuk sesi utama, tanpa perlu membuka jendela penuh. Tekan Option-Space secara default, pilih dari menu bar, atau rekam pintasan lain di Dashboard → Settings → This Mac → App.

```
⌥ Space          # Quick Chat
Cmd-,              # Dashboard settings
tombol hijau       # masuk layar penuh native
```

Pintasan yang paling sering dipakai sehari-hari.

Tombol hijau jendela membawa aplikasi ke layar penuh native. Sidebar Dasbor dan kontrol chat tetap tersedia di bagian atas jendela. Keluar dari layar penuh mengembalikan titlebar dan kontrol jendela seperti biasa.

BAB 10 / LAMPIRAN

Gambar masuk dan gambar keluar

Chat native penuh menerima lampiran gambar lewat pemilih berkas, tempel, dan seret lepas. Gambar yang dihasilkan asisten direnderer inline melalui URL artefak Gateway berumur pendek dan terbuka pada pratinjau yang lebih besar. iOS dan macOS berbagi model gambar terbatas dan perender yang sama.

> **JANGAN MENYIMPAN TAUTANNYA**
> URL artefak berumur pendek berarti tautan gambar tidak dapat dipakai sebagai penyimpanan permanen. Bila sebuah gambar penting, simpan berkasnya — jangan menyimpan tautannya.

BAB 10 / MAC TAB

Tautan web di dalam aplikasi

Di dasbor tertanam aplikasi macOS, mengeklik tautan web eksternal membukanya sebagai Mac tab pada tab Browser di panel samping chat. Pada rute non-chat, ia terbuka di dok Browser tingkat shell. Selagi Settings terbuka, tautan eksternal terbuka di peramban default karena panel Browser sedang tersembunyi.

WebKit merender Mac tab secara native, berdampingan dengan Agent browser tabs yang didukung peramban terkendali Gateway. Bundel Control UI yang lebih lama, yang masih mengirim permintaan tautan inline warisan, akan membuka peramban default.

Kontrol	Fungsinya
Tab strip	Memilih atau menutup halaman
Bilah URL	Menavigasi ke alamat lain
Back, forward, reload, stop	Mengelola Mac tab yang aktif
Open in Default Browser	Memindahkan halaman ke peramban sistem
Annotate dan Inspect	Menangkap cuplikan sekali jalan untuk konteks agen

Membuka tautan yang sama lagi akan memakai ulang tab yang sudah ada, termasuk mempertahankan URL asli setelah pengalihan awal. Mac tab milik masing-masing jendela dan bertahan melewati pergantian sesi chat. Menavigasi tab itu ke URL berbeda membuang cuplikan yang ditangkap dan mengembalikan tampilan langsungnya.

Klik kanan pada tautan

Klik kanan sebuah tautan eksternal di dasbor untuk memilih Open in Browser Panel, Open in Default Browser, atau Copy Link. Klik dengan tombol pengubah tetap membuka peramban default. Tautan jendela baru di dalam Mac tab membuka Mac tab lainnya, sementara unduhan yang dipicu penunjuk diserahkan ke peramban default.

BAB 10 / PRAKTIKUM

Menemukan permukaan yang cocok untuk tiap pekerjaan

Tiga permukaan berbicara dengan agen yang sama, tetapi terasa sangat berbeda saat dipakai. Praktikum ini membuat Anda mencoba ketiganya pada pekerjaan yang sama supaya pilihannya menjadi naluri, bukan pertimbangan.

Langkah 1 — satu pertanyaan, tiga permukaan

Permukaan	Cara memanggil	Rasakan
Quick Chat	Tekan Option-Space	Seberapa cepat dari niat ke jawaban
Chat native	Buka jendela penuh	Apa yang Anda dapat dari panel samping
Dasbor peramban	openclaw dashboard	Apa yang hilang tanpa This Mac

Ajukan pertanyaan yang sama di ketiganya dan perhatikan bukan jawabannya — jawabannya akan mirip — melainkan berapa banyak langkah yang Anda butuhkan untuk sampai ke sana.

Langkah 2 — ubah pintasan Quick Chat

Buka Dashboard → Settings → This Mac → App dan rekam pintasan lain bila Option-Space bentrok dengan aplikasi yang sudah Anda pakai. Pintasan yang bentrok adalah alasan paling umum orang berhenti memakai Quick Chat setelah minggu pertama.

Langkah 3 — cobalah lampiran gambar

Seret satu tangkapan layar ke jendela chat native, lalu minta agen menjelaskannya. Perhatikan bahwa gambar yang dihasilkan asisten dirender inline lewat URL artefak berumur pendek — simpan berkasnya bila Anda membutuhkannya besok.

Langkah 4 — pakai Mac tab dan Annotate

```
# Di dasbor tertanam:
1. Klik satu tautan web eksternal
2. Amati ia terbuka sebagai Mac tab di panel Browser
3. Tekan Annotate untuk menangkap cuplikan
4. Navigasikan tab itu ke URL lain dan amati cuplikan dibuang
```

Cuplikan dibuang saat tab dinavigasikan; itu perilaku yang disengaja.

BAB 10 / STUDI KASUS

Tautan gambar yang mati di dokumen rapat

Seorang analis meminta agen membuat bagan, lalu menyalin tautan gambarnya ke dalam dokumen rapat yang akan dibaca minggu depan. Tautan itu terlihat seperti tautan biasa dan berfungsi ketika ia mengujinya.

Pada hari rapat, gambar tidak muncul. URL artefak berumur pendek sudah kedaluwarsa. Dokumen yang sudah dibagikan ke dua belas orang memuat kotak kosong, dan bagannya harus dibuat ulang di tengah rapat.

Kebiasaan	Akibatnya
Menyalin tautan gambar	Mati dalam hitungan jam
Mengunduh berkasnya	Bertahan selama Anda menyimpannya
Menganggap tautan sama dengan berkas	Kesalahan yang terlihat baru kemudian
Menguji tautan segera setelah dibuat	Memberi rasa aman yang keliru

MENGAPA KESALAHAN INI BERULANG

Baris terakhir adalah yang membuat kesalahan ini sulit dihindari: pengujian langsung selalu berhasil, sehingga tidak ada sinyal bahwa ada yang salah sampai waktunya lewat.

BAB 10 / LATIHAN

Latihan mandiri

1. Ajukan satu pertanyaan yang sama di ketiga permukaan dan catat jumlah langkah dari niat sampai jawaban pada masing-masing.
2. Rekam pintasan Quick Chat yang tidak bentrok dengan aplikasi apa pun di Mac Anda, lalu pakai selama satu minggu.
3. Minta agen menghasilkan satu gambar, unduh berkasnya, lalu bandingkan dengan menyalin tautannya. Periksa keduanya lagi keesokan harinya.
4. Tuliskan panduan singkat untuk rekan kerja tentang kapan memakai Quick Chat dan kapan membuka jendela penuh.

BAB 10 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Option-Space tidak memanggil Quick Chat	Pintasan mungkin sudah dipakai aplikasi lain. Rekam ulang di This Mac → App.
Tautan selalu terbuka di Safari	Settings sedang terbuka, atau bundel Control UI masih memakai tautan inline warisan.
Gambar asisten hilang setelah beberapa waktu	URL artefak berumur pendek. Simpan berkasnya, bukan tautannya.
Cuplikan Annotate hilang sendiri	Tab dinavigasi ke URL lain; cuplikan dibuang dan tampilan langsung kembali.
Mac tab hilang saat ganti jendela	Mac tab milik masing-masing jendela, bukan milik aplikasi secara global.

Tiga pertanyaan review

1. Kapan Quick Chat lebih tepat daripada jendela chat penuh, dan kapan sebaliknya?
2. Mengapa gambar yang dihasilkan asisten memakai URL berumur pendek, dan apa konsekuensinya bagi alur kerja Anda?
3. Apa perbedaan Mac tab dengan Agent browser tab, dan mengapa perbedaan itu penting untuk keamanan?

Kunci pembahasan

Quick Chat unggul untuk satu pertanyaan cepat yang tidak memerlukan lampiran atau riwayat panjang, sedangkan jendela penuh diperlukan begitu Anda menyeret berkas atau ingin melihat panel samping. URL berumur pendek mengurangi risiko kebocoran gambar lewat tautan yang tersebar, dengan konsekuensi Anda harus menyimpan berkasnya sendiri bila gambar itu perlu bertahan. Mac tab dirender WebKit dan Anda yang

mengendalikannya, sementara Agent browser tab dikendalikan Gateway atas nama agen; membedakan keduanya penting karena hanya yang kedua yang dapat dipengaruhi oleh isi halaman yang tidak tepercaya.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos bagian Quick Chat dan Open dashboard links, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

Bab 11 Impor login browser dan sinkronisasi cookie

Agen yang bisa membuka peramban tetapi tidak pernah masuk ke situs mana pun hanya berguna setengah. OpenClaw menyediakan dua mekanisme untuk menutup celah itu, dan keduanya menyentuh kredensial — jadi keduanya layak dipahami sebelum dinyalakan.

Tujuan belajar

Mengimpor cookie dari peramban keluarga Chrome ke profil terkelola, menyalakan sinkronisasi cookie ke Gateway di mesin lain, dan mengetahui batas keduanya. Pada akhir bab, pembaca dapat memberi agen akses masuk tanpa menyerahkan kata sandi.

IMPOR COOKIE		SINKRONISASI COOKIE
Profil di Mac ini	**Tujuan**	Peramban di mesin lain
Sekali	**Frekuensi**	Berjalan terus
Local	**Mode**	Remote
Tidak	**Perlu CLI eksternal**	Ya
Ditawarkan lewat spanduk	**Default**	Mati
Seluruh cookie profil	**Cakupan**	Hanya domain di allowlist

Gambar 11.1 — Dua mekanisme yang sering tertukar. Yang kanan hanya masuk akal ketika peramban agen berada di komputer yang berbeda.

BAB 11 / IMPOR

Impor sekali jalan

Pertama kali sebuah Mac tab terbuka sementara aplikasi berjalan terhadap Gateway lokal, dasbor menampilkan spanduk yang dapat ditutup bila ada profil keluarga Chrome yang memiliki cookie di Mac itu. Spanduk menawarkan menyalin cookie tersebut ke profil terkelola yang terisolasi, yang dipakai agen untuk menjelajah.

Pilih sebuah profil dari kontrol Import miliknya — Touch ID mungkin diminta. Progres dan jumlah cookie yang terimpor muncul inline. Hanya cookie yang disalin; kata sandi tidak pernah meninggalkan peramban sumber.

> **MENUTUP SPANDUK BUKAN KEPUTUSAN PERMANEN**
> Menutup spanduk merekam pilihan itu. Dashboard → Settings → This Mac → Browser dapat membuka kembali alur impor native selama Gateway lokal dan profil yang memenuhi syarat tersedia.

Berpindah dari mode Local menyembunyikan spanduk impor dan membuang status atau hasil spanduk yang tertunda. Impor yang sudah terkirim ke Gateway lokal mungkin tetap selesai di sana; berpindah mode tidak membatalkan cookie yang sudah tersalin. Kembali ke mode Local memungkinkan Anda meminta tawaran baru dari Settings.

BAB 11 / SINKRONISASI

Menjaga peramban di komputer lain tetap masuk

Impor menyalin cookie satu kali ke profil di Mac yang sama. Ketika Gateway dan peramban agen berjalan di komputer terpisah — kotak khusus, host Linux tanpa layar, atau kontainer awan — nyalakan sinkronisasi cookie supaya Mac ini menjaga peramban jarak jauh itu tetap masuk ke situs yang Anda pilih.

Gambar 11.2 — Empat langkah. Langkah kedua adalah pengamannya: allowlist kosong tidak menyinkronkan apa pun.

Buka Dashboard → Settings → This Mac → Browser. Sinkronisasi cookie mati secara default dan hanya tersedia pada mode remote dengan CLI eksternal di Mac ini. Nyalakan, tambahkan situs yang ingin dijaga ke allowlist Domains — misalnya github.com dan accounts.google.com — lalu tetapkan Target profile yang menerimanya, yaitu nama profil terkelola pada Gateway jarak jauh, dengan nilai bawaan imported. Satu baris status menunjukkan apakah sinkronisasi sedang berjalan.

```
openclaw browser cookie-sync --watch
```

Perintah yang disupervisi aplikasi selama sinkronisasi menyala.

Cookie didekripsi secara lokal di Mac ini — satu permintaan Keychain macOS atau Touch ID per sesi — lalu didorong ke profil jarak jauh melalui sambungan Gateway terenkripsi yang sudah ada. Hanya domain pada allowlist yang pernah dikirim, dan nilai cookie tidak pernah ditulis ke log. Allowlist kosong tidak menyinkronkan apa pun.

BATAS YANG TIDAK BISA DILEWATI

Sebagian sesi Google memakai kredensial sesi terikat perangkat yang tetap melekat pada Mac ini dan mungkin masih menuntut autentikasi ulang setelah sinkronisasi. Untuk situs seperti itu, kendalikan peramban di Mac itu sendiri lewat proxy node peramban.

Bila sebuah penambahan yang tertunda akan memulihkan domain yang dihapus di jendela Dashboard lain, penambahan itu dibuang. Tinjau daftar yang sudah diperbarui lalu tambahkan kembali domain yang Anda maksud.

BAB 11 / PRAKTIKUM

Memberi agen akses masuk dengan aman

Praktikum ini memberi agen kemampuan menjelajah situs yang menuntut login, tanpa menyerahkan kata sandi apa pun. Kerjakan langkahnya berurutan dan berhenti di langkah mana pun yang terasa membuka terlalu banyak.

Langkah 1 — tentukan situs apa yang benar-benar dibutuhkan

Tuliskan daftarnya sebelum membuka menu apa pun. Daftar yang disusun setelah melihat pilihan yang tersedia selalu lebih panjang daripada yang disusun dari kebutuhan.

```
# Contoh isian
Butuh   : github.com        (membaca isu repositori internal)
Butuh   : docs.internal     (membaca dokumentasi perusahaan)
Tidak   : accounts.google.com, email, perbankan, media sosial
```

Kolom kedua adalah alasan; entri tanpa alasan sebaiknya masuk baris ketiga.

Langkah 2 — impor pada mode lokal

Pastikan aplikasi berjalan terhadap Gateway lokal, buka satu Mac tab, lalu amati spanduk impor. Pilih profil dari kontrol Import; Touch ID mungkin diminta. Progres dan jumlah cookie yang terimpor muncul inline.

Langkah 3 — verifikasi apa yang terjadi

Periksa	Hasil yang benar
Jumlah cookie terimpor	Sesuai perkiraan; angka sangat besar layak dicurigai
Profil tujuan	Profil terkelola yang terisolasi, bukan profil pribadi
Kata sandi	Tidak ikut; hanya cookie yang disalin
Peramban sumber	Tidak berubah

Langkah 4 — bila Gateway ada di mesin lain

```
# Dashboard → Settings → This Mac → Browser
1. Nyalakan cookie sync (hanya tersedia pada mode remote)
2. Isi allowlist Domains dengan situs dari langkah 1
3. Tetapkan Target profile pada Gateway jarak jauh
4. Periksa baris status: apakah sinkronisasi berjalan
```

Allowlist kosong menyinkronkan nol domain — itu perilaku yang benar.

BAB 11 / STUDI KASUS

Allowlist yang tumbuh tanpa ditinjau

Sebuah tim menyalakan sinkronisasi cookie untuk dua domain. Selama enam bulan, setiap kali agen gagal mengakses sesuatu, seseorang menambahkan satu domain lagi supaya pekerjaan berlanjut. Tidak ada yang mencatat alasannya.

Ketika audit keamanan tahunan datang, allowlist berisi tujuh belas domain. Tidak ada yang dapat menjelaskan sebelas di antaranya. Tiga domain adalah layanan yang sudah tidak dipakai perusahaan, dan satu adalah penyedia surel.

PENAMBAHAN CEPAT		PENAMBAHAN TERCATAT
Tiga puluh detik	**Waktu saat itu**	Lima menit
Tidak	**Alasan tercatat**	Ya
Tidak	**Dapat ditinjau**	Ya
Tidak ada yang tahu	**Enam bulan kemudian**	Dapat dihapus dengan yakin
Semua dipertahankan	**Saat audit**	Yang usang dibuang

Gambar 11.3 — Selisih empat menit saat penambahan menjadi selisih berjam-jam saat audit.

Pola ini berlaku jauh di luar allowlist cookie. Setiap daftar izin di buku ini — nomor WhatsApp, anggota grup, perintah node, domain — tumbuh dengan cara yang sama dan membusuk dengan cara yang sama.

> **ATURAN YANG MENCEGAH PEMBUSUKAN DAFTAR IZIN**
> Tetapkan aturan sederhana: setiap entri pada daftar izin apa pun harus punya alasan tertulis dan tanggal. Entri tanpa alasan dihapus pada tinjauan berikutnya, dan yang benar-benar dibutuhkan akan ditambahkan kembali dengan cepat.

BAB 11 / LATIHAN

Latihan mandiri

1. Susun daftar situs yang benar-benar dibutuhkan agen Anda, lengkap dengan alasan tiap entri, sebelum menyalakan impor apa pun.
2. Jalankan impor cookie pada mode lokal dan verifikasi keempat butir pada praktikum langkah tiga.
3. Bila Gateway Anda jarak jauh, nyalakan sinkronisasi dengan satu domain saja, lalu buktikan domain lain tidak ikut terkirim.
4. Tetapkan jadwal tinjauan allowlist dan tuliskan siapa yang bertanggung jawab menjalankannya.

BAB 11 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Spanduk impor tidak pernah muncul	Butuh mode Local, Mac tab yang terbuka, dan profil Chrome yang punya cookie.
Sinkronisasi tidak tersedia	Ia hanya ada pada mode remote dengan CLI eksternal terpasang di Mac ini.
Tidak ada cookie yang terkirim	Allowlist domain kosong. Allowlist kosong menyinkronkan nol domain.
Google tetap meminta login ulang	Sesi terikat perangkat. Pakai proxy node peramban untuk situs semacam itu.
Domain yang ditambahkan hilang lagi	Ada penghapusan di jendela Dashboard lain. Tinjau daftar, tambahkan ulang.

Tiga pertanyaan review

1. Mengapa hanya cookie yang disalin dan bukan kata sandi? Sebutkan konsekuensinya bila kata sandi ikut disalin.
2. Anda memakai Gateway di host Linux tanpa layar. Mekanisme mana yang Anda butuhkan, dan mengapa yang satunya tidak berguna di sana?
3. Rancang allowlist domain untuk agen yang tugasnya hanya membaca isu GitHub milik perusahaan. Domain apa saja yang pantas masuk, dan apa yang sengaja Anda tinggalkan?

Kunci pembahasan

Hanya cookie yang disalin karena cookie dapat dicabut per sesi sementara kata sandi membuka seluruh akun tanpa batas waktu; menyalin kata sandi akan mengubah satu kompromi profil peramban menjadi kompromi identitas penuh. Untuk Gateway di host

Linux tanpa layar, yang dibutuhkan adalah sinkronisasi cookie, karena impor hanya menyalin ke profil pada Mac yang sama dan peramban agen tidak berada di sana. Allowlist untuk agen pembaca isu cukup berisi github.com; domain autentikasi pihak ketiga dan layanan lain sengaja ditinggalkan supaya kompromi pada agen itu tidak membuka apa pun di luar cakupannya.

Rujukan: dokumentasi OpenClaw, halaman platforms/macos bagian Import browser logins dan Sync cookies, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

Bab 12 Node Mac: kamera, layar, lokasi, dan kontrol komputer

Node adalah perangkat pendamping yang menyambung ke WebSocket Gateway dengan peran node dan memaparkan permukaan perintah. Mac Anda dapat menjadi salah satunya. Di sinilah OpenClaw berhenti menjadi program percakapan dan mulai menjadi sesuatu yang dapat melihat dan menyentuh dunia.

Tujuan belajar

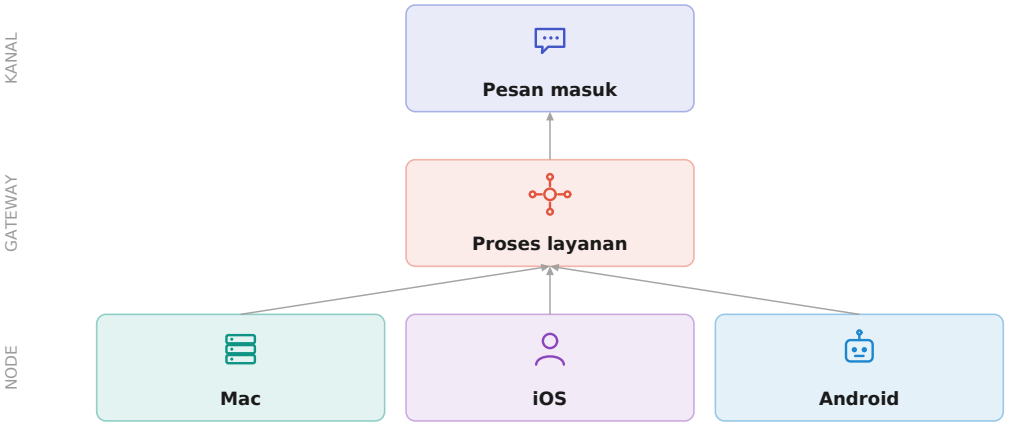
Memasangkan node, membaca permukaan perintahnya, dan memahami lima lapis izin yang harus sepakat sebelum sebuah perintah berbahaya berjalan. Pada akhir bab, pembaca dapat memutuskan kapabilitas mana yang pantas dinyalakan dan mana yang tidak.

Mac sebagai node

macOS dapat berjalan dalam mode node: aplikasi menu bar menyambung ke server WS Gateway dan memaparkan perintah lokalnya sebagai sebuah node, sehingga openclaw nodes bekerja terhadap Mac ini. Aplikasi menambahkan perintah panel widget native, kamera, layar, notifikasi, dan kontrol komputer ke permukaan perintah node-host yang sama dengan yang dipakai openclaw node run.

> **SATU MAC, SATU NODE**
> Jangan menjalankan node CLI kedua pada Mac itu. Aplikasi menjalankan runtime node-host CLI yang cocok sebagai worker internal dan tetap menjadi satu-satunya sambungan Gateway serta identitas node.

Node adalah periferal, bukan gerbang. Ia tidak menjalankan layanan Gateway. Pesan Telegram, WhatsApp, dan kanal lain mendarat di Gateway, bukan di node. Kekeliruan membaca hubungan ini menghasilkan diagnosis yang salah arah selama berjam-jam.



Gambar 12.1 — Node menggantung di bawah Gateway sebagai periferal. Tidak ada pesan kanal yang pernah mendarat langsung pada node.

BAB 12 / PEMASANGAN

Pemasangan perangkat dan dua penyimpanan yang berbeda

Node WS memakai pemasangan perangkat. Node menyajikan identitas perangkat saat connect, dan Gateway membuat permintaan pemasangan perangkat untuk peran node. Setujui lewat CLI devices atau lewat antarmuka.

```
openclaw devices list
openclaw devices approve <requestId>
openclaw devices reject  <requestId>

openclaw nodes status
openclaw nodes describe --node <idOrNameOrIp>
openclaw nodes pending
```

Dua kelompok perintah yang mudah tertukar: devices dan nodes.

DUA PENYIMPANAN, DUA FUNGSI

node.pair.* — dengan CLI openclaw nodes pending, approve, dan reject — adalah penyimpanan pemasangan node terpisah yang dimiliki Gateway. Ia tidak menggerbangi handshake connect WS. Yang menggerbangi handshake adalah pemasangan perangkat lewat devices.

Persetujuan otomatis dari jaringan tepercaya

```
{
  gateway: {
    nodes: {
```

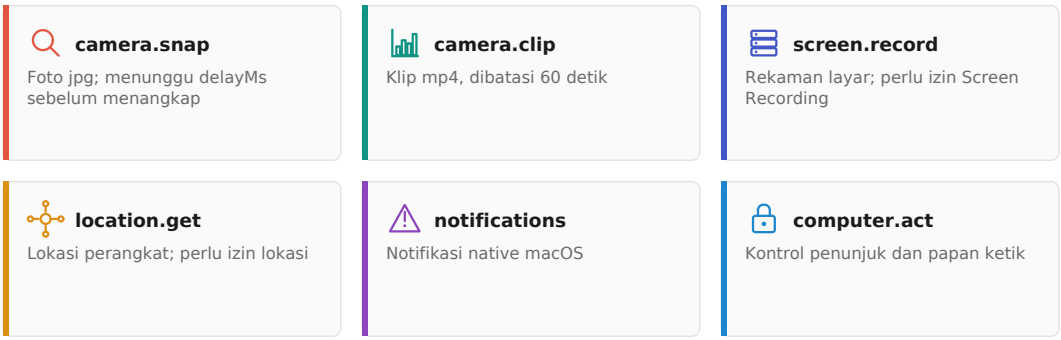
```
      pairing: {
        autoApproveCidrs: ["192.168.1.0/24"],
      },
      allowCommands: ["camera.snap", "screen.record"],
      denyCommands: ["camera.clip"],
    },
  },
}
```

autoApproveCidrs mati bila tidak disetel, dan hanya berlaku untuk permintaan role:node pertama kali tanpa scope yang diminta; ia tidak menyetujui peningkatan.

denyCommands memblokir nama perintah persis meskipun default atau allowCommands menyertakannya. Urutan ini penting: penolakan selalu menang, sehingga Anda dapat membuka satu keluarga perintah lalu mengunci satu anggotanya yang paling berisiko.

BAB 12 / KAPABILITAS

Apa yang dapat dilakukan node Mac



Gambar 12.2 — Enam keluarga perintah node Mac. Kotak terakhir berada pada kelas risiko tersendiri dan dibahas di halaman berikutnya.

Pada macOS, camera.snap menunggu delayMs — bawaannya 2000 milidetik — setelah pemanasan dan penyetelan eksposur sebelum menangkap. Muatan foto dikompresi ulang supaya base64-nya tetap di bawah 5 MB. Klip video dibatasi 60 detik untuk menghindari muatan node yang terlalu besar.

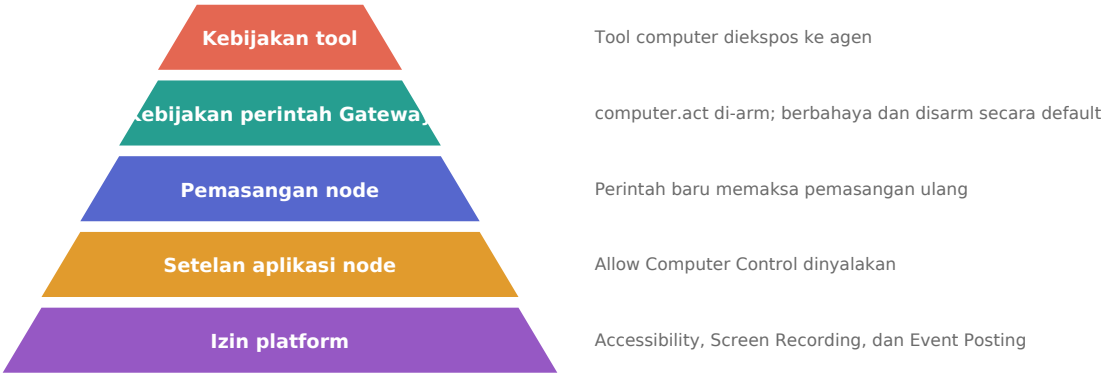
Kapabilitas	Izin macOS	Kode kegagalan umum
camera.snap dan camera.clip	Kamera, plus mikrofon untuk audio klip	*_PERMISSION_REQUIRED
screen.record	Screen Recording	*_PERMISSION_REQUIRED
location.get	Izin lokasi	LOCATION_PERMISSION_REQUIRED

Kapabilitas	Izin macOS	Kode kegagalan umum
computer.act	Accessibility dan Screen Recording	Ditolak sebelum dispatch

BAB 12 / KONTROL KOMPUTER

Lima lapis yang harus sepakat

Computer use memungkinkan agen Gateway melihat dan mengendalikan desktop Mac yang terpasang: ia menangkap tangkapan layar lewat perintah node screen.snapshot dan menggerakkan penunjuk serta papan ketik lewat satu perintah node berbahaya, computer.act. Set aksinya mencerminkan tool computer Anthropic, sehingga model apa pun yang mampu melihat dapat menggerakkannya.



Gambar 12.3 — Lima lapis izin. Semuanya harus sepakat; satu lapis yang menolak sudah cukup untuk menghentikan seluruh aksi.

Aksi dijalankan selama kontrol tahan lama itu tetap menyala; tidak ada konfirmasi per aksi. Fulfiller macOS mengirim teks satu grafem pada satu waktu, sehingga pembatalan, pemutusan, jeda, penonaktifan, atau penggantian endpoint menghentikannya sebelum grafem berikutnya.

> **DESKTOP HARUS TERBUKA**
> Pada macOS, penangkapan dan masukan menuntut desktop yang terbuka dan terverifikasi. Keep-awake sementara menutupi satu eksekusi Computer yang aktif, termasuk aksi latar, paling lama satu jam. Penguncian manual, logout, atau keadaan yang tidak diketahui melepaskan asersi dan memensiunkan eksekusi itu.

Pada macOS, Dashboard → Settings → This Mac → Capabilities memilih penyedia lokal node: Peekaboo adalah bawaannya dan mempertahankan jalur aksi berkoordinat dalam proses yang sudah ada, sementara CUA memakai daemon driver yang tertanam di dalam OpenClaw.app.

BAB 12 / PRAKTIKUM

Memasangkan node dan menguji batasnya

Praktikum ini memasangkan Mac Anda sebagai node, lalu menguji setiap gerbang izin satu per satu. Menguji gerbang ketika semuanya tenang jauh lebih baik daripada menemukannya saat sesuatu mendesak.

Langkah 1 — pasangkan dan periksa

```
openclaw devices list
openclaw devices approve <requestId>

openclaw nodes status
openclaw nodes describe --node <idOrNameOrIp>
```

Perintah terakhir memperlihatkan kapabilitas apa yang benar-benar tersedia.

Langkah 2 — uji kapabilitas yang aman lebih dulu

Uji	Yang Anda pelajari
Panggil kapabilitas tanpa memberi izin TCC	Bentuk pesan galat izin dan cara mengenalinya
Berikan izin lalu panggil ulang	Bahwa izin diberikan per konteks proses
Cabut izin lalu panggil lagi	Bahwa pencabutan berlaku seketika
Tambahkan perintah ke denyCommands	Bahwa penolakan menang atas allowCommands

Langkah 3 — susun kebijakan perintah node

```
{
  gateway: {
    nodes: {
      allowCommands: ["screen.snapshot"],
      denyCommands: ["camera.clip", "screen.record"],
    },
  },
}
```

Mulai dari daftar izin yang sangat sempit, lalu perluas hanya bila ada kebutuhan yang dapat dinamai.

Langkah 4 — jangan arm computer.act dulu

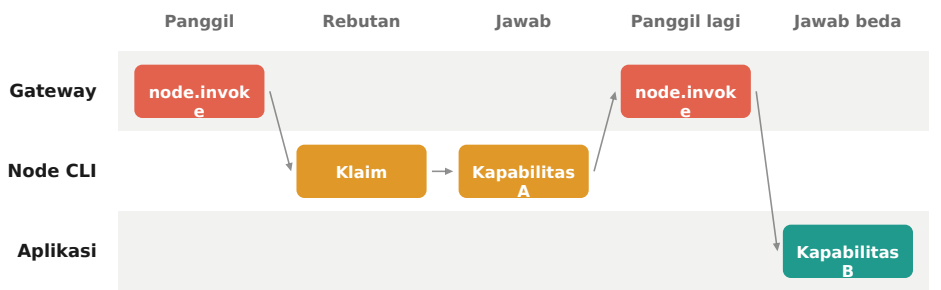
Biarkan kontrol komputer tetap disarm sampai Anda benar-benar membutuhkannya dan sudah membaca Bab 51. Kemampuan menggerakkan penunjuk dan papan ketik pada Mac yang terbuka adalah kapabilitas dengan konsekuensi terbesar di seluruh buku ini.

BAB 12 / STUDI KASUS

Node kedua yang tidak seharusnya ada

Seorang insinyur memasang aplikasi macOS pada Mac yang sudah menjalankan node CLI dari percobaan sebelumnya. Keduanya berjalan bersamaan selama beberapa minggu.

Gejalanya membingungkan: perintah node kadang berhasil kadang gagal, kapabilitas yang terdaftar berubah-ubah antar pemeriksaan, dan satu Mac muncul dua kali pada daftar perangkat dengan nama yang mirip. Setiap kali seseorang memeriksa, hasilnya berbeda.



Gambar 12.4 — Dua node dengan identitas berbeda pada satu mesin menghasilkan jawaban yang tidak dapat diprediksi.

Penyelesaiannya satu baris: hentikan node CLI. Aplikasi menjalankan runtime node-host CLI yang cocok sebagai worker internal dan tetap menjadi satu-satunya sambungan Gateway serta identitas node untuk Mac itu.

GEJALA YANG LANGSUNG MENUNJUK PENYEBAB
Bila satu Mac muncul lebih dari sekali pada daftar node atau perangkat, itu hampir selalu berarti ada proses kedua yang berjalan. Periksa sebelum mendiagnosis hal lain.

BAB 12 / LATIHAN

Latihan mandiri

1. Pasangkan Mac Anda sebagai node dan jalankan nodes describe. Daftarkan setiap kapabilitas yang muncul beserta izin macOS yang dibutuhkannya.
2. Jalankan keempat uji pada praktikum langkah dua dan catat bentuk pesan galat untuk masing-masing. Ini akan mempercepat diagnosis Anda nanti.
3. Susun kebijakan allowCommands dan denyCommands paling sempit yang masih memenuhi kebutuhan nyata Anda hari ini.
4. Tuliskan syarat apa yang harus terpenuhi sebelum Anda bersedia meng-arm computer.act, lalu simpan sampai Bab 51 selesai dibaca.

BAB 12 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Perintah node gagal padahal node terlihat di status	Jalankan nodes describe; pastikan kapabilitas yang dipanggil memang ada di sana.
*_PERMISSION_REQUIRED pada kamera atau layar	Izin TCC belum diberikan. Buka This Mac → Permissions.
computer.act ditolak terus	Ia berbahaya dan disarm secara default. Periksa kelima lapis pada Gambar 12.3.
Dua node muncul untuk satu Mac	Ada node CLI kedua yang berjalan. Hentikan; aplikasi sudah menjalankan node-host.
Prompt pemasangan tidak pernah muncul	Jalankan openclaw nodes pending dan setujui secara manual.

Tiga pertanyaan review

1. Mengapa computer.act dirancang disarm secara default, padahal semua izin macOS-nya sudah bisa diminta terpisah?
2. Jelaskan perbedaan penyimpanan pemasangan perangkat dan pemasangan node, serta mana yang menggerbangi handshake WS.
3. Anda ingin agen dapat memotret layar tetapi tidak pernah merekam video. Tuliskan konfigurasi gateway.nodes yang mencapainya, dan jelaskan mengapa urutan allow dan deny penting.

Kunci pembahasan

computer.act disarm secara default karena izin macOS hanya membuktikan bahwa sistem operasi mengizinkan, bukan bahwa Anda menginginkannya; lapis arm pada Gateway adalah tempat keinginan itu dinyatakan secara eksplisit. Pemasangan perangkat lewat

devices menggerbangi handshake connect WS, sedangkan node.pair.* adalah penyimpanan terpisah milik Gateway yang mengatur pemaparan perintah dan tidak menyentuh handshake. Untuk memotret tanpa merekam, cantumkan camera.snap pada allowCommands dan camera.clip beserta screen.record pada denyCommands; urutannya penting karena penolakan selalu menang, sehingga satu entri deny tetap mengunci perintah yang secara bawaan terbuka.

Rujukan: dokumentasi OpenClaw, halaman nodes, nodes/camera, nodes/computer-use, dan nodes/troubleshooting, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

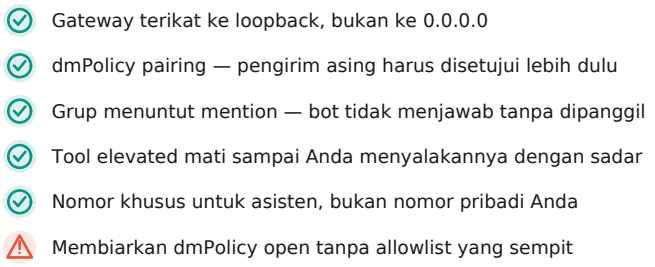
Bab 13 Sikap aman sejak hari pertama

Anda menempatkan agen pada posisi yang dapat menjalankan perintah di mesin Anda, membaca dan menulis berkas di workspace Anda, serta mengirim pesan keluar lewat kanal yang terpasang. Dokumentasi resmi menyebut ini di bagian paling atas panduan asisten pribadinya, sebelum satu pun perintah instalasi. Bab ini mengikuti urutan yang sama.

Tujuan belajar

Menetapkan lantai keamanan yang benar sebelum kanal pertama dibuka, memahami kebijakan DM dan grup, dan menjalankan audit keamanan bawaan. Pada akhir bab, pembaca memiliki konfigurasi yang layak disebut aman untuk dipakai sendiri.

Lantai keamanan



Gambar 13.1 — Lantai keamanan. Lima yang pertama adalah tempat Anda memulai; yang terakhir adalah kesalahan paling mahal yang masih sering terjadi.

Port kendali WebSocket terikat ke 127.0.0.1:18789. Akses jarak jauh menuntut Tailscale atau terowongan SSH dengan autentikasi kata sandi atau token. Mengikat Gateway ke 0.0.0.0 pada jaringan yang tidak tepercaya adalah kesalahan konfigurasi paling berbahaya yang dapat Anda buat — ia memberi jalur langsung ke agen Anda.

BAB 13 / KEBIJAKAN DM

Siapa yang boleh berbicara dengan agen

Pertahanan pertama OpenClaw adalah kebijakan DM. Ia mengendalikan DM masuk sebelum pesan menyentuh agen Anda. Ketika sebuah kanal dikonfigurasi dengan kebijakan DM pairing, pengirim yang tidak dikenal menerima kode pendek dan pesannya tidak diproses sampai Anda menyetujuinya.

Kebijakan	Perilaku	Pantas untuk
pairing	Pengirim asing dapat kode, harus disetujui	Pemakaian pribadi; ini bawaannya
allowlist	Hanya entri yang terdaftar yang dilayani	Layanan dengan daftar pelanggan yang tetap
open	Publik, dan hanya publik bila allowlist memuat *	Nyaris tidak pernah; butuh alasan yang sangat kuat

OPEN TIDAK SELALU BERARTI PUBLIK

dmPolicy open baru benar-benar publik ketika daftar izin DM efektif memuat tanda bintang. Penyiapan dan validasi menuntut wildcard itu untuk konfigurasi yang memang publik. Bila keadaan lama berisi open dengan entri allowFrom yang konkret, runtime tetap melayani entri tersebut.

Persetujuan disimpan ke berkas kredensial per kanal di bawah ~/.openclaw/credentials/. Jangan membagikan kode pemasangan Anda kepada siapa pun; perlakukan ia seperti kata sandi sekali pakai. Permintaan yang tertunda dibatasi — secara bawaan tiga per kanal — sehingga seseorang yang membanjiri bot Anda tidak akan memenuhi antrean.

Gambar 13.2 — Alur pairing. Pesan pertama tidak pernah mencapai model; ia hanya memicu penerbitan kode.

BAB 13 / GRUP DAN KONTEKS

Grup, mention, dan apa yang terlihat model

Grup adalah persoalan yang berbeda. Otorisasi pemicu — siapa yang boleh memicu agen — diatur lewat dmPolicy, groupPolicy, allowlist, dan gerbang mention. Visibilitas konteks adalah persoalan terpisah: apa saja konteks tambahan yang sampai ke model, termasuk badan balasan, teks yang dikutip, riwayat thread, dan metadata pesan yang diteruskan.

Membedakan keduanya penting ketika membaca laporan keamanan. Laporan yang hanya menunjukkan bahwa model dapat melihat teks kutipan atau riwayat dari pengirim yang tidak ada di allowlist adalah temuan pengerasan yang dapat ditangani dengan contextVisibility — bukan dengan sendirinya sebuah pelanggaran otorisasi atau sandbox.

Perintah dan direktif

Slash command dan direktif hanya dihormati untuk pengirim yang terotorisasi. Konfigurasikan daftar commands.allowFrom eksplisit per penyedia, atau biarkan otorisasi perintah mengikuti allowlist kanal dan keadaan pemasangan. Entri access group yang dirujuk allowlist kanal diselesaikan secara otomatis; tidak ada tombol opt-in untuk itu.

BAB 13 / AUDIT

Menjalankan audit keamanan

```
openclaw security audit
openclaw security audit --deep
openclaw security audit --fix

openclaw security audit --json | jq '.summary'
openclaw security audit --deep --json \
  | jq '.findings[] | select(.severity=="critical") | .checkId'
```

Audit bawaan; --fix menerapkan perbaikan mekanis dan melaporkan hasilnya.

Di antara yang dikerjakan --fix: membalik groupPolicy open yang umum menjadi allowlist, menyemai groupAllowFrom dari berkas allowFrom tersimpan ketika daftar itu ada dan konfigurasi belum mendefinisikannya, serta memperketat izin berkas untuk state dan konfigurasi beserta berkas sensitif yang umum — termasuk credentials/*.json dan openclaw-agent.sqlite — juga berkas include yang dirujuk dari openclaw.json.

> **MENEKAN TEMUAN BUKAN TINDAKAN RINGAN**
> Karena penekanan temuan dapat menyembunyikan risiko yang berdiri, menambah atau menghapusnya lewat perintah shell yang dijalankan agen menuntut persetujuan eksekusi, kecuali eksekusi memang sudah berjalan dengan security full dan ask off untuk otomasi lokal yang tepercaya.

Audit juga memperingatkan ketika beberapa pengirim DM berbagi sesi utama, dan menyarankan mode DM yang aman lewat session.dmScope. Ini adalah pengerasan untuk

kotak masuk bersama yang kooperatif, bukan isolasi untuk operator yang saling tidak percaya; untuk batas kepercayaan yang benar-benar terpisah, pakailah Gateway terpisah, atau bahkan pengguna dan host sistem operasi yang terpisah.

BAB 13 / PRAKTIKUM

Menegakkan lantai keamanan hari ini

Praktikum ini dikerjakan sebelum kanal pertama dibuka. Bila kanal Anda sudah terbuka dan langkah-langkah ini belum dikerjakan, kerjakan sekarang — urutan yang terlambat tetap lebih baik daripada tidak sama sekali.

Langkah 1 — pastikan Gateway tidak terjangkau dari luar

```
openclaw config get gateway.bind
openclaw config get gateway.auth.mode

# dari mesin lain di jaringan yang sama:
nc -vz <ip-mesin-gateway> 18789
```

Perintah terakhir harus gagal. Bila ia berhasil, berhenti dan perbaiki dulu.

Langkah 2 — jalankan audit dan baca temuannya

```
openclaw security audit
openclaw security audit --deep
openclaw security audit --json | jq '.summary'
```

Baca dulu; jangan langsung menjalankan --fix sebelum Anda memahami temuannya.

Langkah 3 — catat temuan sebelum memperbaiki

Kolom yang dicatat	Mengapa
checkId	Stabil antar versi, sehingga dapat dibandingkan
Tingkat keparahan	Menentukan urutan penanganan
Apakah --fix dapat menanganinya	Sebagian menuntut keputusan manusia
Keputusan Anda dan alasannya	Bahan untuk audit berikutnya

Kolom terakhir adalah yang membedakan audit yang berguna dari ritual. Temuan yang Anda putuskan untuk tidak perbaiki harus punya alasan tertulis, karena orang berikutnya akan menanyakannya.

Langkah 4 — uji kebijakan DM dengan nomor asing

Minta seseorang yang tidak ada di daftar izin mengirim pesan. Pesan itu seharusnya tidak pernah mencapai model; yang muncul hanyalah permintaan pemasangan. Menguji ini

sendiri jauh lebih meyakinkan daripada memercayai setelan.

```
openclaw pairing list <kanal>
openclaw pairing approve <kanal> <CODE>
```

Permintaan kedaluwarsa setelah satu jam dan dibatasi tiga per kanal.

BAB 13 / STUDI KASUS

Satu setelan yang membatalkan semua yang lain

Sebuah tim membangun konfigurasi yang rapi: kebijakan tool ketat, sandbox menyala, daftar izin sempit, audit berjalan bulanan. Mereka juga memindahkan Gateway ke server kantor supaya hidup dua puluh empat jam.

Saat pemindahan, seseorang mengubah alamat ikat menjadi 0.0.0.0 supaya lebih mudah diakses dari beberapa mesin. Perubahan itu masuk akal pada saat itu dan tidak ada yang meninjaunya.

Kontrol yang ada	Apakah masih bermakna
Kebijakan tool ketat	Ya, tetapi hanya setelah penyerang melewati autentikasi
Sandbox menyala	Ya, mengurangi radius ledakan
Daftar izin kanal	Tidak relevan; jalur masuknya bukan kanal
Audit bulanan	Akan menemukannya — sampai empat minggu kemudian

Port kendali yang terbuka ke jaringan yang tidak tepercaya memberi jalur langsung ke agen. Kontrol lain tidak hilang, tetapi seluruhnya berpindah ke belakang satu pintu yang seharusnya tidak pernah ada.

> **MENGAPA URUTAN LANTAI KEAMANAN PENTING**
> Inilah alasan binding berada di baris pertama lantai keamanan, bukan di tengah daftar. Kontrol yang paling penting adalah yang kegagalannya membatalkan kontrol lain, dan binding adalah satu-satunya yang punya sifat itu.

BAB 13 / LATIHAN

Latihan mandiri

1. Jalankan uji jangkauan port dari mesin lain dan buktikan ia gagal. Ulangi setiap kali Anda memindahkan Gateway.
2. Jalankan audit keamanan dan catat setiap temuan dengan empat kolom pada praktikum langkah tiga.
3. Uji kebijakan DM Anda dengan nomor yang tidak terdaftar dan pastikan pesannya tidak pernah mencapai model.
4. Susun lantai keamanan Anda sendiri dalam urutan menurun berdasarkan seberapa banyak kontrol lain yang dibatalkan bila butir itu gagal.

BAB 13 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Orang asing bisa memerintah agen	dmPolicy kemungkinan open dengan wildcard. Kembalikan ke pairing.
Bot menjawab semua obrolan grup	Gerbang mention mati atau groupPolicy open. Jalankan security audit --fix.
Kode pemasangan tidak pernah sampai	Antrean tertunda penuh; bawaannya tiga per kanal. Bersihkan lalu ulangi.
Model melihat teks dari pengirim tak terdaftar	Itu visibilitas konteks, bukan otorisasi. Setel contextVisibility.
Gateway dapat dijangkau dari jaringan lain	Binding bukan loopback. Kembalikan ke 127.0.0.1 dan pakai Tailscale atau SSH.

Tiga pertanyaan review

1. Mengapa pairing dipilih sebagai kebijakan DM bawaan, bukan allowlist yang terasa lebih ketat?
2. Jelaskan perbedaan otorisasi pemicu dan visibilitas konteks, lalu berikan satu contoh temuan yang termasuk kategori kedua.
3. Tim Anda ingin dua orang yang tidak saling percaya memakai satu agen. Mengapa session.dmScope saja tidak cukup, dan apa yang sebenarnya dibutuhkan?

Kunci pembahasan

Pairing dipilih karena ia menyeimbangkan keamanan dan kemudahan: allowlist murni menuntut Anda mengetahui identitas setiap pengirim lebih dulu, sementara pairing membiarkan orang baru meminta akses tanpa pernah menyentuh model sebelum Anda menyetujui. Otorisasi pemicu menentukan siapa yang boleh membuat agen bekerja,

sedangkan visibilitas konteks menentukan apa yang boleh dilihat model; contoh kategori kedua adalah model yang membaca teks kutipan dari pengirim di luar allowlist. Untuk dua orang yang saling tidak percaya, dmScope hanya memisahkan sesi di dalam satu batas kepercayaan yang sama; yang dibutuhkan adalah Gateway terpisah, idealnya pada pengguna atau host sistem operasi yang berbeda.

Rujukan: dokumentasi OpenClaw, halaman gateway/security, gateway/security/access-control, channels/pairing, dan cli/security, ditinjau 17 September 2026.

OPERASI HARIAN DI MACOS

Bab 14 Membaca dokumentasi resmi dan memverifikasi sendiri

Buku ini akan menua. OpenClaw merilis beberapa kali sebulan, dan sebagian perintah di halaman-halaman berikutnya pasti bergeser. Bab ini mengajarkan keterampilan yang tidak menua: cara memeriksa sendiri apakah sesuatu masih benar.

Tujuan belajar

Menemukan halaman dokumentasi yang tepat dengan cepat, mengambil versi Markdown-nya untuk dibaca alat, dan memakai perintah bawaan untuk memverifikasi keadaan mesin Anda. Pada akhir bab, pembaca tidak lagi bergantung pada buku ini untuk hal yang berubah cepat.

Tiga pintu masuk dokumentasi

Gambar 14.1 — Tiga cara masuk. Yang pertama paling berguna ketika Anda belum tahu nama halaman yang dicari.

```
# indeks seluruh halaman, satu berkas
curl -s https://docs.openclaw.ai/llms.txt | less

# halaman mana pun sebagai Markdown bersih
curl -s https://docs.openclaw.ai/gateway/security.md

# mencari dari terminal
openclaw docs "dm policy"
```

Akhiran .md bekerja pada setiap halaman dokumentasi.

BAB 14 / VERIFIKASI

Memeriksa keadaan mesin Anda

Dokumentasi memberi tahu apa yang seharusnya. Perintah berikut memberi tahu apa yang sebenarnya terjadi pada mesin Anda, dan selisih antara keduanya adalah tempat sebagian besar masalah berada.

Perintah	Menjawab pertanyaan
`openclaw --version`	Versi apa yang benar-benar terpasang
`openclaw status`	Diagnostik, probe, dan ringkasan pemakaian
`openclaw health`	Potret kesehatan Gateway lewat RPC
`openclaw doctor`	Pemeriksaan kesehatan dan perbaikan terpandu
`openclaw doctor --fix`	Menerapkan perbaikan dan migrasi yang aman
`openclaw config validate`	Apakah konfigurasi lolos skema
`openclaw config schema`	Skema JSON kanonik yang dipakai validasi
`openclaw logs`	Menarik log Gateway lewat RPC
`openclaw triage`	Diagnostik tersanitasi untuk diserahkan ke orang lain

SEBELUM MEMINTA BANTUAN PUBLIK

openclaw triage menghasilkan diagnostik yang sudah dibersihkan. Pakai ia — bukan tangkapan layar log mentah — ketika meminta bantuan di forum publik. Log mentah sering memuat token, nomor telepon, dan potongan percakapan.

BAB 14 / KEBIASAAN

Membaca seperti operator, bukan seperti turis

Halaman dokumentasi OpenClaw memuat medan read_when di bagian atas berkas sumbernya — daftar situasi yang membuat halaman itu relevan. Ketika Anda membaca versi Markdown-nya, medan itu terlihat, dan ia sering menghemat waktu lebih banyak daripada judulnya sendiri.

- ✓ Mencocokkan perintah dengan openclaw --version sebelum menjalankannya
- ✓ Membaca catatan rilis sebelum memperbarui
- ✓ Menjalankan config validate sesudah menyunting berkas konfigurasi
- ✓ Memakai triage, bukan tangkapan layar log, saat meminta bantuan
- ⚠ Menyalin perintah dari artikel blog lama tanpa memeriksa versinya

Gambar 14.2 — Lima kebiasaan. Yang terakhir adalah sumber sebagian besar pertanyaan yang jawabannya ternyata sudah berubah dua rilis lalu.

Satu catatan tentang artikel pihak ketiga. OpenClaw pernah bernama lain, dan ia berganti nama lebih dari sekali. Artikel yang memakai nama lama hampir pasti memuat default yang sudah berubah — terutama default keamanan, yang justru paling berbahaya bila salah.

BAB 14 / PRAKTIKUM

Membangun kebiasaan verifikasi

Praktikum ini membangun alur kerja yang akan Anda pakai setiap kali sebuah perintah di buku ini tidak cocok dengan mesin Anda. Kerjakan sekali sekarang supaya ia menjadi refleks.

Langkah 1 — ambil peta dokumentasi

```
curl -s https://docs.openclaw.ai/llms.txt > peta-dokumentasi.txt
wc -l peta-dokumentasi.txt
grep -i 'sandbox' peta-dokumentasi.txt
```

Satu berkas memetakan seluruh halaman beserta ringkasan satu barisnya.

Langkah 2 — baca satu halaman sebagai Markdown

```
curl -s https://docs.openclaw.ai/gateway/security.md | less
```

Akhiran .md memberi teks bersih beserta medan read_when di bagian atas.

Medan read_when adalah bagian yang paling menghemat waktu: ia mendaftar situasi yang membuat halaman itu relevan. Memindainya lebih cepat daripada membaca judul dan menebak.

Langkah 3 — bandingkan dengan mesin Anda

Pertanyaan	Perintah yang menjawabnya
Versi apa yang terpasang	`openclaw --version`
Apakah konfigurasi saya sah	`openclaw config validate`
Apa yang aktif sekarang	`openclaw status`
Apakah Gateway sehat	`openclaw health`
Apa yang perlu diperbaiki	`openclaw doctor`

Langkah 4 — latih membuat berkas triage

```
openclaw triage
```

Jalankan sekali dan baca isinya; ketahui apa yang dibagikan sebelum Anda membagikannya.

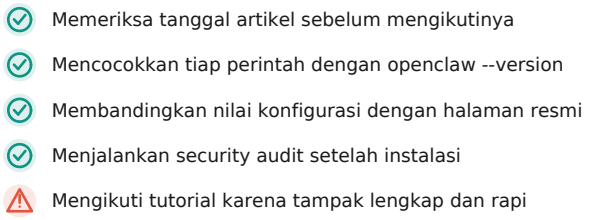
Membaca keluaran triage sendiri sebelum membagikannya adalah kebiasaan yang layak. Sanitasi bekerja, tetapi Anda tetap pihak yang bertanggung jawab atas apa yang keluar dari mesin Anda.

BAB 14 / STUDI KASUS

Tutorial yang sudah dua nama tertinggal

Seorang pemula mencari panduan instalasi dan menemukan artikel yang tampak lengkap dan ditulis rapi. Ia mengikuti setiap langkahnya. Artikel itu ditulis pada era ketika proyek ini masih memakai nama lain.

Konfigurasi yang dihasilkan berjalan. Agen menjawab. Semuanya tampak benar. Yang tidak ia ketahui: artikel itu memakai kebijakan DM dari versi yang belum memiliki kontrol akses, sehingga siapa pun yang menemukan nomornya dapat memerintah agennya.



Gambar 14.3 — Lima kebiasaan. Butir terakhir adalah alasan artikel lama tetap berbahaya: kualitas tulisannya tidak menua, tetapi isinya menua.

Yang menyelamatkan kasus ini akhirnya adalah audit keamanan bawaan, yang menandai kebijakan DM terbuka sebagai temuan. Satu perintah menemukan apa yang berminggu-minggu membaca tidak menemukan.

BAB 14 / LATIHAN

Latihan mandiri

1. Ambil peta dokumentasi dan cari tiga halaman yang relevan dengan use case Anda. Baca medan read_when masing-masing.
2. Pilih satu perintah dari bab mana pun di buku ini, cari halaman resminya, dan verifikasi bahwa ia masih cocok dengan versi Anda.
3. Jalankan triage dan baca seluruh keluarannya. Tandai apa yang Anda tidak nyaman bagikan, bila ada.
4. Susun daftar lima perintah verifikasi yang akan Anda jalankan setiap kali sesuatu berperilaku tidak seperti yang tertulis di buku ini.

BAB 14 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Perintah dari tutorial tidak dikenali	Cocokkan dengan openclaw --version, lalu cari halaman resminya.
Tidak tahu halaman mana yang relevan	Ambil llms.txt; ia memetakan seluruh halaman beserta ringkasannya.
Konfigurasi ditolak tanpa penjelasan jelas	openclaw config validate menyebut medan yang bermasalah.
Perlu berbagi log tanpa membocorkan rahasia	openclaw triage menghasilkan diagnostik yang sudah tersanitasi.
Default keamanan tidak cocok dengan artikel yang dibaca	Artikel kemungkinan dari era nama lama. Percayai dokumentasi resmi.

Tiga pertanyaan review

1. Mengapa akhiran .md pada URL dokumentasi lebih berguna bagi operator, bukan hanya bagi alat otomatis?
2. Kapan openclaw doctor --fix lebih tepat daripada menyunting konfigurasi dengan tangan?
3. Anda menemukan artikel yang memakai nama lama proyek ini. Bagian apa dari artikel itu yang paling berbahaya untuk diikuti, dan mengapa?

Kunci pembahasan

Akhiran .md memberi teks bersih tanpa navigasi dan memperlihatkan medan read_when, sehingga operator dapat memindai relevansi sebuah halaman dalam hitungan detik. doctor --fix lebih tepat ketika masalahnya adalah migrasi atau normalisasi yang sudah dikenali sistem, karena ia menerapkan perbaikan yang sudah diuji; suntingan tangan lebih

tepat ketika Anda memang bermaksud mengubah kebijakan. Bagian paling berbahaya dari artikel era nama lama adalah default keamanannya — versi awal tidak memiliki kontrol akses DM sama sekali, sehingga mengikuti konfigurasinya dapat membuka agen Anda kepada siapa saja.

Rujukan: dokumentasi OpenClaw, llms.txt, halaman cli/docs, cli/doctor, cli/status, dan cli/triage, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

Bab 15 Arsitektur Gateway dan protokol WebSocket

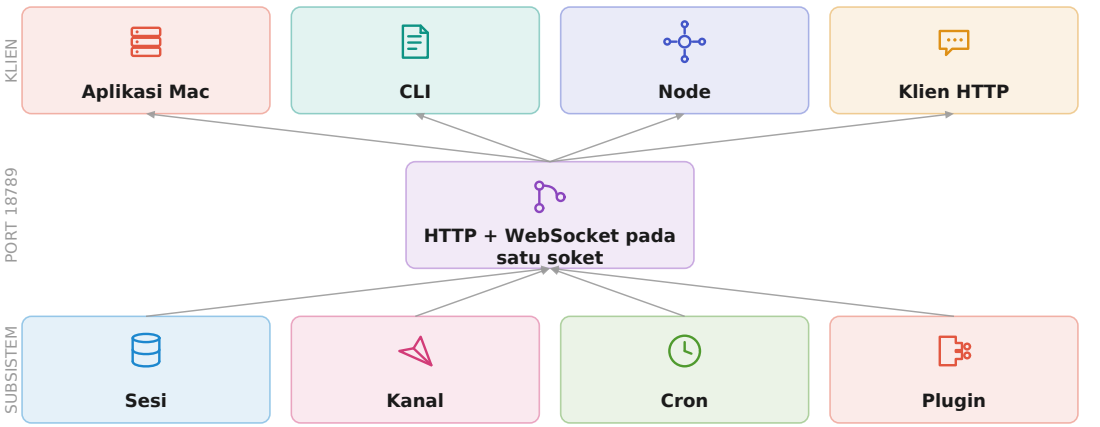
Gateway adalah pusat dari seluruh sistem: satu proses, satu port, beberapa protokol. Memahami bentuknya mengubah cara Anda mendiagnosis — dari menebak menjadi menelusuri.

Tujuan belajar

Menjelaskan bagaimana satu port melayani HTTP dan WebSocket sekaligus, menyebutkan peran dan cakupan klien, dan mengetahui apa yang bertahan melewati restart. Pada akhir bab, pembaca dapat membaca status Gateway sebagai gambaran, bukan sebagai daftar angka.

Satu proses, satu port

Gateway melayani lalu lintas HTTP dan WebSocket pada satu port — bawaannya 18789 — tanpa memerlukan reverse proxy, dengan memanfaatkan peristiwa upgrade native Node.js untuk memultipleks protokol.



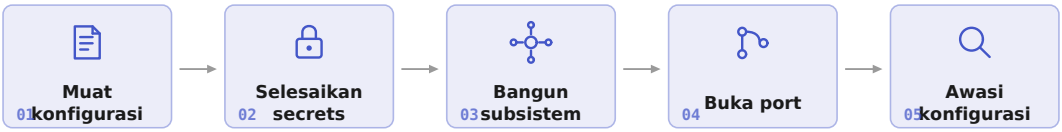
Gambar 15.1 — Satu port melayani semuanya. Tidak ada reverse proxy yang wajib, dan itu menyederhanakan penyebaran sekaligus memusatkan risiko.

Konsekuensi memusatkan risiko layak disadari: satu port yang terbuka ke jaringan yang salah membuka seluruh permukaan sekaligus. Itulah sebabnya Bab 13 menempatkan binding loopback pada baris pertama lantai keamanan.

BAB 15 / STARTUP

Urutan yang harus berhasil

Startup Gateway adalah urutan berfase — dari migrasi konfigurasi dan penyelesaian secrets sampai pengelolaan siklus hidup sidecar — dan tiap fase punya semantik kegagalan yang jelas. Fase terakhir menyalakan pengawas konfigurasi yang dibahas di Bab 16.



Gambar 15.2 — Fase startup yang disederhanakan. Kegagalan pada dua fase pertama menghentikan Gateway sebelum port pernah terbuka.

SALINAN BAIK TIDAK DIPULIHKAN SENDIRI

Bila openclaw.json gagal validasi, startup Gateway gagal atau reload dilewati, dan runtime saat ini mempertahankan konfigurasi terakhir yang diterima. Gateway menyimpan salinan tepercaya terakhir yang diketahui baik setelah setiap startup berhasil, tetapi startup dan hot reload tidak memulihkannya otomatis — hanya openclaw doctor --fix yang melakukannya.

BAB 15 / KETAHANAN

Apa yang bertahan melewati restart

Restart bukan peristiwa yang menghapus segalanya. Giliran agen yang terputus dilanjutkan secara otomatis, subagen dan tugas latar dipulihkan, dan pengiriman yang mengantre dikuras. Mengetahui ini mengubah sikap Anda terhadap restart dari takut menjadi terukur.

Keadaan	Setelah restart
Giliran agen yang sedang berjalan	Dilanjutkan otomatis
Subagen dan tugas latar	Dipulihkan
Pengiriman yang mengantre	Dikuras setelah kanal siap
Pekerjaan cron yang jatuh tempo saat mati	Bergantung pada penjadwal; periksa riwayat
Sesi dan riwayat	Aman di SQLite per agen

Selama hot reload kanal atau plugin, rute webhook kanal yang dihosting Gateway mengembalikan 503 dengan Retry-After satu detik sampai ingress pengganti terdaftar. Pengirim harus menghormati respons percobaan ulang itu; respons tersebut tidak mengakui pengiriman.

BAB 15 / PRAKTIKUM

Mengamati Gateway dari luar

Praktikum ini memperlakukan Gateway sebagai kotak hitam dan memeriksa apa yang terlihat dari luar. Sudut pandang ini yang akan Anda pakai ketika sesuatu bermasalah dan Anda tidak punya akses ke dalamnya.

Langkah 1 — lihat satu port melayani dua protokol

```
lsof -i :18789
curl -s -o /dev/null -w '%{http_code}\n' http://127.0.0.1:18789/
```

Satu proses, satu port, dua jenis lalu lintas.

Langkah 2 — amati urutan startup

```
openclaw gateway stop
openclaw logs --follow &
openclaw gateway
```

Perhatikan fase mana yang muncul lebih dulu dan berapa lama masing-masing.

Yang perlu Anda catat: pada fase mana port mulai menerima sambungan. Segala sesuatu yang gagal sebelum titik itu menghasilkan Gateway yang tidak pernah dapat dihubungi, dan diagnosisnya berbeda sama sekali dari Gateway yang dapat dihubungi tetapi berperilaku salah.

Langkah 3 — uji pemulihan setelah restart

Uji	Cara	Hasil yang diharapkan
Giliran berjalan	Mulai pekerjaan panjang, restart di tengah	Dilanjutkan otomatis
Pengiriman mengantre	Restart saat ada balasan tertunda	Dikuras setelah siap
Sesi dan riwayat	Periksa percakapan lama sesudahnya	Utuh di SQLite
Kanal	Kirim pesan segera setelah naik	Tersambung ulang sendiri

Langkah 4 — kenali bunyi 503 yang normal

Ubah satu setelan kanal dan amati bahwa rute webhook mengembalikan penolakan sementara sampai ingress pengganti terdaftar. Mengenali ini sebagai perilaku normal akan menghemat satu laporan insiden palsu di kemudian hari.

BAB 15 / STUDI KASUS

Restart yang ditakuti selama setahun

Sebuah tim tidak pernah merestart Gateway mereka kecuali terpaksa. Alasannya: pada bulan pertama, satu restart memutus pekerjaan yang sedang berjalan dan mereka menyimpulkan restart itu berbahaya.

Akibatnya, pembaruan ditunda berbulan-bulan, konfigurasi yang menuntut restart dibiarkan tidak aktif, dan mesin berjalan dengan versi yang semakin tertinggal. Ketakutan terhadap satu operasi membentuk seluruh kebiasaan operasional mereka.

ASUMSI MEREKA		KENYATAAN
Hilang	**Giliran berjalan**	Dilanjutkan otomatis
Hilang	**Tugas latar**	Dipulihkan
Hilang	**Pengiriman antre**	Dikuras setelah siap
Berisiko	**Riwayat**	Aman di SQLite
Jangan restart	**Kesimpulan**	Restart adalah operasi biasa

Gambar 15.3 — Selisih antara asumsi dan kenyataan yang berlangsung setahun. Satu jam pengujian akan menutupnya.

Yang mereka lewatkan adalah menguji asumsinya. Satu sesi pengujian terencana pada mesin percobaan akan memperlihatkan bahwa pemulihan bekerja, dan mengubah restart dari peristiwa menakutkan menjadi operasi rutin.

> **POLA YANG LAYAK DIPERHATIKAN**
> Asumsi operasional yang tidak pernah diuji punya kecenderungan mengeras menjadi kebijakan. Bila tim Anda menghindari sebuah operasi, tanyakan kapan terakhir kali ada yang mengujinya — jawabannya sering mengejutkan.

BAB 15 / LATIHAN

Latihan mandiri

1. Amati urutan startup Gateway Anda dan catat pada fase mana port mulai menerima sambungan.
2. Jalankan keempat uji pemulihan pada praktikum langkah tiga dan catat hasil sebenarnya, bukan yang Anda harapkan.
3. Picu penolakan sementara pada rute webhook dengan mengubah setelan kanal, lalu pastikan tim Anda mengenalinya sebagai perilaku normal.
4. Daftarkan setiap operasi yang tim Anda hindari, lalu tentukan kapan terakhir kali masing-masing benar-benar diuji.

BAB 15 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Gateway tidak pernah membuka port	Kegagalan pada fase awal. Jalankan config validate dan baca log startup.
Webhook menerima 503 sesaat	Normal selama hot reload kanal. Pengirim harus menghormati Retry-After.
Konfigurasi rusak dan Gateway memakai yang lama	Perilaku yang benar. Perbaiki berkasnya, lalu doctor --fix bila perlu.
Pekerjaan hilang setelah restart	Periksa riwayat run; giliran dan tugas latar seharusnya pulih sendiri.
Port 18789 dapat dijangkau dari luar	Binding salah. Satu port memusatkan seluruh permukaan risiko.

Tiga pertanyaan review

1. Sebutkan satu keuntungan dan satu kerugian dari melayani HTTP dan WebSocket pada satu port tanpa reverse proxy.
2. Mengapa salinan konfigurasi yang diketahui baik sengaja tidak dipulihkan otomatis saat startup?
3. Sebuah integrasi mengeluh menerima 503 saat Anda mengubah konfigurasi kanal. Apakah ini kerusakan? Jelaskan jawabannya.

Kunci pembahasan

Keuntungannya adalah penyebaran yang sederhana tanpa komponen tambahan yang harus dirawat; kerugiannya adalah seluruh permukaan risiko terpusat pada satu port, sehingga satu kesalahan binding membuka semuanya sekaligus. Salinan baik tidak dipulihkan otomatis karena pemulihan diam-diam akan menyembunyikan fakta bahwa

suntingan Anda ditolak, dan Anda akan terus mengira perubahan sudah aktif. 503 saat mengubah konfigurasi kanal bukan kerusakan melainkan perilaku yang dirancang: ia menandakan ingress pengganti belum terdaftar, dan pengirim yang menghormati Retry-After akan berhasil pada percobaan berikutnya.

Rujukan: dokumentasi OpenClaw, halaman concepts/architecture, gateway/restart-recovery, dan gateway/configuration, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

Bab 16 Berkas openclaw.json, validasi ketat, dan hot reload

Hampir semua perilaku OpenClaw ditentukan satu berkas. Berkas itu divalidasi dengan keras, diawasi terus-menerus, dan diperlakukan sebagai tidak tepercaya sampai ia lolos skema. Sikap yang tampak berlebihan ini justru yang menyelamatkan Anda dari konfigurasi setengah jadi pada mesin produksi.

Tujuan belajar

Menyunting konfigurasi dengan aman, membaca hasil validasi, dan memperkirakan apakah sebuah perubahan akan diterapkan panas atau menuntut restart. Pada akhir bab, pembaca dapat mengubah konfigurasi produksi tanpa memutus layanan.

Lokasi dan bentuk

OpenClaw membaca konfigurasi JSON5 opsional dari ~/.openclaw/openclaw.json. Bila berkas itu tidak ada, OpenClaw memakai default yang aman. Jalur konfigurasi yang aktif harus berupa berkas biasa.

```
openclaw config get  <path>
openclaw config set  <path> <value>
openclaw config patch <json>
openclaw config unset <path>
openclaw config file
openclaw config schema
openclaw config validate
```

Permukaan perintah konfigurasi. Menyunting berkas langsung juga sah.

VALIDASI KETAT, TANPA KOMPROMI
OpenClaw hanya menerima konfigurasi yang benar-benar cocok dengan skema. Kunci yang tidak dikenal, tipe yang salah bentuk, atau nilai yang tidak valid membuat Gateway menolak untuk mulai. Satu-satunya pengecualian di tingkat akar adalah $schema bertipe string, supaya editor dapat melampirkan metadata JSON Schema.

BAB 16 / PENGAWAS

Bagaimana perubahan diterapkan

Gateway mengawasi ~/.openclaw/openclaw.json dan menerapkan perubahan secara otomatis — tidak perlu restart manual untuk sebagian besar setelan. Suntingan berkas langsung diperlakukan sebagai tidak tepercaya sampai ia lolos validasi. Pengawas menunggu keributan tulis-sementara dan ganti-nama dari editor mereda, membaca berkas final, lalu menolak suntingan eksternal yang tidak valid tanpa menulis ulang openclaw.json.

Perencanaan reload mengklasifikasikan setiap jalur yang berubah menjadi salah satu hasil: restart Gateway, atau hot reload yang menerapkan perubahan sambil menjaga proses tetap berjalan — yang dapat mencakup menjalankan ulang subsistem pemilik seperti kanal, cron, atau heartbeat.

HOT RELOAD		RESTART
Tetap hidup	Proses Gateway	Dimulai ulang
Sebagian direstart	Sambungan kanal	Semua tersambung ulang
Tidak terganggu	Sesi berjalan	Dipulihkan setelah naik
Kredensial provider	Contoh perubahan	Struktur inti gateway
Seketika	Waktu	Beberapa detik

Gambar 16.1 — Dua hasil perencanaan reload. Sistem yang memutuskan, bukan Anda; yang dapat Anda pilih hanyalah modenya.

Mode reload

Mode	Perilaku
hybrid	Menerapkan panas perubahan yang aman seketika, dan otomatis merestart untuk yang kritis
off	Mematikan pengawasan berkas; perubahan berlaku pada restart manual berikutnya

MODE LAMA SUDAH TIDAK ADA

Mode hot dan restart yang lebih lama sudah dipensiunkan; openclaw doctor --fix memetakan keduanya ke hybrid. Debounce reload tidak lagi dapat dikonfigurasi dan berjalan di balik nilai bawaan bawaan sistem.

BAB 16 / KEGAGALAN

Ketika konfigurasi ditolak

Bila Anda melihat pesan config reload skipped dengan keterangan invalid config, atau startup melaporkan Invalid config, periksa konfigurasinya, jalankan openclaw config validate, lalu jalankan openclaw doctor --fix untuk perbaikan.

```
openclaw config validate
openclaw doctor --fix

# tulisan yang ditolak disimpan untuk diperiksa
ls ~/.openclaw/openclaw.json.rejected.*
```

Tulisan yang ditolak disimpan sebagai berkas .rejected bertanda waktu.

Gateway memblokir tulisan yang tampak seperti penimpaan tidak sengaja — menjatuhkan gateway.mode, kehilangan blok meta, atau menyusutkan berkas lebih dari separuh — kecuali tulisan itu secara eksplisit mengizinkan perubahan destruktif.

> **DITOLAK BUKAN BERARTI TIDAK TERSIMPAN**
> Tulisan yang dikelola Gateway, termasuk config.set, menolak kandidat sebelum disimpan. Suntingan tangan dan tulisan dari proses CLI terpisah dapat tetap berada di disk meskipun pengawas menolak mengaktifkannya. Reload tidak pernah memigrasikan keadaan workspace.

Berkas yang dipecah dengan $include

Ketika Anda menyunting berkas sumber yang dirujuk lewat $include, OpenClaw merencanakan reload dari tata letak yang ditulis pada sumbernya, bukan dari tampilan rata di memori. Itu menjaga keputusan hot-apply dan restart tetap dapat diperkirakan meskipun satu bagian tingkat atas berada di berkas terpisah. Perencanaan reload gagal tertutup bila tata letak sumbernya ambigu.

BAB 16 / PRAKTIKUM

Menyunting konfigurasi tanpa memutus layanan

Praktikum ini membangun alur penyuntingan yang aman untuk dipakai pada mesin yang melayani orang. Kerjakan di mesin percobaan lebih dulu sampai urutannya menjadi kebiasaan.

Langkah 1 — jadikan konfigurasi dapat ditelusuri

```
cd ~/.openclaw
git init                      # bila belum
git add openclaw.json
git commit -m 'garis dasar konfigurasi'
```

Pastikan konfigurasi memuat rujukan rahasia, bukan nilai mentah; lihat Bab 53.

Langkah 2 — sunting lalu validasi sebelum menyimpan final

```
cp openclaw.json /tmp/openclaw.json.bak
# sunting berkasnya
openclaw config validate
openclaw logs --follow          # amati keputusan reload
```

Pengawas menunggu keributan tulis-sementara editor mereda sebelum membaca berkas final.

Langkah 3 — kenali dua jenis kegagalan

Jenis	Gejala	Pemulihan
Ditolak sebelum disimpan	Perubahan tidak pernah menyentuh disk	Berkas lama masih benar; perbaiki lalu ulangi
Ditolak setelah disimpan	Berkas berubah tetapi tidak diaktifkan	Perbaiki berkasnya sendiri; restart tidak cukup
Penimpaan tidak sengaja	Berkas menyusut drastis	Diblokir kecuali diizinkan eksplisit
Tata letak include ambigu	Perencanaan reload gagal tertutup	Rapikan supaya tiap bagian punya sumber jelas

Langkah 4 — amati hot reload versus restart

Ubah satu setelan yang aman — misalnya tingkat logging — dan amati ia diterapkan tanpa proses berhenti. Lalu ubah sesuatu yang menyentuh struktur inti dan amati Gateway memulai ulang sendiri. Mengetahui bentuk kedua perilaku itu mencegah Anda mengira restart otomatis adalah kerusakan.

```
ls ~/.openclaw/openclaw.json.rejected.* 2>/dev/null
```

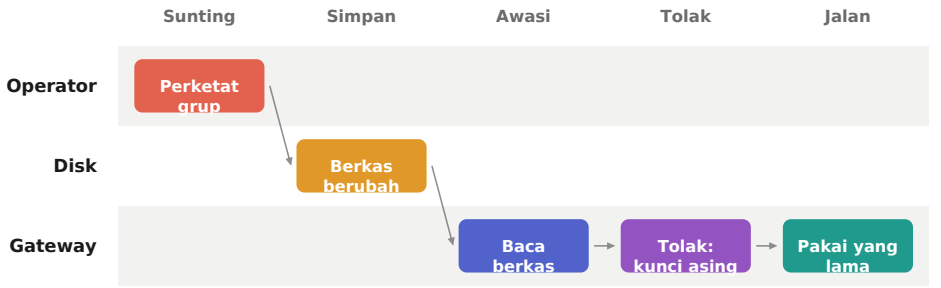
Tulisan yang ditolak disimpan bertanda waktu untuk diperiksa.

BAB 16 / STUDI KASUS

Suntingan yang tersimpan tetapi tidak pernah aktif

Seorang operator menyunting konfigurasi dari terminal untuk memperketat kebijakan grup. Ia menyimpan berkas, melihat perubahannya di editor, dan menganggap pekerjaan selesai.

Suntingan itu memuat satu kunci yang salah ketik. Pengawas menolak mengaktifkannya, dan Gateway mempertahankan konfigurasi terakhir yang diterima. Berkas di disk memuat kebijakan baru; yang berjalan adalah kebijakan lama.



Gambar 16.2 — Berkas dan runtime berpisah. Melihat berkas tidak membuktikan apa yang sedang berjalan.

Selama tiga minggu, tim percaya grup mereka sudah diperketat. Mereka baru menyadari setelah seseorang di luar daftar berhasil memicu agen di sebuah grup — perilaku yang seharusnya sudah mustahil.

> **BUKTI BUKAN PADA BERKAS, TETAPI PADA RUNTIME**
> Menyunting berkas dan melihat hasilnya di editor bukan bukti bahwa konfigurasi aktif. Jalankan openclaw config validate setelah setiap suntingan, dan periksa adanya berkas .rejected. Keduanya memakan sepuluh detik.

BAB 16 / LATIHAN

Latihan mandiri

1. Jadikan konfigurasi Anda repositori git dan buat komit garis dasar. Periksa lebih dulu bahwa ia tidak memuat rahasia mentah.
2. Sunting konfigurasi dengan kunci yang sengaja salah ketik, lalu telusuri seluruh gejalanya sampai Anda menemukan berkas yang ditolak.
3. Ubah satu setelan yang hot-apply dan satu yang menuntut restart, lalu catat perbedaan perilakunya pada log.
4. Susun prosedur penyuntingan konfigurasi produksi Anda dalam lima langkah, termasuk cara membuktikan perubahan benar-benar aktif.

BAB 16 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Suntingan tidak berpengaruh sama sekali	Kemungkinan ditolak validasi. Cari berkas .rejected dan jalankan config validate.
Gateway menolak mulai setelah disunting	Ada kunci tak dikenal atau tipe salah. Skema tidak memberi toleransi.
Perubahan model tidak muncul di menu	Sebagian katalog dibangun saat startup. Restart Gateway bila perlu.
Berkas menyusut drastis setelah tulisan otomatis	Gateway seharusnya memblokirnya. Periksa apakah tulisan mengizinkan destruktif.
Reload tidak dapat diperkirakan pada konfigurasi berpecah	Tata letak $include ambigu. Rapikan supaya tiap bagian punya sumber yang jelas.

Tiga pertanyaan review

1. Mengapa suntingan berkas langsung diperlakukan sebagai tidak tepercaya sampai lolos validasi, padahal Anda pemilik mesinnya?
2. Apa bedanya tulisan yang ditolak sebelum disimpan dan tulisan yang tetap ada di disk tetapi tidak diaktifkan? Mengapa perbedaan ini penting saat memulihkan?
3. Rancang prosedur menyunting konfigurasi produksi yang meminimalkan risiko layanan terputus.

Kunci pembahasan

Suntingan langsung tidak tepercaya karena editor menulis berkas dalam beberapa tahap, sehingga pengawas dapat membaca keadaan setengah jadi; menunggu validasi mencegah Gateway mengaktifkan konfigurasi yang belum utuh. Tulisan yang ditolak sebelum disimpan tidak pernah menyentuh disk sehingga berkas lama tetap benar, sedangkan

suntingan tangan yang ditolak tetap ada di disk — artinya saat memulihkan Anda harus memperbaiki berkasnya sendiri, bukan sekadar merestart. Prosedur yang aman: salin berkas lebih dulu, sunting, jalankan config validate sebelum menyimpan final, terapkan pada jam sepi, lalu verifikasi dengan health dan satu pesan uji.

Rujukan: dokumentasi OpenClaw, halaman gateway/configuration, gateway/configuration/hot-reload, dan cli/config, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

Bab 17 Autentikasi Gateway, binding, dan peran operator

Gateway menuntut autentikasi secara bawaan, dan terikat ke loopback secara bawaan. Dua default itu adalah alasan mengapa pemasangan OpenClaw yang polos tetap relatif aman. Bab ini membahas apa yang terjadi ketika Anda mengubah salah satunya.

Tujuan belajar

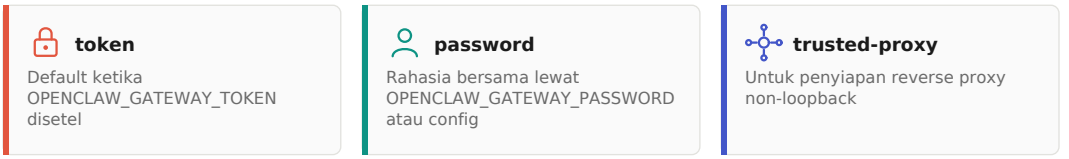
Memilih mode binding yang tepat, memahami tiga mode autentikasi, dan mengetahui kapan identitas dapat menggantikan rahasia bersama. Pada akhir bab, pembaca dapat membuka akses jarak jauh tanpa membuka pintu untuk orang lain.

Mode binding

Mode	Perilaku
loopback	Bawaannya. Hanya dapat dijangkau dari mesin itu sendiri
lan	Mendengarkan pada antarmuka jaringan lokal
tailnet	Mendengarkan langsung pada alamat IP Tailscale
auto	Default efektif di dalam kontainer; menjadi 0.0.0.0 untuk port-forwarding

KONTAINER MENGUBAH DEFAULT
Di dalam lingkungan kontainer yang terdeteksi, default efektifnya adalah auto dan menjadi 0.0.0.0 — kecuali Tailscale serve atau funnel aktif, yang selalu memaksa loopback. Bila Anda menjalankan di Docker dan mengira masih loopback, periksa ulang.

Tiga mode autentikasi



Gambar 17.1 — Tiga mode handshake. Yang ketiga memindahkan keputusan autentikasi ke proxy di depannya, dan itu hanya aman bila proxy itu benar.

```
{
  gateway: {
    bind: "loopback",
    auth: { mode: "token", token: "rahasia-anda" },
  },
}
```

Bentuk paling sederhana yang tetap aman untuk dipakai sendiri.

BAB 17 / IDENTITAS

Ketika identitas menggantikan rahasia

Ketika tailscale.mode bernilai serve dan gateway.auth.allowTailscale bernilai true, permintaan proxy Serve yang valid dapat terautentikasi lewat header identitas Tailscale tanpa menyodorkan token atau kata sandi. OpenClaw memverifikasi identitas itu dengan menyelesaikan alamat x-forwarded-for melalui daemon Tailscale lokal lalu mencocokkannya dengan header sebelum menerimanya.

OpenClaw hanya memperlakukan sebuah permintaan sebagai Serve bila ia tiba dari loopback dengan header x-forwarded-for, x-forwarded-proto, dan x-forwarded-host milik Tailscale. Untuk menuntut kredensial eksplisit, setel gateway.auth.allowTailscale ke false, atau paksa gateway.auth.mode ke password.

> **SATU ASUMSI YANG HARUS ANDA SETUJUI**
> Alur tanpa token ini mengandaikan host Gateway tepercaya. Setel ia ke false bila Anda menginginkan autentikasi rahasia bersama di semua jalur. Endpoint HTTP API tidak memakai autentikasi header itu dan tetap mengikuti mode autentikasi HTTP normal Gateway.

Autentikasi trusted-proxy mengharapkan proxy sadar-identitas yang bukan loopback secara bawaan. Reverse proxy loopback pada host yang sama menuntut gateway.auth.trustedProxy.allowLoopback bernilai true secara eksplisit.

BAB 17 / LAYANAN

Layanan launchd di macOS

```
ai.openclaw.gateway        # label bawaan
ai.openclaw.<profile>      # label untuk profil bernama

openclaw gateway restart   # cara yang benar untuk merestart
```

Jangan merantai stop lalu start sebagai pengganti restart.

Perintah kendali layanan native hanya menerima lingkungan sistem operasi yang diperlukan untuk pencarian eksekutabel, identitas akun, locale, dan perutean manajer layanan. Ia tidak mewarisi kredensial aplikasi atau variabel shell sembarangan. Muatan Gateway dan definisi layanannya yang terpasang mempertahankan lingkungan yang dikonfigurasi terpisah.

Konsekuensinya sering mengejutkan: variabel lingkungan yang bekerja ketika Anda menjalankan Gateway dari terminal bisa hilang ketika ia berjalan sebagai layanan. Bila kredensial tiba-tiba tidak ditemukan setelah memasang layanan, inilah tempat pertama yang harus diperiksa.

BAB 17 / PRAKTIKUM

Membuka akses tanpa membuka pintu

Praktikum ini memindahkan akses dari “bekerja di mesin ini saja” menjadi “bekerja dari mana saja, untuk orang yang berhak” — tanpa satu pun langkah yang memaparkan port ke jaringan.

Langkah 1 — catat keadaan autentikasi sekarang

```
openclaw config get gateway.bind
openclaw config get gateway.auth.mode
openclaw config get gateway.tailscale.mode
launchctl list | grep ai.openclaw
```

Perintah terakhir memperlihatkan apakah layanan launchd benar-benar terpasang.

Langkah 2 — buktikan autentikasi benar-benar menolak

```
curl -s -o /dev/null -w '%{http_code}\n' http://127.0.0.1:18789/
# lalu ulangi dengan token yang benar
```

Permintaan tanpa kredensial harus ditolak. Bila ia diterima, berhenti dan periksa mode autentikasi.

Langkah 3 — uji lingkungan layanan

Ini langkah yang menjelaskan kelas kesalahan yang membingungkan banyak orang. Jalankan Gateway dari terminal, catat bahwa kredensial ditemukan. Lalu jalankan sebagai

layanan dan periksa apakah kredensial yang sama masih terbaca.

Cara menjalankan	Lingkungan yang diwarisi	Akibatnya
Dari terminal	Seluruh variabel shell Anda	Kredensial dari shell terbaca
Sebagai layanan	Hanya yang dibutuhkan sistem	Variabel shell tidak ada
Layanan dengan env terkonfigurasi	Yang Anda tetapkan sendiri	Terkendali

Perintah kendali layanan native hanya menerima lingkungan sistem operasi yang diperlukan untuk pencarian eksekutabel, identitas akun, locale, dan perutean manajer layanan. Ia tidak mewarisi kredensial aplikasi atau variabel shell sembarangan.

Langkah 4 — restart dengan cara yang benar

```
openclaw gateway restart
# bukan: openclaw gateway stop && openclaw gateway start
```

Merantai stop dan start bukan pengganti restart.

BAB 17 / STUDI KASUS

Kredensial yang hilang saat dipasang sebagai layanan

Seorang insinyur menguji konfigurasi dari terminal selama dua hari. Semuanya bekerja: model merespons, kanal tersambung, tool berjalan. Ia lalu memasang Gateway sebagai layanan supaya hidup terus, dan semuanya berhenti bekerja.

Galat yang muncul menyebut kredensial penyedia model tidak ditemukan. Ia memeriksa berkas konfigurasi — tidak ada yang berubah. Ia menjalankan ulang dari terminal — berfungsi lagi. Dua hari berlalu sebelum ia menyadari bahwa kunci API selama ini berasal dari berkas profil shell-nya.

Gambar 17.2 — Pengujian dari terminal dapat menyembunyikan ketergantungan pada lingkungan shell selama berhari-hari.

Perbaikan jangka pendeknya adalah mengonfigurasi lingkungan layanan secara eksplisit. Perbaikan jangka panjangnya adalah memindahkan kredensial ke rujukan rahasia seperti dibahas di Bab 53, sehingga sumbernya tidak lagi bergantung pada cara Gateway dijalankan.

> **ATURAN DIAGNOSIS YANG MENGHEMAT BERJAM-JAM**
> Bila sesuatu bekerja dari terminal tetapi gagal sebagai layanan, periksa lingkungan sebelum memeriksa apa pun yang lain. Pola ini menjelaskan mayoritas kasus dalam kategori itu.

BAB 17 / LATIHAN

Latihan mandiri

1. Buktikan bahwa permintaan tanpa kredensial ditolak oleh Gateway Anda, lalu ulangi dengan kredensial yang benar.
2. Daftarkan setiap kredensial yang dipakai Gateway Anda dan dari mana masing-masing berasal: konfigurasi, lingkungan, atau penyimpanan rahasia.
3. Jalankan Gateway dari terminal lalu sebagai layanan, dan bandingkan lingkungan yang terbaca pada keduanya.
4. Bila Anda memakai Tailscale, tuliskan siapa saja yang berada di tailnet Anda dan apakah semuanya pantas memiliki akses operator.

BAB 17 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Gateway terjangkau dari jaringan padahal disetel loopback	Berjalan di dalam kontainer; default efektifnya auto.
Kredensial hilang setelah dipasang sebagai layanan	Layanan tidak mewarisi variabel shell. Konfigurasikan lingkungannya sendiri.
Klien Tailscale ditolak meski identitas benar	allowTailscale kemungkinan false, atau permintaan tidak tiba lewat Serve.
HTTP API menolak header identitas	Perilaku yang benar. Endpoint HTTP mengikuti mode autentikasi HTTP normal.
Restart lewat stop lalu start menimbulkan keanehan	Pakai openclaw gateway restart; rantai manual bukan penggantinya.

Tiga pertanyaan review

1. Mengapa autentikasi lewat identitas Tailscale hanya diterima bila permintaan tiba dari loopback dengan tiga header tertentu?
2. Apa risiko menyalakan allowTailscale pada host Gateway yang dipakai bersama beberapa orang?
3. Rancang konfigurasi untuk satu Mac mini di kantor yang harus dapat diakses dari luar kantor oleh tiga orang.

Kunci pembahasan

Syarat loopback dan tiga header itu memastikan permintaan benar-benar melewati proxy Serve milik Tailscale, bukan dipalsukan oleh klien yang mengaku-ngaku; tanpa syarat itu, siapa pun yang dapat menjangkau port dapat menyuntikkan header identitas. Risikonya pada host bersama adalah setiap pengguna tailnet yang lolos identitas mendapat akses operator tanpa rahasia tambahan, sehingga batas kepercayaan menjadi seluas tailnet Anda. Untuk Mac mini kantor, pertahankan bind loopback, pakai Tailscale Serve untuk Control UI, biarkan allowTailscale menyala hanya bila ketiga orang itu memang setara kewenangannya, dan gunakan mode password bila tidak.

Rujukan: dokumentasi OpenClaw, halaman gateway, gateway/tailscale, dan gateway/remote, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

Bab 18 Model provider, kredensial, dan failover

OpenClaw memisahkan dua hal yang sering dicampur: siapa Anda pada penyedia model, dan bagaimana endpoint penyedia itu dikonfigurasi. Yang pertama disebut profil auth dan tinggal di basis data; yang kedua tinggal di berkas konfigurasi. Mencampurnya adalah sumber kebingungan yang berulang.

Tujuan belajar

Menyimpan kredensial pada tempat yang benar, memahami di mana profil auth tersimpan, dan memigrasikan pemasangan lama. Pada akhir bab, pembaca dapat menambah penyedia baru tanpa merusak yang sudah ada.

Gambar 18.1 — Dua tempat yang berbeda peran. Menaruh kunci API di models.providers atau baseUrl di profil auth sama-sama salah.

OpenClaw membaca profil auth dari openclaw-agent.sqlite milik tiap agen. Detail endpoint — baseUrl, api, id model, header, timeout — termasuk di bawah models.providers.<id> pada openclaw.json atau models.json, bukan di profil auth.

BAB 18 / MIGRASI

Pemasangan lama dan jalur migrasinya

Bila pemasangan yang lebih tua masih memiliki auth-profiles.json, auth-state.json, atau bentuk datar seperti objek yang memetakan nama penyedia langsung ke apiKey, jalankan openclaw doctor --fix untuk mengimpornya ke SQLite. Doctor menyimpan cadangan bertanda waktu di samping berkas JSON aslinya.

```
openclaw doctor --fix

openclaw models status
openclaw models list
openclaw models set <provider/model>
openclaw models scan
```

Migrasi lebih dulu, baru periksa katalog model.

Token Anthropic lewat Claude Code

Jalankan claude setup-token pada mesin mana pun yang memiliki Claude Code terpasang. Ia mencetak token berumur panjang yang diawali sk-ant-oat01-. Simpan token itu pada host Gateway. Perintah tersebut menuntut TTY interaktif, jadi ia tidak dapat dijalankan dari skrip latar tanpa terminal.

> **SEBELUM MENYIMPAN KUNCI DI MESIN BERSAMA**
> Menyimpan kredensial adalah tindakan yang meninggalkan jejak di disk. Bab 53 membahas apa saja yang ditulis OpenClaw ke disk dan berkas mana yang memuat kredensial. Bila mesin ini berbagi dengan orang lain, baca bab itu sebelum menyimpan token apa pun.

BAB 18 / FAILOVER

Ketika satu penyedia gagal

OpenClaw dapat memutar profil auth dan jatuh ke model cadangan. Mekanisme ini berguna justru pada saat Anda tidak sedang memperhatikan — ketika penyedia utama membalas dengan batas laju pada jam sibuk.

Gambar 18.2 — Urutan failover. Rantai cadangan hanya berguna bila Anda benar-benar mengujinya, bukan sekadar menuliskannya.

Satu saran praktis: uji rantai cadangan Anda dengan sengaja mematikan kredensial utama pada lingkungan uji. Rantai failover yang tidak pernah diuji punya kebiasaan gagal pada percobaan pertamanya yang nyata, biasanya karena model cadangan tidak memiliki kemampuan yang sama — misalnya tidak mendukung tool atau gambar.

BAB 18 / PRAKTIKUM

Menata kredensial dan rantai cadangan

Praktikum ini memisahkan dua hal yang sering tercampur — kredensial dan endpoint — lalu menguji rantai cadangan Anda sebelum ia dibutuhkan sungguhan.

Langkah 1 — periksa apa yang ada sekarang

```
openclaw models status
openclaw models list
openclaw config get models.providers
```

Perintah ketiga memperlihatkan endpoint; kredensialnya ada di tempat lain.

Langkah 2 — pastikan tidak ada kredensial di berkas konfigurasi

```
grep -iE 'api[_-]?key|token|secret|sk-' ~/.openclaw/openclaw.json
```

Keluaran kosong adalah hasil yang Anda inginkan. Bila ada, lihat Bab 53.

Langkah 3 — migrasikan format lama bila ada

```
ls ~/.openclaw/auth-profiles.json ~/.openclaw/auth-state.json 2>/dev/null
openclaw doctor --fix
```

Doctor mengimpor format lama ke SQLite dan menyimpan cadangan bertanda waktu.

Langkah 4 — uji rantai cadangan dengan sengaja

Uji	Cara	Yang Anda buktikan
Failover terjadi	Nonaktifkan kredensial utama sementara	Rantai benar-benar berpindah
Kemampuan setara	Minta agen memanggil tool pada model cadangan	Cadangan mendukung tool
Gambar terbaca	Kirim lampiran gambar ke model cadangan	Cadangan mendukung visi bila Anda membutuhkannya
Kualitas cukup	Ulangi lima pertanyaan khas Anda	Jawabannya masih dapat dipakai

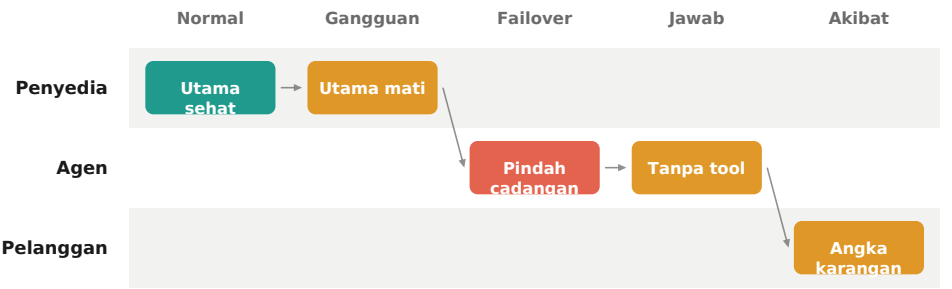
Uji kedua dan ketiga adalah yang paling sering terlewat. Rantai cadangan yang berpindah dengan mulus tetapi kehilangan kemampuan memanggil tool akan menghasilkan agen yang mulai mengarang — kegagalan yang lebih buruk daripada tidak menjawab sama sekali.

BAB 18 / STUDI KASUS

Cadangan yang menjawab tetapi mengarang

Sebuah tim menyiapkan rantai cadangan dengan model yang lebih murah. Mereka mengujinya sekali dengan mengirim satu pertanyaan sederhana, mendapat jawaban yang wajar, dan menganggap rantai itu siap.

Tiga bulan kemudian penyedia utama mengalami gangguan pada jam sibuk. Failover bekerja sempurna; agen terus menjawab. Yang tidak mereka ketahui: model cadangan tidak mendukung pemanggilan tool pada konfigurasi mereka, sehingga agen kehilangan akses ke data pesanan.



Gambar 18.3 — Failover yang berhasil secara teknis tetapi gagal secara fungsional. Tidak ada galat yang muncul di mana pun.

Selama empat jam, agen memberi angka status pesanan yang tidak berasal dari sistem mana pun. Aturan anti-mengarang yang dibahas di Bab 56 ada persis untuk kasus ini, tetapi aturan itu hanya bekerja bila ditulis ke dalam skill sejak awal.

TERSEDIA TIDAK SAMA DENGAN SETARA
Rantai cadangan harus setara kemampuannya, bukan hanya tersedia. Uji dukungan tool, dukungan gambar, dan panjang konteks pada setiap model cadangan — bukan hanya apakah ia menjawab.

BAB 18 / LATIHAN

Latihan mandiri

1. Jalankan pemeriksaan grep pada praktikum langkah dua dan pastikan tidak ada kredensial mentah di konfigurasi Anda.
2. Jalankan keempat uji rantai cadangan dan catat hasilnya. Perbaiki rantai bila ada uji yang gagal.
3. Daftarkan kemampuan yang wajib dimiliki setiap model dalam rantai Anda, berdasarkan apa yang benar-benar dipakai agen Anda.
4. Tuliskan apa yang harus terjadi ketika seluruh rantai gagal: apakah agen diam, memberi pesan, atau mengeskalasi ke manusia.

BAB 18 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Model baru tidak muncul di menu pemilihan	Sebagian katalog dibangun saat startup. Restart Gateway setelah menambah model.
Kredensial lama tidak terbaca	Format lama. Jalankan doctor --fix untuk mengimpornya ke SQLite.
setup-token gagal di skrip	Ia menuntut TTY interaktif. Jalankan dari terminal sungguhan.
Failover berpindah tetapi jawaban jadi rusak	Model cadangan tidak setara kemampuannya. Cocokkan dukungan tool dan gambar.
Bingung menaruh baseUrl	baseUrl milik models.providers, bukan milik profil auth.

Tiga pertanyaan review

1. Mengapa kredensial disimpan per agen di SQLite, bukan satu tempat global di berkas konfigurasi?
2. Apa yang salah bila seseorang menaruh apiKey di bawah models.providers?
3. Rancang rantai failover untuk agen layanan pelanggan, dan sebutkan satu kemampuan yang wajib sama di seluruh rantai.

Kunci pembahasan

Menyimpan kredensial per agen memungkinkan pemisahan kewenangan yang dibahas di

Bab 9 berlaku sampai ke tingkat penagihan dan akses penyedia, sehingga agen riset tidak dapat memakai kunci milik agen produksi. Menaruh apiKey di models.providers mencampur rahasia ke dalam berkas konfigurasi yang sering disalin, dibagikan, dan dimasukkan ke kendali versi — tempat yang justru paling buruk untuk rahasia. Untuk agen layanan pelanggan, rantai cadangan harus seluruhnya mendukung pemanggilan tool, karena tanpa itu agen kehilangan kemampuan memeriksa data dan akan mulai mengarang jawaban.

Rujukan: dokumentasi OpenClaw, halaman gateway/authentication, concepts/model-providers, dan concepts/model-failover, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

Bab 19 Logging, diagnostik, dan doctor

Ada dua jenis operator: yang membaca log ketika sesuatu rusak, dan yang menebak. Bab ini tentang menjadi yang pertama, dan tentang alat yang membuat pembacaan itu tidak memakan waktu berjam-jam.

Tujuan belajar

Menemukan log yang relevan, menjalankan doctor dengan postur yang tepat, dan menghasilkan berkas diagnostik yang aman dibagikan. Pada akhir bab, pembaca dapat mempersempit satu masalah dari gejala menjadi penyebab.

Permukaan log

Permukaan	Kapan dipakai
openclaw logs	Menarik log Gateway lewat RPC, paling cepat
Berkas log	Untuk rentang waktu yang sudah lewat
Tab Logs di Control UI	Ketika Anda sedang di dasbor
Keluaran konsol	Saat menjalankan Gateway di terminal untuk pengembangan

Tiga tingkat doctor

Gambar 19.1 — Naik satu tingkat hanya setelah tingkat sebelumnya memberi informasi. Melompat ke --fix menghilangkan bukti penyebabnya.

Urutan itu bukan formalitas. Menjalankan --fix lebih dulu sering menyelesaikan gejala sekaligus menghapus jejak yang menjelaskan mengapa gejala itu muncul — dan masalah yang sama akan kembali tanpa Anda mengetahui sebabnya.

BAB 19 / DOCTOR

Apa yang diperiksa doctor

Doctor menjalankan pemeriksaan kesehatan, migrasi konfigurasi, dan langkah perbaikan. Cakupannya luas: normalisasi konfigurasi dan migrasi kunci warisan, perbaikan penyedia dan rute, pemeriksaan layanan Gateway, pemasangan, keamanan, status workspace, autentikasi, kesehatan, dan supervisor, hingga tata letak disk, penyimpanan cron, sesi, autentikasi model, sandbox, dan pemasangan plugin.

```
openclaw doctor                # baca dulu
openclaw doctor --fix          # baru perbaiki
openclaw health                # potret kesehatan lewat RPC
openclaw status                # diagnostik dan pemakaian
openclaw triage                # diagnostik tersanitasi
```

Lima perintah yang menutup sebagian besar kebutuhan diagnosis harian.

ADA MODE YANG HANYA MEMBACA

Doctor juga memiliki mode lint yang hanya membaca, dan probe plugin pasca-pembaruan. Bila Anda ingin tahu apa yang akan diubah tanpa mengubahnya, mode itu adalah tempat yang benar untuk memulai.

BAB 19 / BERBAGI

Meminta bantuan tanpa membocorkan rahasia

Log mentah OpenClaw dapat memuat token, nomor telepon, dan potongan percakapan. Menempelkannya ke forum publik adalah kebocoran data yang sering tidak disadari pelakunya sampai berbulan-bulan kemudian.

- ✓ Memakai openclaw triage untuk diagnostik yang dibagikan
- ✓ Memeriksa berkas ekspor sebelum mengunggahnya
- ✓ Menyebut versi dari openclaw --version, bukan dari ingatan
- ⚠ Menempelkan tangkapan layar log mentah ke kanal publik
- ⚠ Membagikan isi openclaw.json tanpa menyunting rahasianya

Gambar 19.2 — Lima kebiasaan saat meminta bantuan. Dua yang terakhir adalah cara tercepat membocorkan kredensial produksi.

BAB 19 / PRAKTIKUM

Menelusuri satu masalah dari gejala ke penyebab

Praktikum ini melatih urutan diagnosis. Urutannya penting karena melompat ke perbaikan sering menghapus bukti yang menjelaskan penyebabnya.

Langkah 1 — potret keadaan sebelum menyentuh apa pun

```
openclaw status  > /tmp/diag-status.txt
openclaw health  > /tmp/diag-health.txt
openclaw doctor  > /tmp/diag-doctor.txt
```

Tiga berkas ini adalah bukti. Simpan sebelum menjalankan perbaikan apa pun.

Langkah 2 — persempit waktunya

```
openclaw logs --follow
# atau untuk rentang yang sudah lewat, baca berkas lognya
```

Cari peristiwa terakhir yang normal, lalu baca apa yang terjadi sesudahnya.

Teknik yang paling berguna: jangan mulai dari galat. Mulai dari peristiwa terakhir yang jelas normal, lalu maju selangkah demi selangkah. Galat sering muncul beberapa langkah setelah penyebab sebenarnya.

Langkah 3 — baru kemudian perbaiki

Tingkat	Perintah	Kapan
Baca	`openclaw doctor`	Selalu, lebih dulu
Baca terbatas	`Mode lint yang hanya membaca`	Bila ingin tahu dampaknya
Perbaiki	`openclaw doctor --fix`	Setelah bukti tersimpan
Serahkan	`openclaw triage`	Bila butuh bantuan pihak lain

Langkah 4 — tuliskan apa yang Anda pelajari

```
# Catatan insiden
Gejala      :
Waktu mulai :
Peristiwa normal terakhir :
Penyebab    :
Perbaikan   :
Pencegahan  :
```

Baris terakhir adalah yang mengubah insiden menjadi perbaikan permanen.

BAB 19 / STUDI KASUS

Perbaikan yang menghapus penyebabnya

Sebuah tim menemukan agen berhenti menjawab di satu kanal. Orang pertama yang menanganinya langsung menjalankan perbaikan otomatis. Masalahnya selesai dalam dua menit, dan semua orang kembali bekerja.

Sepuluh hari kemudian gejala yang sama muncul. Ditangani dengan cara yang sama, selesai lagi. Pada kejadian keempat, seseorang bertanya mengapa ini terus berulang — dan tidak ada seorang pun yang memiliki informasi untuk menjawabnya, karena setiap kali bukti dihapus bersama masalahnya.

Urutan yang dipakai	Apa yang hilang
Langsung perbaiki	Keadaan sebelum perbaikan
Tidak mencatat gejala	Pola kemunculan
Tidak membaca log	Peristiwa yang mendahuluinya
Tidak menulis catatan	Kesempatan mengenali pola pada kejadian kedua

Pada kejadian kelima mereka mengubah urutan: potret dulu, baca log dulu, baru perbaiki. Penyebabnya ditemukan dalam dua puluh menit — sebuah kredensial yang kedaluwarsa berkala dan tidak diperbarui otomatis. Perbaikannya permanen.

> **ATURAN KEJADIAN KEDUA**
> Perbaikan yang cepat terasa efisien dan sering kali justru mahal. Aturan sederhana: bila sebuah gejala muncul untuk kedua kalinya, berhenti memperbaiki dan mulai mendiagnosis.

BAB 19 / LATIHAN

Latihan mandiri

1. Jalankan ketiga perintah potret dan simpan hasilnya. Ini menjadi garis dasar untuk membandingkan ketika sesuatu berubah.
2. Picu satu masalah dengan sengaja — misalnya konfigurasi kanal yang salah — lalu telusuri dari gejala ke penyebab tanpa memperbaiki apa pun.
3. Buat templat catatan insiden untuk tim Anda dan isi satu kali untuk masalah nyata terakhir yang Anda tangani.
4. Baca keluaran triage Anda dan tandai informasi apa yang sudah disanitasi dan apa yang tetap terlihat.

BAB 19 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Tidak tahu harus mulai dari mana	openclaw status lalu openclaw health; keduanya memberi gambaran cepat.
Masalah hilang lalu kembali	Kemungkinan --fix dijalankan sebelum penyebabnya dicatat. Baca dulu lain kali.
Log terlalu banyak untuk dibaca	Persempit dengan rentang waktu, lalu cari peristiwa di sekitar gejalanya.
Perlu berbagi bukti ke pihak ketiga	openclaw triage; jangan tangkapan layar log mentah.
Ingin tahu dampak perbaikan sebelum menerapkannya	Pakai mode lint yang hanya membaca.

Tiga pertanyaan review

1. Mengapa menjalankan doctor --fix sebagai langkah pertama sering merugikan dalam jangka panjang?
2. Apa yang membedakan triage dari sekadar menyalin log, dan mengapa perbedaan itu penting bagi perusahaan?
3. Rancang prosedur tiga langkah untuk insiden 'agen tidak menjawab di satu kanal'.

Kunci pembahasan

Menjalankan --fix lebih dulu dapat menyembuhkan gejala sambil menghapus jejak penyebabnya, sehingga masalah yang sama kembali tanpa Anda pernah memahami akarnya. Triage menyanitasi keluarannya, sehingga token, nomor telepon, dan isi percakapan tidak ikut keluar; bagi perusahaan, perbedaan ini adalah selisih antara laporan bug dan insiden kebocoran data. Prosedur tiga langkah: pertama pastikan gejalanya benar

dengan openclaw status dan health, kedua persempit ke kanal itu saja dengan memeriksa status dan kredensial kanalnya, ketiga baca log di sekitar waktu pesan terakhir yang berhasil sebelum menyentuh perbaikan apa pun.

Rujukan: dokumentasi OpenClaw, halaman logging, cli/doctor, cli/status, dan cli/triage, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

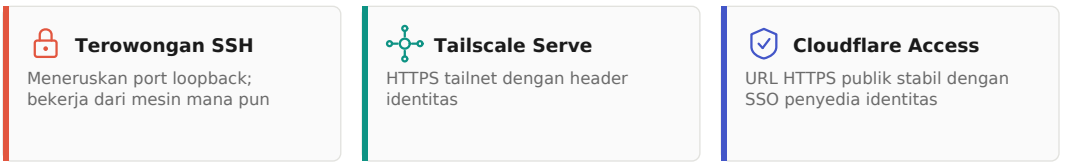
Bab 20 Akses jarak jauh: SSH, Tailscale, dan Cloudflare

Gateway terikat loopback secara bawaan, dan itu keputusan yang benar. Bab ini membahas tiga cara menjangkaunya dari luar tanpa membatalkan keputusan tersebut.

Tujuan belajar

Memilih topologi akses jarak jauh yang sesuai, menyiapkan terowongan yang bertahan melewati reboot, dan memahami mengapa terowongan tidak menggantikan autentikasi. Pada akhir bab, pembaca dapat mengakses Gateway dari mana saja dengan aman.

Tiga topologi



Gambar 20.1 — Tiga jalur yang didukung. Ketiganya mempertahankan Gateway pada loopback; yang berbeda hanyalah siapa yang menjaga pintunya.

TEROWONGAN BUKAN AUTENTIKASI

Terowongan SSH tidak melewati autentikasi Gateway. Untuk penyiapan rahasia bersama, klien tetap harus mengirim token atau kata sandi meski sudah lewat terowongan. Untuk mode yang membawa identitas, permintaan tetap harus memenuhi jalur autentikasi itu.

BAB 20 / TAILSCALE

Serve, Funnel, dan perbedaan yang menentukan

Mode	Jangkauan	Catatan
serve	Hanya tailnet	Gateway tetap di 127.0.0.1; menyuntik header identitas
funnel	Publik	OpenClaw menuntut kata sandi bersama
off	Tidak ada otomasi	Bawaannya

```
{
  gateway: {
    bind: "loopback",
    tailscale: { mode: "serve" },
  },
}
```

Tailnet saja: Gateway tetap loopback, Tailscale menyediakan HTTPS dan identitas.

```
{
  gateway: {
    bind: "tailnet",
    auth: { mode: "token", token: "rahasia-anda" },
  },
}
```

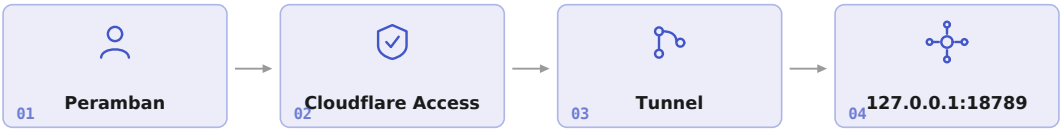
Alternatif: mendengarkan langsung pada IP tailnet, tanpa Serve maupun Funnel.

OpenClaw memegang Serve dan Funnel sebagai klaim Tailscale di latar depan. Startup Gateway berhasil hanya setelah klaim itu aktif, dan menghentikan atau kehilangan Gateway melepaskannya secara otomatis. Layanan Tailscale bernama tidak didukung oleh ingress terkelola, karena Tailscale menuntut mereka berjalan sebagai rute latar yang persisten.

BAB 20 / CLOUDFLARE

URL publik yang stabil

Jalankan Gateway pada loopback, terbitkan ia lewat Cloudflare Tunnel, dan biarkan Cloudflare Access mengautentikasi setiap permintaan sebelum mencapai OpenClaw. Gateway mempertahankan bind loopback, sehingga tidak ada port yang terekspos dan tidak ada aturan firewall masuk yang diperlukan; cloudflared menghubungi keluar dari host itu.



Gambar 20.2 — Jalur Cloudflare. Identitas diverifikasi sebelum lalu lintas pernah menyentuh Gateway.

Pilih topologi ini ketika Anda menginginkan URL HTTPS publik yang stabil beserta SSO penyedia identitas di depan Control UI. Ia memakai autentikasi trusted-proxy, jadi pahami lebih dulu apa yang Anda percayakan kepada proxy tersebut.

Untuk klien macOS

Untuk klien macOS, penyiapan persisten yang paling mudah memakai entri konfigurasi LocalForward pada SSH ditambah sebuah LaunchAgent yang menjaga terowongan tetap hidup melewati reboot dan crash. Tanpa LaunchAgent, terowongan Anda akan mati pada reboot pertama dan Anda akan mengira Gateway yang bermasalah.

BAB 20 / PRAKTIKUM

Membangun akses jarak jauh yang bertahan

Praktikum ini membangun akses dari luar kantor yang masih berfungsi setelah reboot, kehilangan jaringan, dan pergantian alamat IP — tiga hal yang mematikan sebagian besar penyiapan buatan sendiri.

Langkah 1 — pastikan Gateway tetap loopback

```
openclaw config get gateway.bind
```

Nilainya harus loopback. Seluruh langkah berikutnya menambah lapisan di depannya, bukan menggantikannya.

Langkah 2 — pilih satu topologi dan tuliskan alasannya

Topologi	Pilih bila	Yang harus Anda siapkan
Terowongan SSH	Anda butuh sesuatu yang bekerja hari ini	LaunchAgent supaya bertahan melewati reboot
Tailscale Serve	Beberapa orang butuh akses tetap	Tailnet dengan keanggotaan yang dikurasi
Cloudflare Access	Anda butuh URL publik stabil dengan SSO	Domain, tunnel, dan pemahaman trusted-proxy

Langkah 3 — uji ketahanannya

```
# Setelah penyiapan selesai, uji ketiganya:
1. Reboot Mac klien, lalu coba sambung tanpa menyentuh apa pun
2. Matikan Wi-Fi selama satu menit, nyalakan, coba sambung lagi
3. Pindah ke jaringan lain, coba sambung lagi
```

Penyiapan yang gagal pada uji pertama adalah penyiapan yang akan mengecewakan Anda pada hari Senin pagi.

Langkah 4 — buktikan autentikasi masih berlaku

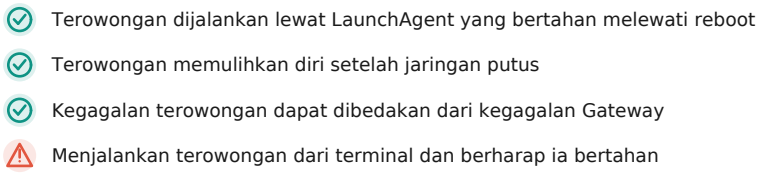
Terowongan tidak menggantikan autentikasi. Pastikan klien Anda tetap harus mengirim kredensial meski sudah berada di dalam terowongan. Bila ia tidak diminta apa pun, periksa ulang mode autentikasi Anda — bukan merasa lega.

BAB 20 / STUDI KASUS

Terowongan yang mati setiap Senin

Seorang operator menyiapkan terowongan SSH dari laptopnya ke Gateway kantor. Ia menjalankannya dari terminal, memverifikasi berfungsi, dan menutup laptop pada Jumat sore.

Senin pagi terowongan mati. Ia menjalankannya lagi. Selasa mati lagi setelah laptop tertidur. Selama tiga minggu ia mengulang perintah yang sama setiap pagi, dan menyimpulkan bahwa Gateway-nya tidak stabil.



Gambar 20.3 — Empat butir. Butir ketiga adalah yang membuat operator ini menyalahkan komponen yang salah selama tiga minggu.

Penyebab sebenarnya tidak pernah berada di Gateway. Gateway berjalan sempurna sepanjang waktu; yang mati adalah proses terowongan di laptopnya. Tiga minggu diagnosis diarahkan ke komponen yang tidak pernah bermasalah.

> **PERTANYAAN PERTAMA: SISI MANA**
> Ketika akses jarak jauh gagal, tentukan lebih dulu sisi mana yang gagal. Periksa Gateway dari mesin Gateway itu sendiri sebelum mendiagnosis apa pun dari jarak jauh.

BAB 20 / LATIHAN

Latihan mandiri

1. Pilih satu topologi akses jarak jauh, siapkan, lalu jalankan ketiga uji ketahanan pada praktikum langkah tiga.
2. Buktikan bahwa autentikasi masih berlaku di dalam terowongan Anda dengan mencoba menyambung tanpa kredensial.
3. Tuliskan cara membedakan kegagalan terowongan dari kegagalan Gateway dalam dua langkah pemeriksaan.
4. Bila Anda memakai Tailscale, daftarkan seluruh anggota tailnet dan putuskan apakah allowTailscale pantas menyala pada penyiapan Anda.

BAB 20 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Klien ditolak meski terowongan hidup	Terowongan tidak menggantikan autentikasi. Kirim token atau kata sandi.
Akses hilang setelah Mac reboot	Terowongan SSH tidak persisten. Pasang LaunchAgent yang menjaganya.
Gateway gagal start saat Tailscale bermasalah	Startup berhasil hanya setelah klaim Serve atau Funnel aktif.
Funnel menolak berjalan tanpa kata sandi	Perilaku yang benar. Funnel publik menuntut kata sandi bersama.
Rute HTTPS lama menghalangi	Jalankan openclaw doctor untuk memeriksa; ia tidak mengubah rute milik pihak lain.

Tiga pertanyaan review

1. Mengapa mempertahankan bind loopback tetap disarankan meski Anda sudah memakai Tailscale atau Cloudflare?
2. Kapan Funnel pantas dipilih dibanding Serve, dan syarat tambahan apa yang menyertainya?
3. Rancang akses untuk konsultan luar yang butuh akses sementara selama dua minggu.

Kunci pembahasan

Bind loopback tetap disarankan karena ia membuat port tidak pernah terekspos ke jaringan sama sekali; Tailscale dan Cloudflare hanya menambah lapisan di depan, dan bila lapisan itu salah konfigurasi, loopback adalah jaring pengaman terakhir. Funnel pantas hanya ketika Anda benar-benar butuh URL publik yang dapat dijangkau tanpa tailnet, dan syarat tambahannya adalah kata sandi bersama yang dituntut OpenClaw. Untuk konsultan sementara, jalur paling bersih adalah mengundangnya ke tailnet dengan masa berlaku terbatas, bukan membuka Funnel publik yang harus diingat untuk ditutup.

Rujukan: dokumentasi OpenClaw, halaman gateway/remote, gateway/tailscale, dan gateway/cloudflare-access, ditinjau 17 September 2026.

GATEWAY DAN KONFIGURASI

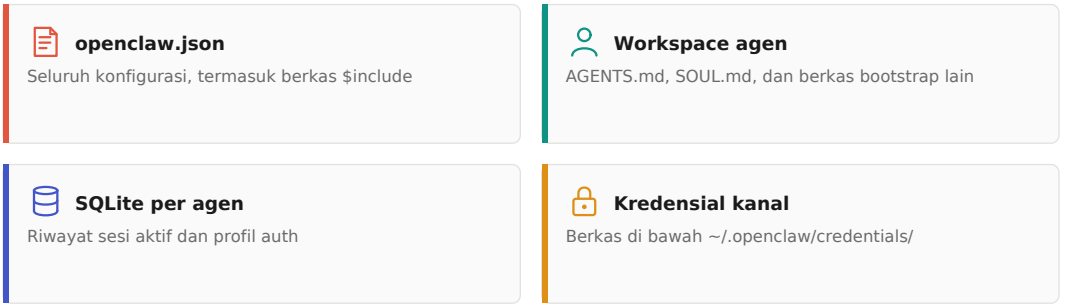
Bab 21 Cadangan, pemulihan, dan pemindahan mesin

Cadangan yang tidak pernah diuji bukan cadangan; ia hanya berkas yang membuat Anda merasa tenang. Bab ini membahas apa yang benar-benar perlu dicadangkan, dan bagaimana memindahkan seluruh pemasangan ke mesin lain tanpa kehilangan ingatan agen.

Tujuan belajar

Menyebutkan seluruh keadaan yang harus ikut pindah, menjalankan cadangan, dan memverifikasi bahwa pemulihan benar-benar bekerja. Pada akhir bab, pembaca memiliki rencana pemulihan yang sudah pernah dicoba.

Apa yang harus ikut



Gambar 21.1 — Empat kelompok keadaan. Melewatkan yang ketiga membuat agen kehilangan seluruh ingatan percakapannya.

```
~/.openclaw/openclaw.json
~/.openclaw/workspace/
~/.openclaw/agents/<agentId>/agent/openclaw-agent.sqlite
~/.openclaw/credentials/
```

Empat jalur yang menampung hampir seluruh keadaan yang berarti.

SINKRONISASI AWAN BUKAN CADANGAN
Jangan meletakkan direktori ini di dalam folder yang tersinkronisasi awan seperti iCloud Drive atau Dropbox. Basis data SQLite yang disinkronkan layanan berkas menghasilkan kerusakan yang sulit didiagnosis, dan sinkronisasi bukan pengganti cadangan.

BAB 21 / MENJALANKAN

Perintah cadangan bawaan

OpenClaw menyediakan perintah cadangan yang menangani arsip, snapshot SQLite, dan riwayat Git. Snapshot SQLite penting karena menyalin berkas basis data yang sedang dipakai dengan cp biasa dapat menghasilkan salinan yang tidak konsisten.

```
openclaw backup

# workspace sebagai repositori git
cd ~/.openclaw/workspace && git log --oneline -5
```

Cadangan bawaan plus riwayat git workspace yang dibahas di Bab 7.

Menguji pemulihan

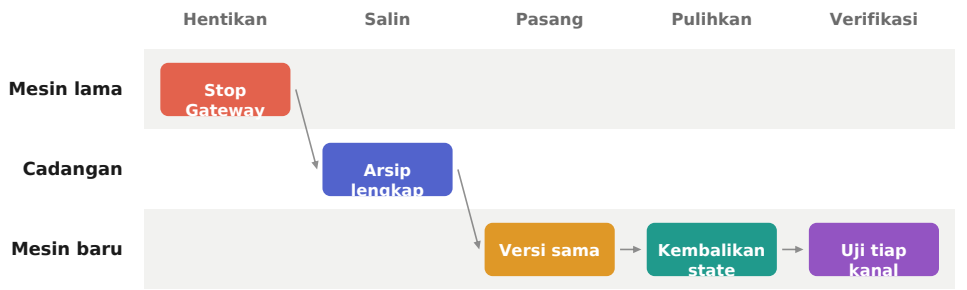
- Cadangan dipulihkan ke mesin uji, bukan hanya diperiksa ukurannya
- Agen dijalankan dan ditanyai sesuatu dari riwayat lama
- Satu kanal disambungkan ulang dan diuji dengan pesan sungguhan
- Waktu pemulihan dicatat, supaya ekspektasi saat insiden realistis
- Menganggap cadangan berhasil karena perintahnya tidak menampilkan galat

Gambar 21.2 — Lima langkah uji pemulihan. Yang terakhir adalah keyakinan palsu yang paling umum di antara operator berpengalaman sekalipun.

BAB 21 / PINDAH MESIN

Memindahkan seluruh pemasangan

Pemindahan mesin adalah pemulihan yang direncanakan. Urutannya penting, terutama pada kanal yang sesinya terikat ke perangkat — WhatsApp dan iMessage adalah contoh yang paling sering menimbulkan kejutan.



Gambar 21.3 — Urutan pemindahan. Menghentikan Gateway lebih dulu menghindari menyalin basis data yang sedang ditulis.

Pasang versi yang sama di mesin baru sebelum memulihkan keadaan. Memulihkan keadaan lama ke versi yang jauh lebih baru memaksa migrasi berjalan di saat Anda sedang terburu-buru — waktu terburuk untuk menemukan bahwa sebuah migrasi butuh campur tangan.

BAB 21 / PRAKTIKUM

Membuktikan cadangan Anda benar-benar bekerja

Praktikum ini tidak selesai dengan membuat cadangan. Ia selesai ketika cadangan itu sudah dipulihkan ke mesin lain dan agen di sana menjawab pertanyaan dari riwayat lama.

Langkah 1 — catat apa yang harus ikut

```
du -sh ~/.openclaw/openclaw.json
du -sh ~/.openclaw/workspace
du -sh ~/.openclaw/agents
du -sh ~/.openclaw/credentials
```

Ukuran memberi gambaran; yang penting adalah keempatnya ikut, bukan besarnya.

Langkah 2 — hentikan sebelum menyalin

```
openclaw gateway stop
openclaw backup
openclaw gateway restart
```

Menyalin SQLite yang sedang ditulis menghasilkan berkas yang terlihat utuh tetapi gagal dibuka.

Langkah 3 — pulihkan ke mesin uji

Tahap	Verifikasi	Bila gagal
Versi disamakan	openclaw --version cocok dengan sumber	Pasang versi yang sama dulu
State dikembalikan	Keempat direktori ada dan ukurannya wajar	Periksa arsip cadangan
Gateway naik	openclaw health sehat	Baca log startup
Riwayat terbaca	Tanyakan sesuatu dari percakapan lama	SQLite kemungkinan tidak ikut
Kanal	Sambungkan satu kanal dan kirim pesan uji	Sebagian sesi kanal terikat perangkat

Langkah 4 — catat berapa lama seluruhnya memakan waktu

Angka ini yang akan ditanyakan manajemen saat insiden: berapa lama sampai pulih. Jawaban “saya belum pernah mencobanya” adalah jawaban yang mahal. Catat waktunya sekarang, saat tidak ada tekanan.

BAB 21 / STUDI KASUS

Cadangan yang ada selama dua tahun

Sebuah perusahaan menjalankan cadangan otomatis setiap malam selama dua tahun. Berkasnya ada, ukurannya wajar, dan tidak ada satu pun pekerjaan cadangan yang gagal. Semua orang merasa tenang.

Ketika mesin Gateway rusak, mereka memulihkan cadangan ke mesin baru. Konfigurasi kembali, workspace kembali, kanal tersambung. Yang tidak kembali adalah seluruh riwayat percakapan: skrip cadangan hanya menyalin direktori yang mereka anggap penting dua tahun lalu, dan basis data per agen tidak termasuk.

YANG DICADANGKAN		YANG DIBUTUHKAN
Ya	**openclaw.json**	Ya
Ya	**workspace**	Ya
Ya	**credentials**	Ya
Tidak	**SQLite per agen**	Ya
Tidak	**Pernah diuji pulih**	Wajib

Gambar 21.4 — Empat dari lima baris benar. Baris keempat yang hilang menghapus dua tahun riwayat.

Yang membuat kasus ini pedih: satu kali uji pemulihan pada tahun pertama akan menemukannya dalam tiga puluh menit. Cadangan yang tidak pernah dipulihkan bukan cadangan; ia hanya berkas yang membuat Anda merasa tenang.

> **SATU ANGKA YANG LAYAK DIPAJANG**
> Jadwalkan uji pemulihan tahunan ke mesin uji, dan jadikan keberhasilannya sebagai syarat — bukan sekadar kegiatan. Tuliskan tanggal uji terakhir di tempat yang terlihat oleh orang yang bertanggung jawab.

BAB 21 / LATIHAN

Latihan mandiri

1. Jalankan cadangan lengkap, lalu pulihkan ke mesin uji dan jalankan kelima verifikasi pada praktikum langkah tiga.
2. Catat waktu total pemulihan dan bandingkan dengan ekspektasi yang pernah Anda sampaikan kepada manajemen.
3. Periksa skrip cadangan Anda yang sudah ada dan pastikan basis data per agen benar-benar termasuk.
4. Tetapkan tanggal uji pemulihan berikutnya dan siapa yang menjalankannya.

BAB 21 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Riwayat percakapan hilang setelah pindah	SQLite per agen tidak ikut. Ia bukan berada di folder sessions.
Basis data rusak tanpa sebab jelas	Direktori berada di folder tersinkronisasi awan. Pindahkan keluar.
Kanal minta autentikasi ulang di mesin baru	Sebagian sesi kanal terikat perangkat. Rencanakan penyambungan ulang.
Pemulihan memicu migrasi tak terduga	Versi mesin baru lebih tinggi. Samakan versi lebih dulu, baru perbarui.
Cadangan ada tetapi tidak pernah dicoba	Itu bukan cadangan. Jadwalkan uji pemulihan berkala.

Tiga pertanyaan review

1. Mengapa Gateway harus dihentikan sebelum menyalin keadaannya, dan apa yang mungkin terjadi bila tidak?
2. Mengapa memasang versi yang sama di mesin baru lebih aman daripada langsung memasang versi terbaru?
3. Rancang jadwal cadangan untuk agen layanan pelanggan yang melayani seratus pesan per hari, dan sebutkan satu hal yang harus diuji setiap bulan.

Kunci pembahasan

Gateway harus dihentikan karena basis data SQLite yang sedang ditulis dapat tersalin dalam keadaan setengah transaksi, menghasilkan berkas yang terlihat utuh tetapi gagal dibuka saat dipulihkan. Memasang versi yang sama memisahkan dua risiko — pemindahan dan migrasi — sehingga bila ada masalah Anda tahu mana penyebabnya. Untuk agen CS seratus pesan per hari, cadangan harian otomatis sudah memadai dengan retensi beberapa minggu, dan yang wajib diuji setiap bulan adalah pemulihan penuh ke mesin uji sampai agen benar-benar menjawab pertanyaan dari riwayat lama.

Rujukan: dokumentasi OpenClaw, halaman cli/backup, install/backups, dan install/migrating, ditinjau 17 September 2026. Sebagian saran operasional adalah praktik penyusun, bukan kutipan dokumentasi.

KANAL PERCAKAPAN

Bab 22 Peta kanal dan kemampuan per platform

OpenClaw menyambung ke banyak platform percakapan. Daftarnya panjang, dan panjangnya itu menipu: kanal yang tersedia tidak berarti kanal yang setara. Bab ini memetakan mana yang benar-benar sepadan untuk pekerjaan Anda.

Tujuan belajar

Memilih kanal berdasarkan kemampuan dan bukan popularitas, memahami konsep bersama yang berlaku di semua kanal, dan mengetahui di mana perbedaan antar platform akan menggigit. Pada akhir bab, pembaca dapat menyusun daftar kanal untuk kasusnya sendiri.

Kanal yang didukung

Gateway swakelola ini menyambungkan Discord, Google Chat, iMessage, Matrix, Microsoft Teams, Signal, Slack, Telegram, WhatsApp, Zalo, dan lainnya ke agen AI. Sebagian dibangun ke dalam inti, sebagian datang sebagai plugin resmi, dan sebagian lagi dikelola komunitas.

Gambar 22.1 — Enam kanal yang paling relevan untuk pembaca buku ini. Pilihannya lebih banyak, tetapi keenam inilah yang menutup sebagian besar kebutuhan nyata.

BAB 22 / KONSEP BERSAMA

Yang berlaku di semua kanal

Sebelum membahas perbedaan, ada baiknya menetapkan yang sama. Konsep berikut muncul di setiap kanal, hanya dengan nama kunci konfigurasi yang berbeda.

Konsep	Perannya
dmPolicy	Siapa yang boleh mengirim pesan langsung
allowFrom	Daftar izin DM yang eksplisit
groupPolicy dan groupAllowFrom	Perlakuan terhadap obrolan grup
Gerbang mention	Apakah bot menjawab tanpa dipanggil di grup
Pairing	Persetujuan eksplisit untuk pengirim yang belum dikenal
contextVisibility	Apa yang boleh dilihat model dari konteks sekitar
Access group	Daftar izin yang dapat dipakai ulang antar kanal

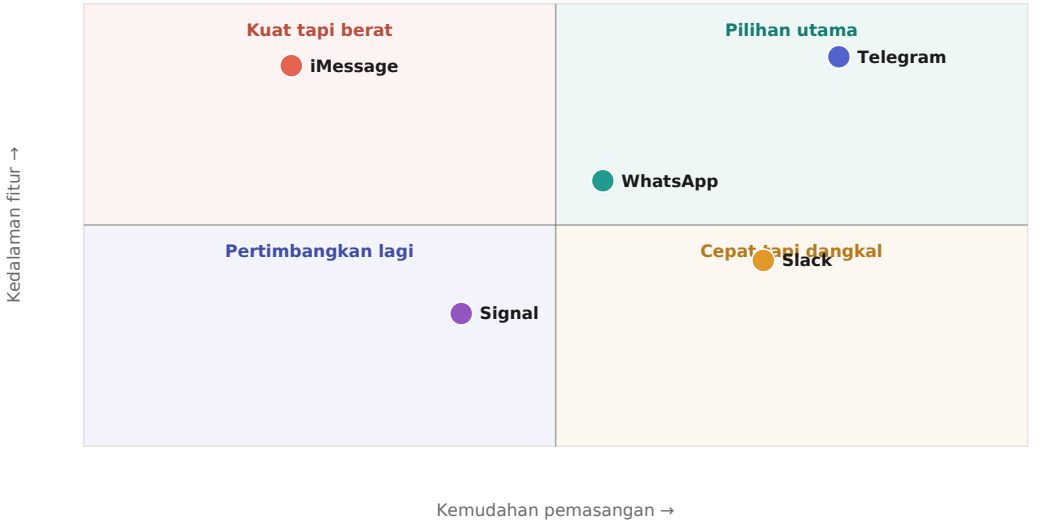
Access group adalah yang paling sering terlewat. Ia memungkinkan satu daftar pengirim dipakai ulang oleh beberapa kanal sekaligus, sehingga menambah seorang karyawan baru tidak menuntut penyuntingan lima tempat berbeda. Entri access group yang dirujuk allowlist kanal diselesaikan otomatis.

Perutean ke agen

Setiap pesan masuk harus dipetakan ke satu agen dan satu sesi. Bab 9 sudah membahas pengikatan rute; di sini cukup diingat bahwa kanal adalah salah satu dimensi pengikatan itu, dan satu kanal dapat dipecah lebih jauh menjadi akun, grup, atau topik tertentu.

BAB 22 / MEMILIH

Memilih kanal untuk pekerjaan nyata



Gambar 22.2 — Lima kanal pada dua sumbu. Penilaian ini relatif dan ditujukan untuk membantu memilih, bukan untuk menilai kualitas platformnya.

iMessage berada di kiri atas karena ia menuntut Mac yang selalu hidup dan, untuk fitur penuh, penonaktifan System Integrity Protection — syarat yang dibahas di Bab 24 dan tidak boleh diambil ringan. Telegram berada di kanan atas karena pemasangannya paling mudah sekaligus paling kaya fitur bot.

JANGKAUAN MENGALAHKAN KENYAMANAN

Untuk melayani pelanggan di Indonesia, WhatsApp hampir selalu menjadi jawaban meskipun ia bukan yang paling mudah dipasang. Kanal terbaik adalah kanal yang sudah dipakai pelanggan Anda, bukan yang paling nyaman bagi tim teknis.

BAB 22 / PRAKTIKUM

Memilih kanal berdasarkan bukti

Praktikum ini mengganti perdebatan kanal dengan data. Kerjakan sebelum menyambungkan apa pun, karena kanal yang sudah tersambung sulit dicabut tanpa membingungkan orang yang sudah memakainya.

Langkah 1 — hitung di mana orang Anda sudah berada

Pertanyaan	Cara mendapatkan angkanya
Kanal apa yang dipakai pelanggan menghubungi Anda sekarang	Hitung dari riwayat satu bulan terakhir
Berapa persen di luar jam kerja	Hitung dari stempel waktu
Kanal apa yang dipakai tim internal	Tanyakan; jangan diasumsikan
Adakah kanal yang diwajibkan kepatuhan	Tanyakan ke bagian hukum

Langkah 2 — petakan kebutuhan ke kemampuan kanal

Untuk tiap kanal kandidat, buka halaman dokumentasinya dan periksa apakah kemampuan yang Anda butuhkan benar-benar ada di sana. Jangan mengandalkan asumsi; kemampuan berbeda cukup jauh antar kanal.

Langkah 3 — pahami konsep bersama sebelum menyentuh kanal apa pun

```
# Tujuh konsep yang muncul di setiap kanal:
dmPolicy            # siapa yang boleh mengirim pesan langsung
allowFrom           # daftar izin DM eksplisit
groupPolicy         # perlakuan terhadap obrolan grup
groupAllowFrom      # daftar izin pengirim di grup
requireMention      # apakah bot menjawab tanpa dipanggil
contextVisibility   # apa yang boleh dilihat model
access group        # daftar izin yang dipakai ulang antar kanal
```

Menguasai tujuh ini membuat kanal kelima Anda jauh lebih cepat dipasang daripada kanal pertama.

Langkah 4 — tuliskan keputusan kanal Anda

```
# Contoh isian
Pelanggan : WhatsApp   (91% kontak masuk sudah lewat sini)
Internal  : Slack      (tim sudah bekerja di sana)
Operasi   : Telegram   (butuh topik untuk tingkat kewenangan)
Tidak     : iMessage   (tidak ada Mac khusus; lihat Bab 23)
```

Baris terakhir sama pentingnya: mencatat kanal yang sengaja tidak dipakai mencegah perdebatan yang sama berulang.

BAB 22 / STUDI KASUS

Kanal yang dipilih karena paling mudah

Sebuah tim teknis memilih Telegram untuk layanan pelanggan. Alasannya jujur: pemasangannya paling mudah, tokennya didapat dalam dua menit, dan tidak perlu URL publik. Dalam satu sore, bot mereka sudah berjalan.

Enam minggu kemudian, jumlah pelanggan yang memakainya berjumlah sebelas orang. Sisanya tetap mengirim WhatsApp ke nomor lama, karena di sanalah mereka sudah berada. Tim menghabiskan enam minggu membangun sesuatu yang benar secara teknis untuk kanal yang salah.

DIPILIH TIM TEKNIS		DIPAKAI PELANGGAN
Telegram	**Kanal**	WhatsApp
Satu sore	**Waktu pemasangan**	Beberapa hari
11 orang	**Adopsi setelah 6 minggu**	Seluruhnya
Baik	**Kualitas teknis**	Baik
Nyaris nol	**Hasil bisnis**	Yang diharapkan

Gambar 22.3 — Kualitas teknis yang setara dengan hasil bisnis yang sangat berbeda. Perbedaannya hanya pada di mana orang sudah berada.

Mereka kemudian memindahkannya ke WhatsApp dan adopsi terjadi dalam dua minggu tanpa kampanye apa pun. Pekerjaan teknisnya sebagian besar dapat dipakai ulang; yang hilang adalah enam minggu.

ATURAN PEMILIHAN KANAL

Kanal terbaik adalah kanal yang sudah dipakai pengguna Anda, bukan yang paling nyaman bagi tim teknis. Kemudahan pemasangan adalah biaya sekali; salah kanal adalah biaya berulang.

BAB 22 / LATIHAN

Latihan mandiri

1. Hitung keempat angka pada praktikum langkah satu dari data nyata organisasi Anda.
2. Tuliskan keputusan kanal Anda beserta kanal yang sengaja tidak dipakai dan alasannya.
3. Untuk kanal utama pilihan Anda, buka halaman dokumentasinya dan periksa apakah setiap kemampuan yang Anda rencanakan benar-benar tersedia.
4. Susun satu access group yang dapat dipakai ulang di semua kanal Anda, dan tetapkan siapa yang boleh menambah anggotanya.

BAB 22 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Menambah orang baru menuntut sunting banyak tempat	Pakai access group supaya satu daftar dipakai ulang antar kanal.
Bot menjawab semua obrolan grup	Gerbang mention mati atau groupPolicy terlalu terbuka.
Fitur yang ada di satu kanal tidak ada di kanal lain	Normal. Periksa halaman kanal itu sebelum menjanjikannya ke pengguna.
Pesan jatuh ke agen yang salah	Pengikatan rute belum menyebut kanal atau akun yang benar.
Memilih kanal berdasarkan kemudahan teknis	Periksa dulu kanal apa yang benar-benar dipakai pengguna Anda.

Tiga pertanyaan review

1. Mengapa daftar kanal yang panjang tidak berarti kebebasan memilih yang sesungguhnya?
2. Apa keuntungan access group dibanding menuliskan allowlist terpisah di tiap kanal?
3. Perusahaan Anda melayani pelanggan ritel di Indonesia dan tim internal yang memakai Slack. Rancang pembagian kanalnya dan sebutkan alasan tiap pilihan.

Kunci pembahasan

Daftar panjang menipu karena kemampuan tiap kanal berbeda, dan sebagian kanal menuntut prasyarat berat seperti Mac yang selalu hidup atau SIP yang dimatikan, sehingga pilihan sesungguhnya jauh lebih sempit daripada daftarnya. Access group memusatkan daftar izin sehingga satu perubahan keanggotaan berlaku di semua kanal sekaligus, mengurangi kemungkinan satu kanal tertinggal dan menjadi celah. Untuk perusahaan ritel Indonesia, WhatsApp untuk pelanggan karena di sanalah mereka berada, Slack untuk tim internal karena percakapan kerja sudah hidup di sana, dan keduanya diikat ke agen yang berbeda supaya kewenangan internal tidak pernah menyentuh kanal pelanggan.

Rujukan: dokumentasi OpenClaw, halaman channels, channels/access-groups, dan gateway/security/access-control, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 23 iMessage lewat imsg: pemasangan di Mac

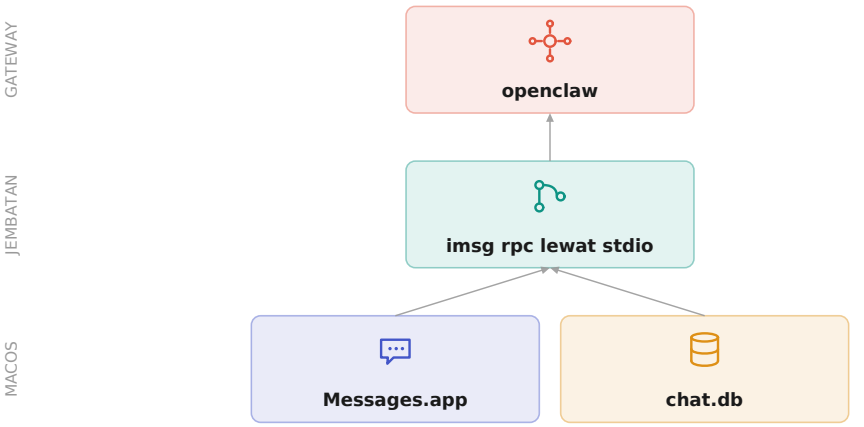
iMessage adalah alasan terkuat mengapa buku ini hanya membahas macOS. Tidak ada sistem operasi lain yang dapat menjadi tuan rumahnya. Bab ini memasang kanal itu dari nol, dalam mode dasar yang tidak menuntut pengorbanan keamanan apa pun.

Tujuan belajar

Memasang plugin dan biner imsg, memberikan izin macOS yang tepat pada konteks proses yang benar, dan menyetujui pemasangan DM pertama. Pada akhir bab, pembaca memiliki kanal iMessage yang berfungsi untuk teks dan media.

Bagaimana ia bekerja

Statusnya adalah integrasi CLI eksternal native. Gateway memunculkan imsg rpc dan berbicara JSON-RPC lewat stdio — tidak ada daemon atau port terpisah. OpenClaw membaca dan mengamati Messages melalui imsg pada Mac tempat Messages.app masuk.



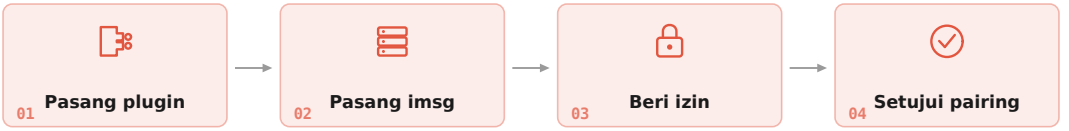
Gambar 23.1 — Tidak ada server HTTP, webhook, atau kata sandi REST di jalur ini. Hanya satu proses anak yang berbicara JSON-RPC.

BLUEBUBBLES SUDAH DIHAPUS

Jalur iMessage yang didukung tidak memiliki server HTTP BlueBubbles, rute webhook, kata sandi REST, maupun runtime plugin BlueBubbles. Bila Anda menemukan panduan yang menyebut hal-hal itu, panduan tersebut sudah usang.

BAB 23 / PEMASANGAN

Empat langkah



Gambar 23.2 — Empat langkah mode dasar. Langkah ketiga adalah yang paling sering gagal, dan penyebabnya hampir selalu konteks proses.

Pasang plugin iMessage resmi pada host Gateway, lalu restart Gateway:

```
openclaw plugins install @openclaw/imessage
```

Plugin dipasang di host Gateway, bukan di Mac Messages bila keduanya berbeda.

Pasang dan verifikasi imsg pada Mac tempat Messages masuk:

```
brew install steipete/tap/imsg
imsg --version
imsg chats --limit 3
imsg rpc --help
```

Verifikasi imsg lebih dulu; bila perintah ini gagal, OpenClaw pasti ikut gagal.

Untuk penyiapan lokal yang umum, penyiapan OpenClaw dapat menawarkan pemasangan atau pembaruan Homebrew untuk imsg yang dikonfirmasi pengguna pada Mac Messages yang sudah masuk. Penyiapan manual dan topologi pembungkus SSH tetap dikelola operator: pasang atau perbarui imsg dalam konteks pengguna yang sama dengan yang akan menjalankan Gateway atau pembungkusnya.

Konfigurasi kanal

```
{
  channels: {
    imessage: {
      enabled: true,
      cliPath: "/usr/local/bin/imsg",
      dbPath: "/Users/user/Library/Messages/chat.db",
    },
  },
}
```

Sesuaikan cliPath dengan keluaran which imsg pada mesin Anda.

BAB 23 / IZIN

Izin macOS dan jebakan konteks proses

Syarat	Keterangan
Messages masuk	Harus sudah masuk pada Mac yang menjalankan imsg
Full Disk Access	Untuk konteks proses yang menjalankan OpenClaw atau imsg, demi akses basis data Messages
Automation	Untuk mengirim pesan melalui Messages.app
SIP dimatikan	Hanya untuk aksi lanjutan; teks dan media dasar tidak memerlukannya

INI PENYEBAB KEGAGALAN NOMOR SATU

Izin diberikan per konteks proses. Bila Gateway berjalan headless lewat LaunchAgent atau SSH, jalankan satu perintah interaktif sekali dalam konteks yang sama untuk memicu prompt izinnya.

```
imsg chats --limit 1
# atau
imsg send <handle> "test"
```

Jalankan dari konteks yang sama dengan yang akan dipakai Gateway.

Bila pembacaan berhasil tetapi pengiriman gagal dengan AppleEvents -1743, periksa apakah izin Automation mendarat pada /usr/libexec/sshd-keygen-wrapper alih-alih pada proses imsg atau shell lokal. Bila itu terjadi, macOS mungkin tidak memunculkan tombol Messages yang dapat dipakai untuk klien sisi server SSH tersebut. Bila Automation tidak dapat diberikan, konfigurasikan ulang ke penyiapan imsg pengguna tunggal alih-alih mengandalkan pembungkus SSH untuk pengiriman.

BAB 23 / PAIRING

Menyetujui percakapan pertama

DM iMessage memakai mode pairing secara bawaan. Setelah Gateway berjalan, pesan pertama dari nomor yang belum dikenal akan menghasilkan kode, bukan balasan.

```
openclaw gateway

openclaw pairing list imessage
openclaw pairing approve imessage <CODE>
```

Permintaan pemasangan kedaluwarsa setelah satu jam.

```
openclaw channels status --probe
```

Probe memastikan jembatan benar-benar hidup, bukan sekadar terkonfigurasi.

Gateway di mesin lain

Gateway Linux dan Windows tetap dapat memakai iMessage dengan mengarahkan channels.imessage.cliPath ke sebuah pembungkus SSH yang menjalankan imsg pada Mac yang sudah masuk. Topologi ini bekerja, tetapi ia menambah satu titik kegagalan dan memperumit masalah izin yang baru saja dibahas.

BAB 23 / PRAKTIKUM

Memasang iMessage tanpa terjebak izin

Bagian tersulit pemasangan iMessage bukan perintahnya, melainkan izin macOS yang diberikan per konteks proses. Praktikum ini menyusun urutannya supaya Anda tidak kehilangan sore hari untuk satu galat yang sama.

Langkah 1 — buktikan imsg bekerja sendiri dulu

```
which imsg
imsg --version
imsg chats --limit 3
imsg rpc --help
```

Bila perintah ketiga gagal, OpenClaw pasti ikut gagal. Perbaiki di tingkat ini lebih dulu.

Langkah 2 — pahami konteks proses Anda

Cara Gateway berjalan	Konteks yang memegang izin
Dari Terminal	Terminal.app
Dari layanan launchd	Proses layanan itu sendiri
Lewat pembungkus SSH	Proses sisi server SSH
Dari aplikasi macOS	OpenClaw.app

Izin yang Anda berikan lewat Terminal tidak berlaku untuk proses layanan. Inilah penyebab tunggal dari mayoritas kasus “bisa membaca tetapi tidak bisa mengirim”.

Langkah 3 — picu prompt dari konteks yang benar

```
# Jalankan dari konteks yang sama dengan yang akan dipakai Gateway
imsg chats --limit 1
imsg send <handle> "test"
```

Satu perintah interaktif sekali saja cukup untuk memunculkan prompt izinnya.

Langkah 4 — pasang plugin dan sambungkan

```
openclaw plugins install @openclaw/imessage
openclaw gateway restart
openclaw channels status --probe

openclaw pairing list imessage
openclaw pairing approve imessage <CODE>
```

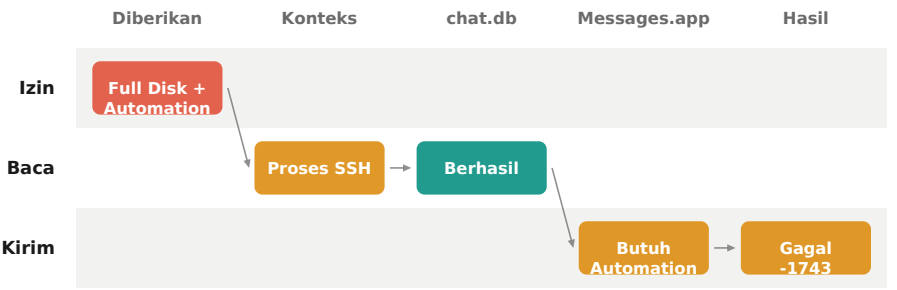
Probe membuktikan jembatan hidup; menyimpan konfigurasi tidak membuktikan apa pun.

BAB 23 / STUDI KASUS

Membaca berhasil, mengirim gagal

Sebuah tim menjalankan Gateway di Mac mini kantor dan mengaksesnya lewat SSH untuk perawatan. Mereka memasang iMessage dan memberikan seluruh izin yang diminta saat prompt muncul di layar Mac itu.

Agen dapat membaca pesan masuk dengan sempurna. Setiap upaya mengirim gagal dengan galat AppleEvents. Mereka mencabut dan memberikan ulang izin berkali-kali, memasang ulang imsg, bahkan memasang ulang macOS pada akhirnya.



Gambar 23.3 — Pembacaan hanya butuh akses berkas; pengiriman butuh izin otomasi yang mendarat pada konteks yang salah.

Penyebabnya: izin Automation terdaftar untuk pembungkus SSH, bukan untuk proses imsg atau shell lokal. Pada situasi itu macOS mungkin tidak memunculkan tombol Messages yang dapat dipakai untuk klien sisi server tersebut.

GEJALA YANG LANGSUNG MENUNJUK PENYEBAB

Bila pembacaan berhasil tetapi pengiriman gagal dengan galat AppleEvents, periksa ke mana izin Automation mendarat sebelum mencurigai hal lain. Bila Automation tidak dapat diberikan, konfigurasikan ulang ke penyiapan imsg pengguna tunggal alih-alih mengandalkan pembungkus SSH untuk pengiriman.

BAB 23 / LATIHAN

Latihan mandiri

1. Jalankan keempat perintah verifikasi imsg dan pastikan semuanya berhasil sebelum menyentuh konfigurasi OpenClaw.
2. Tentukan konteks proses mana yang akan menjalankan Gateway Anda, lalu picu prompt izin dari konteks itu secara sengaja.
3. Uji pengiriman dan penerimaan secara terpisah, dan catat pesan galat masing-masing bila ada.
4. Bila Anda memakai pembungkus SSH, periksa ke mana izin Automation mendarat dan putuskan apakah penyiapan pengguna tunggal lebih tepat.

BAB 23 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
imsg chats gagal dijalankan tangan	Perbaiki dulu di tingkat imsg; OpenClaw tidak akan berhasil bila ini gagal.
Pesan terbaca tetapi tidak terkirim	AppleEvents -1743. Automation mendarat pada konteks proses yang salah.
Gateway headless tidak punya izin	Jalankan sekali perintah interaktif dalam konteks yang sama untuk memicu prompt.
Bot diam terhadap nomor baru	Perilaku pairing yang benar. Setujui lewat openclaw pairing approve.
Panduan menyebut server BlueBubbles	Sudah dihapus. Jalur yang didukung hanya imsg.

Tiga pertanyaan review

1. Mengapa izin macOS diberikan per konteks proses, dan apa akibat praktisnya untuk Gateway yang dijalankan sebagai LaunchAgent?
2. Kapan pembungkus SSH pantas dipakai, dan risiko tambahan apa yang ia bawa?
3. Anda memasang iMessage untuk bisnis. Argumenkan mengapa Mac khusus lebih baik daripada Mac pribadi Anda sendiri.

Kunci pembahasan

macOS mengaitkan izin TCC pada biner dan konteks yang memintanya, sehingga izin yang Anda berikan lewat Terminal tidak otomatis berlaku untuk proses LaunchAgent; akibatnya Anda harus memicu prompt sekali dari konteks yang sama persis. Pembungkus SSH pantas hanya ketika Gateway memang tidak dapat berada di Mac Messages, dan risikonya adalah izin Automation yang mendarat pada pembungkus SSH dan bukan pada imsg,

menghasilkan kanal yang bisa membaca tetapi tidak bisa mengirim. Mac khusus lebih baik karena kanal bisnis menuntut mesin yang selalu hidup, karena Bab 24 akan menuntut SIP dimatikan untuk fitur penuh, dan karena percakapan pribadi Anda tidak seharusnya berada di basis data yang sama dengan percakapan pelanggan.

Rujukan: dokumentasi OpenClaw, halaman channels/imessage, channels/imessage/setup, dan announcements/bluebubbles-imessage, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 24 iMessage: private API, SIP, dan aksi native

Mode dasar mengirim dan menerima teks serta media. Segala sesuatu yang membuat iMessage terasa seperti iMessage — balasan berulir, tapback, efek, polling native, pengelolaan grup — menuntut jembatan private API. Jembatan itu menuntut System Integrity Protection dimatikan, dan itu keputusan yang tidak boleh diambil dengan santai.

Tujuan belajar

Menimbang pertukaran keamanan SIP dengan jujur, menyalakan jembatan private API bila keputusannya memang ya, dan mengetahui apa persisnya yang Anda dapatkan. Pada akhir bab, pembaca dapat mengambil keputusan ini secara sadar, bukan karena terdorong fitur.

Dua mode operasional

MODE DASAR		PRIVATE API
Tetap menyala	**SIP**	Harus dimatikan
Ya	**Kirim teks dan media**	Ya
Tidak	**Balasan berulir**	Ya
Tidak	**Tapback dan efek**	Ya
Tidak	**Edit dan unsend**	Ya
Tidak	**Polling native**	Ya
Tidak	**Pengelolaan grup**	Ya
Tidak	**Indikator mengetik**	Ya

Gambar 24.1 — Apa yang Anda tukarkan. Kolom kiri adalah hasil pemasangan bersih tanpa perubahan sistem apa pun.

Mode dasar adalah bawaannya dan tidak menuntut perubahan SIP: pengiriman teks dan media keluar lewat send, pengamatan dan riwayat masuk, serta daftar obrolan. Itulah yang Anda dapat dari pemasangan brew yang segar ditambah izin macOS standar.

Mode private API menyuntikkan sebuah helper dylib ke dalam Messages.app untuk memanggil fungsi IMCore internal. Teknik penyuntikan helper memakai dylib milik imsg sendiri untuk menjangkau private API Messages; tidak ada server pihak ketiga maupun runtime BlueBubbles di jalur iMessage OpenClaw.

BAB 24 / PERTUKARAN

Apa artinya mematikan SIP

BACA INI DUA KALI SEBELUM MELANJUTKAN
Mematikan SIP adalah pertukaran keamanan yang nyata. SIP adalah salah satu perlindungan inti macOS terhadap berjalannya kode sistem yang dimodifikasi; mematikannya di tingkat sistem membuka permukaan serangan tambahan beserta efek sampingnya. Pada Mac Apple Silicon, mematikan SIP juga menonaktifkan kemampuan memasang dan menjalankan aplikasi iOS di Mac Anda.

Perlakukan ini sebagai pilihan operasional yang disengaja, terutama pada Mac pribadi utama. Untuk iMessage OpenClaw berkualitas produksi, lebih baik memakai Mac khusus atau pengguna macOS bot tempat Anda merasa nyaman menyalakan jembatan ini.

- Mac khusus atau pengguna bot, bukan Mac pribadi utama
- Keputusan dicatat beserta alasannya, untuk audit di kemudian hari
- Pemilik bernama yang bertanggung jawab atas mesin itu
- Mematikan SIP di laptop kerja yang juga dipakai untuk hal lain
- Menyalakan private API hanya karena fiturnya terlihat menarik

Gambar 24.2 — Lima pertimbangan. Dua yang terakhir adalah cara paling umum keputusan ini berubah menjadi penyesalan.

Bila model ancaman Anda tidak dapat menoleransi SIP dimatikan di mana pun, plugin iMessage terbatas pada mode dasar — kirim dan terima teks serta media saja, tanpa reaksi, edit, unsend, efek, maupun operasi grup. Itu pembatasan yang sah dan sering kali merupakan jawaban yang benar.

BAB 24 / MENYALAKAN

Menyalakan jembatan

Private API menuntut SIP mati, validasi pustaka dilonggarkan, dan penyuntikan helper yang berhasil. imsg menolak menyuntik ketika SIP masih menyala.

```
imsg launch
imsg status --json

openclaw channels status --probe
```

Jalankan launch, verifikasi statusnya, lalu probe dari sisi OpenClaw.

Balasan, tapback, efek, polling, balasan lampiran, dan aksi grup menuntut imsg launch beserta probe private API yang berhasil. Bila probe gagal, kanal tetap berjalan dalam mode dasar — ia tidak mati, hanya kehilangan aksi lanjutannya.

Aksi yang terbuka

Kelompok	Aksi
Reaksi dan penyuntingan	react, edit, unsend
Percakapan	reply berulir, sendWithEffect
Polling	poll dan poll-vote, memakai polling Messages native
Grup	renameGroup, setGroupIcon, addParticipant, removeParticipant, leaveGroup
Sinyal kehadiran	Indikator mengetik dan tanda dibaca

BAB 24 / PRAKTIKUM

Mengambil keputusan SIP dengan sadar

Praktikum ini bukan panduan mematikan SIP. Ia adalah proses pengambilan keputusan yang harus Anda lewati lebih dulu, dan bagi sebagian pembaca hasilnya adalah tidak mematikannya sama sekali.

Langkah 1 — jalankan mode dasar dan ukur kekurangannya

Pakai mode dasar selama dua minggu penuh. Catat setiap kali Anda benar-benar membutuhkan aksi yang hanya tersedia lewat private API — bukan setiap kali Anda menginginkannya.

Yang dicatat	Contoh
Aksi yang dibutuhkan	Balasan berulir pada grup yang ramai
Berapa kali dalam dua minggu	Angka nyata, bukan perkiraan
Dampak bila tidak ada	Pesan membingungkan; pelanggan salah paham
Alternatif tanpa private API	Kutip teks pesan secara manual

Langkah 2 — periksa mesin yang tersedia

```
csrutil status               # apakah SIP menyala
system_profiler SPHardwareDataType | grep Chip
```

Pada Apple Silicon, mematikan SIP juga menonaktifkan kemampuan menjalankan aplikasi iOS di Mac itu.

Langkah 3 — jawab empat pertanyaan sebelum memutuskan

- ✓ Ada Mac khusus atau pengguna bot yang bukan mesin pribadi utama
- ✓ Kebijakan perusahaan mengizinkan SIP dimatikan pada mesin itu
- ✓ Ada pemilik bernama yang bertanggung jawab atas mesin tersebut
- ✓ Kebutuhannya terbukti dari pengukuran dua minggu, bukan dari keinginan
- ⚠ Mesin itu juga dipakai untuk pekerjaan lain

Gambar 24.3 — Empat syarat dan satu penghalang. Bila salah satu dari empat yang pertama tidak terpenuhi, jawabannya adalah tetap di mode dasar.

Langkah 4 — bila memutuskan ya, verifikasi dan catat

```
imsg launch
imsg status --json
openclaw channels status --probe
```

Catat tanggal keputusan, alasannya, dan siapa yang menyetujuinya.

BAB 24 / STUDI KASUS

Fitur yang dimatikan diam-diam oleh pembaruan macOS

Sebuah tim menjalankan iMessage dengan private API selama empat bulan. Tapback, balasan berulir, dan polling bekerja, dan pelanggan menyukainya.

Sebuah pembaruan macOS berjalan otomatis pada akhir pekan. Jembatan private API berhenti bekerja, dan kanal turun ke mode dasar. Pesan tetap terkirim dan diterima, sehingga tidak ada peringatan apa pun. Selama sebelas hari tidak ada yang menyadari bahwa tapback tidak pernah lagi terkirim.

Yang tetap bekerja	Yang diam-diam berhenti
Kirim dan terima teks	Tapback dan reaksi
Kirim dan terima media	Balasan berulir
Riwayat dan daftar obrolan	Efek dan polling native
Pairing dan kontrol akses	Pengelolaan grup

Sifat yang membuat kanal tetap hidup dalam mode dasar adalah rancangan yang baik — pesan pelanggan tidak berhenti mengalir hanya karena satu fitur hilang. Tetapi sifat yang sama membuat kehilangan itu tidak terlihat.

> **PEMERIKSAAN YANG LAYAK DIOTOMATISKAN**
> Masukkan probe private API ke dalam pemeriksaan rutin Anda, bukan hanya ke dalam prosedur pemasangan. Satu perintah pada pemeriksaan harian akan menemukan ini dalam dua puluh empat jam, bukan sebelas hari.

BAB 24 / LATIHAN

Latihan mandiri

1. Jalankan mode dasar selama dua minggu dan catat berapa kali Anda benar-benar membutuhkan aksi private API.
2. Jawab keempat pertanyaan pada Gambar 24.3 untuk situasi Anda, lalu tuliskan keputusannya beserta alasan.
3. Bila jembatan private API Anda aktif, susun pemeriksaan harian yang akan mendeteksi ketika ia berhenti bekerja.
4. Tuliskan apa yang akan Anda sampaikan kepada pelanggan bila fitur private API harus dimatikan permanen.

BAB 24 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
imsg launch menolak berjalan	SIP masih menyala. imsg menolak menyuntik dalam keadaan itu.
Tapback dan efek tidak berfungsi	Probe private API gagal. Kanal turun ke mode dasar secara diam-diam.
Aplikasi iOS hilang dari Mac setelah SIP dimatikan	Efek samping yang terdokumentasi pada Apple Silicon, bukan kerusakan.
Ingin fitur penuh tetapi kebijakan melarang SIP mati	Mode dasar adalah batasnya. Itu jawaban yang sah.
Setelah pembaruan macOS, jembatan mati	Penyuntikan helper perlu dijalankan ulang; verifikasi dengan imsg status.

Tiga pertanyaan review

1. Mengapa kegagalan probe private API tidak mematikan kanal, dan mengapa perilaku itu justru berisiko menyesatkan operator?
2. Sebutkan dua efek samping mematikan SIP yang tidak berhubungan langsung dengan OpenClaw.
3. Perusahaan Anda ingin polling native di grup pelanggan iMessage. Susun argumen untuk menyetujui atau menolak permintaan itu.

Kunci pembahasan

Kanal tetap hidup dalam mode dasar supaya pesan pelanggan tidak berhenti mengalir hanya karena satu fitur tidak tersedia; risikonya adalah operator mengira semuanya normal sampai seseorang menyadari tapback tidak pernah terkirim, sehingga probe perlu dimasukkan ke pemeriksaan rutin. Dua efek samping di luar OpenClaw adalah permukaan

serangan tambahan terhadap kode sistem yang dimodifikasi, dan hilangnya kemampuan menjalankan aplikasi iOS pada Mac Apple Silicon. Untuk permintaan polling native, jawaban yang bertanggung jawab adalah menyetujuinya hanya bila tersedia Mac khusus dengan pemilik bernama; bila fitur itu harus dijalankan di mesin yang juga dipakai untuk pekerjaan lain, tolak dan tawarkan polling berbasis teks di mode dasar.

Rujukan: dokumentasi OpenClaw, halaman channels/imessage/private-api dan channels/imessage/rich-messages, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 25 iMessage: kontrol akses, grup, dan penempatan

Dua bab sebelumnya memasang dan membuka fitur. Bab ini menguncinya. iMessage membawa satu sifat yang tidak dimiliki kanal bot: ia hidup di akun pribadi seseorang, bukan pada identitas bot terpisah. Sifat itu mengubah seluruh perhitungan keamanan.

Tujuan belajar

Menyetel kebijakan DM dan grup untuk iMessage, memilih pola penempatan yang sesuai, dan memahami konsekuensi berbagi akun. Pada akhir bab, pembaca dapat menjalankan iMessage untuk bisnis tanpa mencampurnya dengan kehidupan pribadi.

Kebijakan akses

```
{
  channels: {
    imessage: {
      enabled: true,
      dmPolicy: "pairing",
      groupPolicy: "allowlist",
      groupAllowFrom: ["chat_id:123"],
    },
  },
}
```

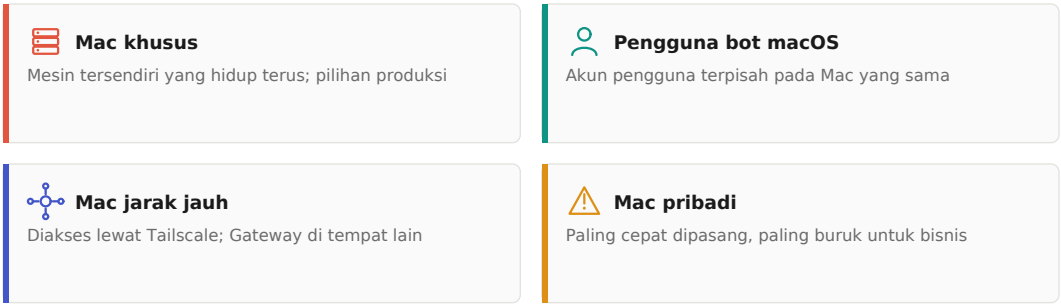
iMessage memakai chat_id pada groupAllowFrom, bukan nomor telepon.

Perhatikan bentuk entri grup pada iMessage: ia memakai identitas obrolan, bukan nomor pengirim. Ini berbeda dari WhatsApp dan Signal yang memakai nomor E.164, dan perbedaan itu adalah sumber kebingungan yang berulang ketika seseorang menyalin konfigurasi antar kanal.

Kanal	Bentuk entri groupAllowFrom
WhatsApp	Nomor E.164, misalnya +6281234567890
Telegram	ID pengguna numerik; wizard menyelesaikan @username
Signal	Nomor E.164
iMessage	chat_id:<id>
Microsoft Teams	Alamat surel pengguna

BAB 25 / PENEMPATAN

Empat pola penempatan



Gambar 25.1 — Empat pola. Kotak keempat sah untuk percobaan pribadi, dan hampir selalu keliru untuk melayani pelanggan.

MENGAPA MAC KHUSUS BUKAN KEMEWAHAN

Untuk iMessage OpenClaw berkualitas produksi, lebih baik memakai Mac khusus atau pengguna macOS bot tempat Anda merasa nyaman menyalakan jembatan private API. Alasannya berlapis: mesin harus hidup terus, SIP kemungkinan dimatikan, dan percakapan pelanggan tidak seharusnya berada di basis data yang sama dengan percakapan keluarga Anda.

Riwayat DM yang sudah ada

Satu kejutan yang sering terjadi pada pemasangan pertama: basis data Messages sudah berisi riwayat bertahun-tahun. Agen yang diberi akses membaca dapat melihatnya. Sebelum menyalakan kanal ini pada Mac yang pernah dipakai pribadi, pertimbangkan apa yang sebenarnya ada di dalam chat.db tersebut.

BAB 25 / GRUP

Grup dan gerbang mention

Aturan grup OpenClaw berlaku konsisten di seluruh kanal yang mendukung grup. Secara bawaan, grup dibatasi dengan groupPolicy allowlist, pengirim grup diblokir sampai masuk daftar izin, dan balasan menuntut mention kecuali Anda mematikan gerbang mention untuk grup itu.

Gambar 25.2 — Empat gerbang berurutan. Satu saja gagal, pesan berhenti — dan itu perilaku yang benar.

Otorisasi pengirim grup tetap eksplisit terhadap daftar izin grup. Persetujuan pairing DM — entri pada penyimpanan allowFrom — berlaku untuk akses DM saja. Menyetujui seseorang di DM tidak dengan sendirinya memberinya kewenangan memicu agen di dalam grup.

BAB 25 / PRAKTIKUM

Mengunci iMessage sebelum ia melayani siapa pun

Praktikum ini dikerjakan setelah kanal berfungsi tetapi sebelum orang pertama di luar tim Anda mengetahui nomornya. Jendela waktu itu sempit dan tidak kembali.

Langkah 1 — periksa apa yang ada di basis data Messages

```
imsg chats --limit 20
ls -lh ~/Library/Messages/chat.db
```

Bila mesin ini pernah dipakai pribadi, riwayatnya ada di sana dan agen dapat membacanya.

Hentikan sejenak dan baca keluaran perintah pertama. Bila Anda melihat percakapan pribadi, keluarga, atau apa pun yang tidak seharusnya dibaca agen bisnis, itu jawaban bahwa mesin ini bukan mesin yang tepat.

Langkah 2 — tetapkan bentuk entri yang benar

```
{
  channels: {
    imessage: {
      dmPolicy: "pairing",
      groupPolicy: "allowlist",
      groupAllowFrom: ["chat_id:123"],
```

```
    },
  },
}
```

iMessage memakai chat_id pada groupAllowFrom — bukan nomor telepon seperti WhatsApp dan Signal.

Langkah 3 — uji empat gerbang grup satu per satu

Uji	Cara	Hasil yang benar
groupPolicy	Setel disabled, kirim pesan grup	Tidak ada respons apa pun
Daftar grup	Setel allowlist tanpa grup itu	Pesan dijatuhkan, ada jejak log
Gerbang mention	Kirim tanpa menyebut bot	Diam
Otorisasi pengirim	Minta orang di luar daftar menyebut bot	Tetap diam

Uji keempat adalah yang paling sering mengejutkan: seseorang yang sudah disetujui lewat pairing DM tetap tidak berwenang memicu agen di dalam grup. Otorisasi grup berdiri sendiri.

BAB 25 / STUDI KASUS

Agen yang mengutip percakapan keluarga

Sebuah usaha kecil memasang iMessage di MacBook pemiliknya, karena itulah Mac yang tersedia. Kanal berjalan, pelanggan dilayani, dan selama dua bulan tidak ada masalah.

Suatu hari seorang pelanggan menanyakan jadwal pengiriman. Agen menjawab benar, lalu menambahkan satu kalimat yang merujuk rencana perjalanan keluarga pemiliknya — informasi yang ada di dalam riwayat iMessage pribadi pada mesin yang sama.

Keputusan yang diambil	Konsekuensi yang tidak terlihat
Memakai Mac yang tersedia	Riwayat pribadi masuk jangkauan agen
Tidak memeriksa isi chat.db	Tidak ada yang tahu apa yang dapat dibaca
Mengandalkan instruksi prompt	Instruksi bukan batas; lihat Bab 51
Tidak ada pemisahan pengguna macOS	Satu basis data untuk dua konteks hidup

Tidak ada kebocoran ke pihak luar dalam arti teknis: agen hanya memakai data yang memang dapat diaksesnya. Yang bocor adalah batas antara kehidupan pribadi pemilik dan layanan pelanggannya — batas yang tidak pernah dibuat.

SATU PERINTAH YANG MENCEGAH SELURUH KASUS INI
Sebelum menyalakan iMessage pada sebuah Mac, jalankan imsg chats dan baca keluarannya. Bila Anda tidak nyaman agen membaca apa yang muncul di sana, pakailah Mac atau pengguna macOS yang berbeda.

BAB 25 / LATIHAN

Latihan mandiri

1. Jalankan imsg chats pada mesin kandidat Anda dan putuskan apakah isinya pantas berada dalam jangkauan agen.
2. Jalankan keempat uji gerbang grup dan catat hasil masing-masing.
3. Bandingkan bentuk entri daftar izin grup pada iMessage, WhatsApp, dan Telegram, lalu tuliskan perbedaannya supaya Anda tidak menyalin konfigurasi yang keliru.
4. Tuliskan argumen satu paragraf untuk meminta Mac khusus, dalam bahasa yang dipahami orang non-teknis.

BAB 25 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
groupAllowFrom tidak berpengaruh	iMessage memakai chat_id:<id>, bukan nomor telepon.
Orang yang disetujui di DM tidak bisa memicu di grup	Perilaku yang benar. Otorisasi grup terpisah dari pairing DM.
Agen menyebut isi percakapan pribadi lama	chat.db memuat riwayat lama. Pakai Mac atau pengguna terpisah.
Bot diam di grup meski sudah diizinkan	Gerbang mention masih aktif. Sebut namanya atau ubah setelan grup itu.
Kanal mati saat Mac tidur	iMessage menuntut Mac yang hidup terus. Pindahkan ke mesin khusus.

Tiga pertanyaan review

1. Mengapa otorisasi grup sengaja dipisahkan dari pairing DM? Berikan satu skenario yang menunjukkan bahayanya bila digabung.
2. Sebutkan dua alasan teknis dan satu alasan non-teknis untuk memakai Mac khusus.
3. Anda diminta memasang iMessage di MacBook direktur. Susun tanggapan profesional Anda.

Kunci pembahasan

Pemisahan itu ada karena mengizinkan seseorang berbicara berdua dengan agen sangat berbeda dari mengizinkannya memerintah agen di depan dua puluh orang lain; bila digabung, satu pelanggan yang pernah disetujui di DM dapat memicu perintah di grup internal yang kebetulan ia ikuti. Dua alasan teknis: mesin harus hidup terus, dan SIP kemungkinan harus dimatikan; alasan non-teknisnya adalah pemisahan data pribadi dari data pelanggan. Untuk permintaan memasang di MacBook direktur, tanggapan yang benar adalah menjelaskan bahwa laptop itu akan tidur sehingga kanal mati, bahwa fitur penuh menuntut SIP dimatikan pada perangkat yang membawa data perusahaan, dan bahwa seluruh riwayat iMessage pribadinya akan berada dalam jangkauan agen — lalu menawarkan Mac mini khusus sebagai gantinya.

Rujukan: dokumentasi OpenClaw, halaman channels/imessage/access-control, channels/imessage/deployment, dan channels/groups, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 26 WhatsApp: pemasangan dan kontrol akses

Untuk sebagian besar bisnis di Indonesia, WhatsApp bukan salah satu pilihan kanal — ia adalah kanalnya. Bab ini memasangnya dengan benar sejak awal, karena kesalahan konfigurasi di kanal ini langsung terlihat oleh pelanggan.

Tujuan belajar

Menautkan akun lewat QR, menyetel kebijakan DM, dan memahami bahwa OpenClaw hidup di akun WhatsApp Anda sendiri. Pada akhir bab, pembaca memiliki kanal WhatsApp yang hanya melayani orang yang seharusnya dilayani.

Sifat kanal ini

Statusnya siap produksi lewat WhatsApp Web dengan Baileys, dan Gateway memiliki sesi yang tertaut. Tidak ada pengguna bot WhatsApp yang terpisah: OpenClaw hidup pada akun perpesanan Anda sendiri. Bila Anda berada di sebuah grup, OpenClaw dapat melihat grup itu dan menanggapi di sana.

> **NOMOR TERPISAH ADALAH SARAN RESMI**
> OpenClaw menyarankan menjalankan WhatsApp pada nomor terpisah bila memungkinkan. Metadata kanal dan alur onboarding-nya dioptimalkan untuk penyiapan itu, meskipun penyiapan dengan nomor pribadi juga didukung.

Empat langkah

Gambar 26.1 — Urutannya penting: kebijakan akses disetel sebelum akun ditautkan, bukan sesudahnya.

```
{
  channels: {
    whatsapp: {
      enabled: true,
      dmPolicy: "pairing",
      allowFrom: ["+6281234567890"],
      groupPolicy: "allowlist",
      groupAllowFrom: ["+6281234567890"],
    },
  },
```

```
}
```

Setel ini lebih dulu. allowFrom menerima nomor gaya E.164 dan dinormalkan internal.

```
openclaw channels login --channel whatsapp
openclaw channels login --channel whatsapp --account work

openclaw gateway

openclaw pairing list whatsapp
openclaw pairing approve whatsapp <CODE>
```

Permintaan pairing kedaluwarsa setelah satu jam; tertunda dibatasi tiga per kanal.

BAB 26 / KEBIJAKAN DM

Empat kebijakan dan perilaku runtime-nya

Nilai dmPolicy	Perilaku
pairing	Bawaannya; pengirim asing menerima kode dan menunggu persetujuan
allowlist	Hanya nomor yang terdaftar yang dilayani
open	Menuntut allowFrom memuat tanda bintang
disabled	Tidak melayani pesan langsung sama sekali

Beberapa perilaku runtime layak dihafal. Pairing dipertahankan pada penyimpanan izin kanal dan digabungkan dengan allowFrom yang dikonfigurasi. Bila tidak ada daftar izin yang dikonfigurasi, nomor diri yang tertaut diizinkan secara bawaan. DM keluar yang berasal dari Anda sendiri tidak pernah dipasangkan otomatis.

Untuk penyiapan multi-akun, channels.whatsapp.accounts.<id>.dmPolicy beserta allowFrom-nya lebih diutamakan daripada default tingkat kanal untuk akun tersebut. Perintah keluar secara bawaan memakai akun default bila ada; bila tidak, ia memakai id akun pertama menurut urutan abjad.

```
{
  channels: {
    whatsapp: {
      accounts: {
        default: {},
        personal: {},
        biz: {},
      },
    },
  },
}
```

Beberapa akun pada satu Gateway, masing-masing dengan sesi tertautnya sendiri.

BAB 26 / PERILAKU PESAN

Sesi, potongan, dan tanda baca

Obrolan status dan siaran diabaikan. Obrolan langsung memakai aturan sesi DM; bawaannya menciutkan DM ke sesi utama agen. Sesi grup terisolasi dengan kunci sesi yang memuat JID grupnya.

```
{
  channels: {
    whatsapp: {
      textChunkLimit: 4000,
      streaming: { chunkMode: "length" },
      mediaMaxMb: 50,
      sendReadReceipts: true,
    },
  },
}
```

chunkMode menerima length atau newline; sendReadReceipts mengatur centang biru.

PERIKSA SEBELUM DIPAKAI BANYAK ORANG

Menciutkan seluruh DM ke sesi utama berarti beberapa pengirim berbagi satu percakapan. Untuk kotak masuk bersama, audit keamanan akan menyarankan session.dmScope per-channel-peer. Ini pengerasan untuk kerja sama, bukan isolasi antara orang yang saling tidak percaya.

BAB 26 / PRAKTIKUM

Menautkan WhatsApp dengan urutan yang benar

Urutan pada praktikum ini penting. Menautkan akun sebelum kebijakan akses disetel membuka jendela waktu di mana siapa pun yang mengetahui nomor Anda dapat memerintah agen.

Langkah 1 — siapkan daftar izin sebelum menyentuh QR

```
# Kumpulkan nomor dalam format E.164 lebih dulu
+6281234567890
+6281298765432
# lalu tulis ke konfigurasi, baru tautkan akun
```

allowFrom menerima nomor gaya E.164 dan menormalkannya secara internal.

Langkah 2 — tautkan dan verifikasi

```
openclaw channels login --channel whatsapp
openclaw gateway
openclaw channels status --probe
```

Probe membuktikan sesi tertaut benar-benar hidup.

Langkah 3 — uji perilaku runtime yang perlu dihafal

Perilaku	Cara mengujinya	Mengapa penting
Pairing tergabung dengan allowFrom	Setujui satu nomor, periksa daftar efektifnya	Dua sumber izin bekerja bersamaan
Nomor diri diizinkan bila daftar kosong	Kosongkan allowFrom sementara di mesin uji	Menjelaskan mengapa pesan Anda sendiri selalu dilayani
DM keluar tidak dipasangkan otomatis	Kirim pesan ke nomor baru dari akun itu	Mengirim tidak sama dengan memberi izin
Akun bernama menimpa default	Setel dmPolicy berbeda pada satu akun	Multi-akun punya presedensinya sendiri

Langkah 4 — setel perilaku pesan

```
{
  channels: { whatsapp: {
    textChunkLimit: 4000,
    streaming: { chunkMode: "length" },
    mediaMaxMb: 50,
    sendReadReceipts: true,
  } },
}
```

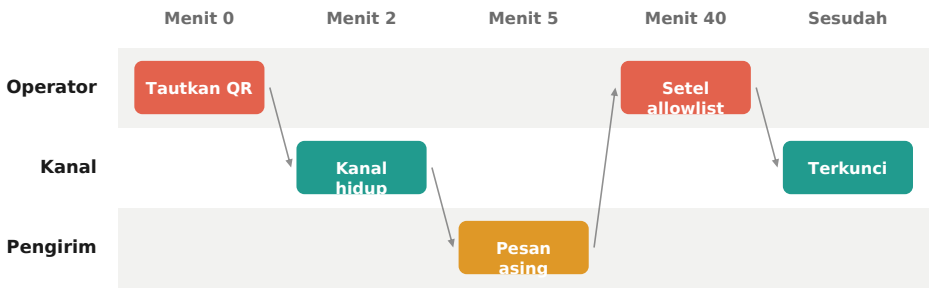
chunkMode menerima length atau newline; pilih sesuai gaya balasan agen Anda.

BAB 26 / STUDI KASUS

Sepuluh menit antara QR dan daftar izin

Seorang operator menautkan WhatsApp terlebih dahulu supaya dapat menguji QR-nya, berniat menyetel daftar izin segera sesudahnya. Di tengah proses ia terpanggil rapat dan meninggalkannya selama sekitar empat puluh menit.

Nomor itu adalah nomor bisnis lama yang masih tercantum di beberapa direktori daring. Dalam rentang itu, tujuh nomor asing mengirim pesan. Kebijakan bawaan pairing menahan sebagian, tetapi konfigurasi yang belum disetel membuat perilakunya tidak seperti yang ia kira.



Gambar 26.2 — Jendela empat puluh menit yang tidak direncanakan. Pada kanal sebesar WhatsApp, jendela semenit pun cukup.

Tidak ada kerusakan yang terjadi, dan itu keberuntungan. Yang tersisa adalah tujuh permintaan pemasangan yang harus ditinjau, dan pengetahuan bahwa nomor bisnisnya beredar lebih luas daripada yang ia duga.

> **URUTAN YANG TIDAK BOLEH DIBALIK**
> Setel kebijakan akses lengkap sebelum menjalankan perintah penautan. Bila Anda terpaksa berhenti di tengah, matikan kanal dulu — satu setelan enabled bernilai false lebih murah daripada meninjau puluhan permintaan asing.

BAB 26 / LATIHAN

Latihan mandiri

1. Susun daftar izin lengkap Anda dalam format E.164 sebelum menautkan akun apa pun.
2. Jalankan keempat uji perilaku runtime pada praktikum langkah tiga di lingkungan uji.
3. Tetapkan apakah Anda memakai nomor terpisah atau nomor pribadi, dan tuliskan alasannya beserta risiko yang Anda terima.
4. Rancang prosedur penambahan nomor baru ke daftar izin, termasuk siapa yang boleh menyetujui dan berapa lama janjinya.

BAB 26 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Siapa pun bisa memerintah agen	dmPolicy open dengan wildcard. Kembalikan ke pairing.
Balasan pelanggan tercampur satu sama lain	DM diciutkan ke sesi utama. Pertimbangkan session.dmScope per-channel-peer.
Perintah keluar memakai akun yang salah	Tanpa akun default, dipakai id pertama menurut abjad.
Pesan panjang terpotong aneh	Setel textChunkLimit dan chunkMode sesuai gaya balasan Anda.
Nomor pribadi ikut menerima pesan bisnis	Pakai nomor terpisah; itu saran resmi, bukan sekadar preferensi.

Tiga pertanyaan review

1. Mengapa kebijakan akses harus disetel sebelum menautkan QR, bukan sesudahnya?
2. Apa arti praktis dari 'tidak ada pengguna bot WhatsApp terpisah' bagi tanggung jawab hukum perusahaan Anda?
3. Rancang penyiapan untuk toko yang punya satu nomor layanan pelanggan dan satu nomor pemilik.

Kunci pembahasan

Menyetel kebijakan lebih dulu mencegah jendela waktu ketika akun sudah tertaut tetapi belum terlindungi; pada kanal sebesar WhatsApp, jendela beberapa menit sudah cukup bagi pesan asing untuk masuk. Karena agen hidup pada akun Anda sendiri, setiap pesan yang dikirimnya secara hukum berasal dari pemilik nomor itu — bukan dari sebuah entitas bot — sehingga pemilik nomor memikul tanggung jawab atas isinya. Untuk toko dengan dua nomor, konfigurasikan dua akun pada satu Gateway, ikat nomor layanan ke agen CS dengan skill terbatas, dan biarkan nomor pemilik memakai agen terpisah dengan kewenangan lebih luas.

Rujukan: dokumentasi OpenClaw, halaman channels/whatsapp dan gateway/config-channels/personal-messaging, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 27 WhatsApp: grup, aktivasi, dan konteks

Grup WhatsApp adalah tempat agen paling mudah menjadi gangguan. Bab ini membahas mekanisme yang mencegahnya: dua lapis izin, dua mode aktivasi, dan satu token diam yang membuat agen tahu kapan harus tidak berbicara.

Tujuan belajar

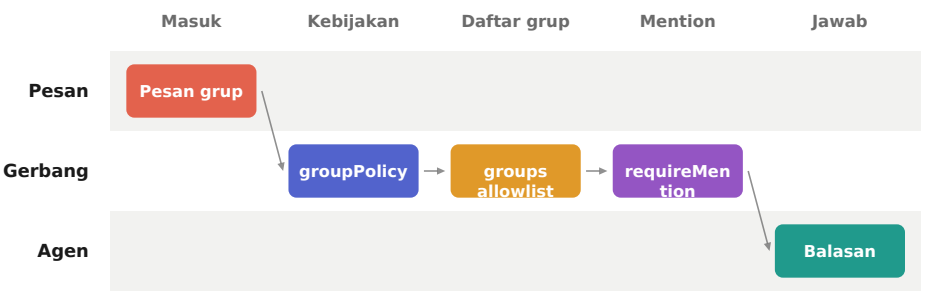
Membatasi grup mana yang dilayani, memilih mode aktivasi yang tepat, dan mengatur berapa banyak konteks yang ikut masuk ke model. Pada akhir bab, pembaca dapat menempatkan agen di grup pelanggan tanpa membuatnya cerewet.

Dua lapis izin grup

Akses grup punya dua lapis. Pertama, groupPolicy menentukan apakah pesan grup diterima sama sekali, dengan nilai open, disabled, atau allowlist. Kedua, ketika channels.whatsapp.groups disetel, hanya JID grup yang terdaftar yang diterima — sertakan tanda bintang untuk mengizinkan semuanya. Pesan dari grup yang tidak terdaftar dijatuhkan disertai petunjuk di log.

```
{
  channels: {
    whatsapp: {
      groupPolicy: "allowlist",
      groupAllowFrom: ["+6281234567890"],
      groups: {
        "*": { requireMention: true },
      },
    },
  },
}
```

groupPolicy allowlist memakai groupAllowFrom; fallback-nya allowFrom eksplisit.



Gambar 27.1 — Tiga gerbang berturut-turut sebelum agen pernah melihat pesan grup.
Masing-masing dapat disetel terpisah.

BAB 27 / AKTIVASI

Mention atau always

MODE MENTION		MODE ALWAYS
Bawaannya	Status	Harus dipilih
Perlu ping eksplisit	Pemicu	Setiap pesan
Rendah	Biaya token	Jauh lebih tinggi
Kecil	Risiko cerewet	Ditahan token diam
Grup pelanggan	Cocok untuk	Grup pemantauan kecil

Gambar 27.2 — Dua mode aktivasi. Mode always membangunkan agen pada setiap pesan, dan itu terasa pada tagihan.

Mode mention menuntut sebuah ping: mention @ WhatsApp yang sungguhan, pola regex yang dikonfigurasi, digit E.164 milik bot di mana pun dalam teks, atau balasan kutipan terhadap salah satu pesan bot — kecuali pada penyiapan self-chat bernomor bersama.

Mode always membangunkan agen pada setiap pesan, tetapi prompt grup yang disuntikkan memberi tahu agen agar hanya membalas ketika ia menambah nilai, dan mengembalikan token diam yang persis bila tidak. Token itu tidak peka huruf besar-kecil.

> **HANYA PEMILIK YANG BOLEH MENGUBAH**
> Default berasal dari konfigurasi pada channels.whatsapp.groups requireMention, dan dapat ditimpa per grup lewat perintah /activation. Hanya nomor pemilik — dari channels.whatsapp.allowFrom, atau nomor E.164 milik bot sendiri bila tidak disetel — yang dapat mengubahnya. Perintah /activation dari orang lain diabaikan dan hanya disimpan sebagai konteks.

```
/activation mention    # kirim sebagai pesan berdiri sendiri
/activation always
/status                # lihat mode aktivasi saat ini
```

Perintah harus dikirim sebagai pesan berdiri sendiri, bukan sisipan kalimat.

BAB 27 / KONTEKS

Berapa banyak yang dilihat model

Prompt agen memuat konteks grup yang tertunda ditambah baris berlabel pengirim, sehingga ia dapat menyapa orang yang tepat. Jendela konteks tertunda diselesaikan secara berjenjang: pengaturan per akun lebih dulu, lalu pengaturan tingkat kanal, lalu pengaturan obrolan grup global, dan akhirnya nilai bawaan lima puluh.

Tingkat	Kunci
Per akun	`channels.whatsapp.accounts.<id>.historyLimit`
Tingkat kanal	`channels.whatsapp.historyLimit`
Global obrolan grup	`messages.groupChat.historyLimit`
Bawaan	`50`

Angka ini menentukan biaya sekaligus kualitas. Terlalu kecil membuat agen kehilangan benang percakapan; terlalu besar membuat setiap balasan mahal dan memperbesar kemungkinan teks dari pengirim yang tidak terotorisasi ikut terbaca model.

BAB 27 / PRAKTIKUM

Menempatkan agen di grup tanpa membuatnya cerewet

Praktikum ini menyetel perilaku grup selangkah demi selangkah, dengan menguji tiap gerbang sebelum melanjutkan. Grup adalah tempat agen paling mudah merusak kepercayaan, dan paling mudah membakar anggaran.

Langkah 1 — mulai dari tertutup

```
{
  channels: { whatsapp: {
    groupPolicy: "disabled",
  } },
}
```

Mulai dari sini, lalu buka satu grup saja setelah Anda siap mengujinya.

Langkah 2 — buka satu grup dan amati tiga gerbang

Gerbang	Yang terjadi bila gagal	Jejaknya
groupPolicy	Pesan grup tidak diterima sama sekali	Tidak ada respons
Daftar groups	Pesan dari grup tak terdaftar dijatuhkan	Petunjuk di log
requireMention	Agen diam meski grup diizinkan	Tidak ada respons

Langkah 3 — ukur biaya kedua mode aktivasi

Jalankan mode mention selama satu minggu dan catat biaya token harian. Lalu ubah ke mode always pada grup yang sama selama satu minggu dan catat lagi. Selisihnya akan mengejutkan sebagian besar orang, dan angka itu yang membuat keputusan Anda mudah.

```
/activation mention    # kirim sebagai pesan berdiri sendiri
/activation always
/status                # lihat mode aktivasi saat ini
```

Hanya nomor pemilik yang dapat mengubahnya; perintah dari orang lain diabaikan dan hanya disimpan sebagai konteks.

Langkah 4 — setel jendela konteks dengan sadar

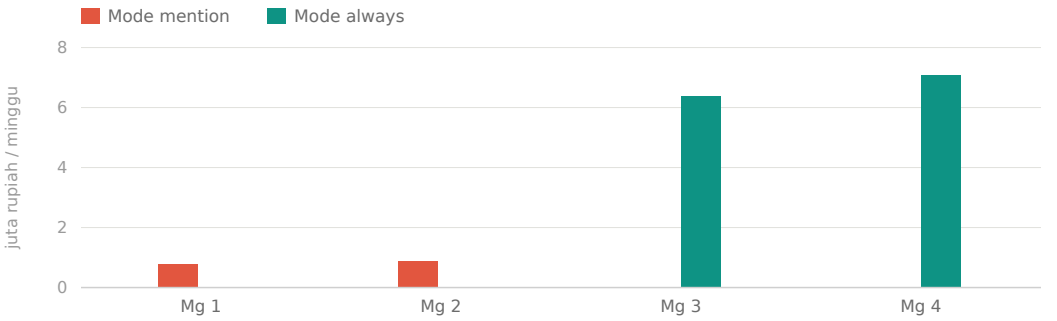
Nilai bawaan lima puluh adalah titik awal, bukan jawaban. Terlalu kecil membuat agen kehilangan benang percakapan; terlalu besar menaikkan biaya dan memperbesar jumlah teks dari pengirim tak terotorisasi yang ikut terbaca model.

BAB 27 / STUDI KASUS

Grup delapan puluh orang dan tagihan yang berlipat

Sebuah distributor menambahkan agen ke grup reseller berisi delapan puluh orang. Karena beberapa reseller lupa menyebut nama bot, tim menyalakan mode always supaya tidak ada pertanyaan yang terlewat.

Grup itu aktif sepanjang hari dengan obrolan biasa: salam, gurauan, foto produk, diskusi antar reseller. Setiap pesan membangunkan agen. Agen menilai sebagian besar tidak layak dijawab dan mengembalikan token diam — tetapi penilaian itu sendiri memakan biaya.



Gambar 27.3 — Biaya mingguan sintetis sebelum dan sesudah mode always. Volume pertanyaan yang dijawab nyaris tidak berubah.

Setelah empat minggu mereka kembali ke mode mention dan menyelesaikan masalah aslinya dengan cara yang lebih murah: menambahkan pola sebutan supaya nama panggilan yang biasa dipakai reseller ikut dikenali sebagai pemicu.

> **SELESAIKAN PENYEBABNYA, BUKAN GEJALANYA**
> Ketika sebuah masalah kecil menggoda Anda menyalakan setelan yang mahal, cari dulu penyelesaian yang menyasar penyebabnya. Reseller yang lupa menyebut bot adalah masalah pengenalan sebutan, bukan masalah mode aktivasi.

BAB 27 / LATIHAN

Latihan mandiri

1. Mulai dari groupPolicy disabled, lalu buka satu grup dan uji ketiga gerbang secara berurutan.
2. Ukur biaya harian pada mode mention selama seminggu. Simpan angkanya sebelum mencoba apa pun yang lain.
3. Tetapkan nilai historyLimit untuk grup teramai Anda dan tuliskan alasan angkanya.
4. Daftarkan pola sebutan yang benar-benar dipakai orang untuk memanggil bot Anda, lalu pastikan semuanya dikenali.

BAB 27 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Agen menjawab di grup yang tidak diinginkan	Daftar groups belum disetel, atau memuat tanda bintang.
Biaya token melonjak setelah masuk grup	Mode always aktif. Kembalikan ke mention kecuali ada alasan kuat.
/activation diabaikan	Pengirimnya bukan nomor pemilik, atau tidak dikirim sebagai pesan berdiri sendiri.
Agen kehilangan benang percakapan	historyLimit terlalu kecil untuk grup seramai itu.
Pesan grup hilang tanpa jejak	Grup tidak terdaftar; periksa petunjuk di log.

Tiga pertanyaan review

1. Mengapa mode always tetap disediakan meski lebih mahal? Sebutkan satu kasus yang membenarkannya.
2. Mengapa hanya nomor pemilik yang boleh menjalankan /activation, dan apa yang terjadi bila aturan itu tidak ada?
3. Rancang pengaturan grup untuk sebuah grup pelanggan berisi 80 orang yang aktif sepanjang hari.

Kunci pembahasan

Mode always dibenarkan pada grup pemantauan kecil yang isinya peringatan sistem, di mana agen memang harus menilai setiap pesan dan sebagian besar waktu diam dengan token diam. Aturan pemilik ada karena tanpa itu siapa pun di grup dapat memaksa agen membaca setiap pesan, mengubah biaya dan perilaku bagi semua orang tanpa persetujuan pemiliknya. Untuk grup pelanggan 80 orang yang ramai, pertahankan

groupPolicy allowlist, daftarkan JID grup itu saja, biarkan requireMention menyala, dan setel historyLimit secukupnya — konteks besar pada grup seramai itu menaikkan biaya dan memperbesar paparan teks dari pengirim yang tidak terotorisasi.

Rujukan: dokumentasi OpenClaw, halaman channels/group-messages dan channels/groups, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 28 Telegram: pemasangan dan transport

Telegram adalah kanal yang paling cepat dipasang di seluruh ekosistem OpenClaw. Token didapat dalam dua menit, dan long polling menghilangkan kebutuhan akan URL publik. Kemudahan itu membuat banyak orang berhenti sebelum mengamankannya.

Tujuan belajar

Membuat token bot, menyetel kebijakan DM, dan memilih antara long polling dan webhook. Pada akhir bab, pembaca memiliki bot Telegram yang berjalan dan hanya melayani orang yang berhak.

Membuat token

Telegram siap produksi untuk DM bot dan grup lewat grammY. Ada dua jalur mendapatkan token, dan keduanya berakhir pada token yang sama. Jalur obrolan: buka Telegram, bicaralah dengan @BotFather — pastikan handle-nya persis seperti itu — jalankan /newbot, ikuti petunjuknya, lalu simpan tokennya. Jalur web: buka aplikasi web BotFather yang berjalan di setiap klien Telegram, buat bot lewat antarmuka, dan salin tokennya.

```
{
  channels: {
    telegram: {
      enabled: true,
      botToken: "123:abc",
      dmPolicy: "pairing",
      groups: { "*": { requireMention: true } },
    },
  },
}
```

Nilai konfigurasi menang atas env; TELEGRAM_BOT_TOKEN hanya berlaku untuk akun default.

BERBEDA DARI WHATSAPP
Telegram tidak memakai openclaw channels login telegram. Setel token di konfigurasi atau lingkungan, lalu jalankan Gateway. Akun bernama harus memakai botToken atau tokenFile, bukan variabel lingkungan.

```
openclaw gateway
openclaw pairing list telegram
openclaw pairing approve telegram <CODE>
```

Kode pairing kedaluwarsa setelah satu jam.

Menemukan ID pengguna Telegram

Cara yang lebih aman, tanpa bot pihak ketiga: kirim DM ke bot Anda, jalankan openclaw

logs --follow, lalu baca medan pengirimnya. Wizard onboarding menerima masukan @username dan menyelesaikannya menjadi ID numerik. Bila Anda memutakhirkan dan konfigurasi masih memuat entri allowlist berupa @username, jalankan openclaw doctor --fix untuk menyelesaikannya — upaya terbaik, dan menuntut token bot.

BAB 28 / TRANSPORT

Long polling atau webhook

LONG POLLING		WEBHOOK
Bawaannya	Status	Opsional
Tidak	Butuh URL publik	Ya, HTTPS
Bekerja	Di balik NAT	Perlu ingress
Satu saja	Jumlah konsumen	Dapat diskalakan
Nyaris nol	Kesulitan pemasangan	Sedang

Gambar 28.1 — Dua transport. Long polling menang untuk hampir semua pemasangan di satu mesin, termasuk Mac di rumah atau kantor.

Untuk mode webhook, setel channels.telegram.webhookUrl dan channels.telegram.webhookSecret. Opsional: webhookPath dengan bawaan /telegram-webhook, webhookHost dengan bawaan 127.0.0.1, webhookPort dengan bawaan 8787, dan webhookCertPath berupa PEM sertifikat swatandatangan untuk penyiapan IP langsung atau tanpa domain.

> **SATU RUTE YANG SUDAH DIPESAN**
> Listener memesan /healthz untuk pemeriksaan kesehatan, sehingga webhookPath harus memakai rute yang berbeda. Bila penyiapan yang sudah ada memakai /healthz, pilih rute lain, perbarui jalur pada webhookUrl dan pemetaan reverse proxy, lalu restart OpenClaw.

Pada mode long polling, OpenClaw menyimpan posisi restart-nya setelah sebuah update dikomitkan ke antrean ingress yang tahan lama. Handler yang gagal tetap dapat dicoba ulang dari antrean itu. Listener lokal terikat ke 127.0.0.1:8787 secara bawaan.

BAB 28 / PRAKTIKUM

Bot Telegram dari nol sampai terkunci

Telegram paling cepat dipasang di seluruh ekosistem OpenClaw, dan kecepatan itu membuat banyak orang berhenti sebelum mengamankannya. Praktikum ini menambahkan langkah-langkah yang biasanya dilewati.

Langkah 1 — buat token dan simpan dengan benar

```
# Di Telegram: bicara dengan @BotFather, jalankan /newbot
# Lalu simpan token — jangan tempel ke konfigurasi bila memungkinkan
openclaw config get channels.telegram.botToken
```

Nilai konfigurasi menang atas variabel lingkungan; env hanya untuk akun default.

Langkah 2 — temukan ID pengguna tanpa bot pihak ketiga

```
# Kirim DM ke bot Anda, lalu:
openclaw logs --follow
# baca medan pengirim pada pesan yang masuk
```

Cara ini tidak menyerahkan identitas Anda kepada layanan pihak ketiga mana pun.

Langkah 3 — kunci akses sebelum membagikan nama bot

```
{
  channels: { telegram: {
    enabled: true,
    dmPolicy: "pairing",
    groups: { "*": { requireMention: true } },
  } },
}
```

Telegram tidak memakai channels login; setel token lalu jalankan Gateway.

Langkah 4 — putuskan transport dengan sadar

Pertanyaan	Bila ya	Bila tidak
Punya ingress HTTPS yang stabil?	Webhook layak dipertimbangkan	Tetap long polling
Perlu lebih dari satu instans?	Webhook	Long polling; ia konsumen tunggal
Di balik NAT tanpa port forwarding?	Long polling	Keduanya bisa
Sudah memakai /healthz di ingress?	Pilih webhookPath lain	Bawaannya aman dipakai

BAB 28 / STUDI KASUS

Dua instans, satu token

Sebuah tim menjalankan bot Telegram di server produksi. Seorang insinyur menyalin konfigurasi ke mesin pengembangannya untuk menguji sesuatu, termasuk token bot yang sama, lalu menjalankan Gateway di sana.

Sejak itu pesan pengguna kadang dijawab dua kali, kadang tidak dijawab sama sekali, dan kadang dijawab oleh agen dengan konfigurasi pengembangan yang belum selesai. Tidak ada galat di kedua mesin, karena keduanya bekerja persis seperti yang dirancang.

YANG TAMPAK		YANG TERJADI
Satu	**Jumlah bot**	Satu token, dua konsumen
Bug	**Balasan ganda**	Dua instans menjawab
Bug	**Pesan hilang**	Diambil instans lain
Model bermasalah	**Jawaban aneh**	Konfigurasi pengembangan
Tidak ada	**Galat**	Memang tidak ada

Gambar 28.2 — Long polling bersifat konsumen tunggal. Dua instans dengan token yang sama memperebutkan pembaruan.

Penyelesaiannya satu baris: hentikan instans pengembangan, atau beri ia token bot sendiri. Yang mahal adalah empat hari ketika tim mencurigai model, jaringan, dan penyedia sebelum seseorang bertanya apakah ada instans kedua.

> **SATU TOKEN, SATU INSTANS**
> Token bot adalah identitas, bukan sekadar kredensial. Satu token untuk satu instans yang berjalan. Untuk lingkungan pengembangan, buat bot terpisah lewat BotFather — prosesnya dua menit.

BAB 28 / LATIHAN

Latihan mandiri

1. Buat bot Telegram dan temukan ID pengguna Anda tanpa memakai bot pihak ketiga.
2. Kunci akses DM dan grup sebelum membagikan nama bot Anda kepada siapa pun.
3. Jawab keempat pertanyaan transport pada praktikum langkah empat untuk situasi Anda, lalu tuliskan pilihannya beserta alasan.
4. Buat bot kedua untuk lingkungan pengembangan Anda, dan pastikan tidak ada token yang dipakai di dua tempat.

BAB 28 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
channels login telegram tidak dikenali	Telegram memang tidak memakainya. Setel token lalu jalankan Gateway.
Token dari env diabaikan	Nilai konfigurasi menang; env hanya untuk akun default.
Webhook tidak pernah menerima apa pun	Periksa webhookPath; /healthz sudah dipesan dan tidak boleh dipakai.
Pesan terduplikasi	Long polling bersifat konsumen tunggal. Jalankan satu instans per token.
Allowlist @username tidak cocok	Jalankan doctor --fix untuk menyelesaikannya menjadi ID numerik.

Tiga pertanyaan review

1. Mengapa long polling menjadi bawaan meski webhook lebih efisien secara jaringan?
2. Apa risiko menjalankan dua instans dengan token bot yang sama pada mode long polling?
3. Kapan Anda akan memilih webhook meski pemasangannya lebih sulit?

Kunci pembahasan

Long polling menjadi bawaan karena ia bekerja tanpa URL publik, tanpa reverse proxy, dan di balik NAT — kondisi yang berlaku untuk hampir semua pemasangan pribadi dan kantor kecil. Menjalankan dua instans dengan token yang sama pada long polling menghasilkan perebutan update sehingga pesan dapat terduplikasi atau hilang, karena transport itu bersifat konsumen tunggal. Webhook pantas dipilih ketika Anda sudah memiliki ingress HTTPS dan membutuhkan penskalaan atau latensi yang lebih rendah, misalnya pada layanan pelanggan bervolume tinggi.

Rujukan: dokumentasi OpenClaw, halaman channels/telegram, channels/telegram/setup, dan channels/telegram/transports, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 29 Telegram: grup, topik forum, dan pesan kaya

Topik forum Telegram adalah fitur yang paling kurang dimanfaatkan dalam penerapan OpenClaw. Ia memungkinkan satu grup menampung beberapa ruang kerja dengan kewenangan, skill, dan prompt yang berbeda — tanpa menjalankan beberapa bot.

Tujuan belajar

Mengonfigurasi grup dan topik secara terpisah, memahami pewarisan setelan, dan memakai pesan kaya beserta persetujuan. Pada akhir bab, pembaca dapat membangun satu grup operasi dengan beberapa tingkat kewenangan.

Sesi dan isolasi

Perutean bersifat deterministik: masukan Telegram dibalas kembali ke Telegram, dan model tidak memilih kanal. DM berbagi sesi utama agen, sementara sesi grup terisolasi berdasarkan ID grup. Topik forum menambahkan akhiran topik pada kunci sesi supaya topik tetap terpisah satu sama lain.

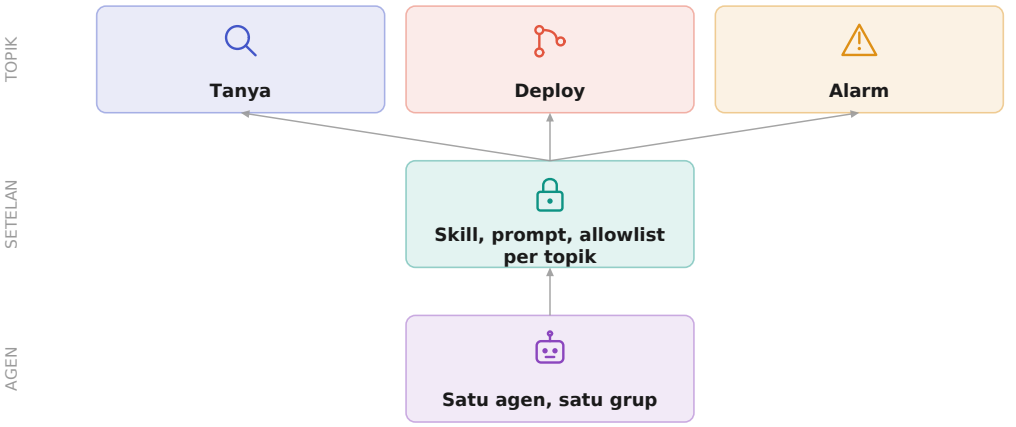
> **TOPIK UMUM BERPERILAKU BERBEDA**
> Topik umum memiliki thread id 1 dan bersifat khusus: pengiriman pesan menghilangkan message_thread_id karena Telegram menolaknya, tetapi indikator mengetik tetap menyertakannya. Grup forum dirutekan ke sesi topik umum, bukan ke topik asal pesannya, pada sebagian kasus.

Konfigurasi bertingkat

```
{
  channels: {
    telegram: {
      groups: {
        "*": { requireMention: true },
        "-1001234567890": {
          allowFrom: ["@admin"],
          systemPrompt: "Jawab singkat.",
          topics: {
            "99": {
              requireMention: false,
              skills: ["search"],
              systemPrompt: "Tetap pada topik.",
            },
          },
        },
      },
    },
  },
}
```

Konfigurasi topik mewarisi setelan grup kecuali ditimpa per topik.

Konfigurasi khusus topik tersedia di bawah channels.telegram.groups lalu chatId lalu topics lalu threadId, mencakup skill, daftar izin, balasan otomatis, prompt sistem, dan penonaktifan. Konfigurasi topik mewarisi setelan grup — requireMention, daftar izin, skill, prompt, dan status aktif — kecuali ditimpa per topik.



Gambar 29.1 — Satu grup, tiga ruang kerja. Inilah cara membangun beberapa tingkat kewenangan tanpa beberapa bot.

BAB 29 / PESAN KAYA

Tombol, aksi, dan persetujuan

Telegram mendukung papan ketik inline dengan tombol callback. Cakupannya dikendalikan supaya tombol tidak muncul di tempat yang tidak Anda inginkan.

Nilai	Perilaku
off	Tombol inline dimatikan
dm	Hanya DM; target grup diblokir
group	Hanya grup; target DM diblokir
all	DM dan grup
allowlist	DM dan grup, hanya pengirim yang diizinkan allowFrom atau groupAllowFrom

```
{
  channels: {
    telegram: {
      replyToMode: "first",
      linkPreview: true,
      streaming: { mode: "partial" },
      actions: { reactions: true, sendMessage: true },
      reactionNotifications: "own",
      historyLimit: 50,
      mediaMaxMb: 100,
      customCommands: [
        { command: "backup", description: "Cadangan Git" },
      ],
    },
  },
}
```

replyToMode menerima off, first, all, atau batched; streaming.mode menerima off, partial, block, atau progress.

Persetujuan eksekusi dapat ditampilkan sebagai tombol di dalam obrolan. Ini membuat gerbang persetujuan yang dibahas di Bab 44 menjadi sesuatu yang benar-benar dipakai — bukan sekadar mekanisme yang ada tetapi diabaikan karena merepotkan.

BAB 29 / PRAKTIKUM

Membangun satu grup dengan tiga kewenangan

Praktikum ini membangun struktur yang akan dipakai lagi di Bab 61. Kerjakan pada grup uji lebih dulu, karena kesalahan pewarisan setelan paling mudah terlihat saat Anda sengaja mencarinya.

Langkah 1 — siapkan grup forum dan catat ID-nya

```
# Jadikan grup Telegram sebagai forum, buat tiga topik
openclaw logs --follow
# kirim satu pesan di tiap topik dan baca chatId serta threadId-nya
```

Topik umum memiliki thread id 1 dan berperilaku berbeda dari topik lainnya.

Langkah 2 — susun konfigurasi bertingkat

```
{
  channels: { telegram: { groups: {
    "*": { requireMention: true },
    "-100xxxxxxxxx": {
      allowFrom: ["@admin"],
      topics: {
        "11": { skills: ["status"] },
        "12": { skills: ["runbook"], allowFrom: ["@senior"] },
        "13": { skills: [], requireMention: false },
      },
    },
  } } },
}
```

Topik mewarisi setelan grup kecuali ditimpa; topik 12 mempersempit daftar izin.

Langkah 3 — buktikan pewarisan bekerja seperti yang Anda kira

Uji	Cara	Hasil yang benar
Pewarisan berlaku	Kirim di topik yang tidak dikonfigurasi	Memakai setelan grup
Penimpaan berlaku	Kirim di topik 12 sebagai orang di luar daftar	Ditolak meski ada di daftar grup
Sesi terpisah	Jalankan openclaw sessions	Kunci sesi memuat akhiran topik
Topik umum khusus	Kirim di topik umum	Perilaku pengiriman berbeda

Langkah 4 — setel cakupan tombol dengan sengaja

Tombol inline sangat berguna untuk persetujuan, dan sangat mengganggu bila muncul di tempat yang salah. Setel cakupannya ke allowlist bila tombol memang diperlukan di grup tetapi hanya sebagian anggota yang berhak menekannya.

BAB 29 / STUDI KASUS

Topik baru yang lahir terbuka

Sebuah tim operasi menyiapkan grup forum dengan tiga topik dan kewenangan yang berbeda-beda. Konfigurasinya rapi dan sudah diuji. Beberapa minggu kemudian seorang anggota tim membuat topik keempat untuk membahas proyek baru.

Topik itu tidak ada di konfigurasi, sehingga ia mewarisi setelan grup. Setelan grup memuat daftar izin seluruh anggota tim dan skill lengkap — karena topik yang sempit diatur lewat penimpaan, bukan lewat pembatasan di tingkat grup.

YANG DIKIRA		YANG TERJADI
Tidak aktif	**Topik baru**	Aktif dengan setelan grup
Kosong	**Skill**	Seluruh skill grup
Sempit	**Daftar izin**	Seluruh anggota grup
Wajib	**Persetujuan**	Mengikuti setelan grup
Segera	**Terdeteksi**	Saat audit tiga minggu kemudian

Gambar 29.2 — Pewarisan yang menguntungkan pada topik yang sudah dirancang menjadi celah pada topik yang belum.

Pewarisan itu sendiri adalah rancangan yang baik — tanpa itu, setiap topik baru lahir tanpa setelan sama sekali. Yang perlu disesuaikan adalah setelan tingkat grup: buat ia seketat mungkin, lalu longgarkan per topik yang memang membutuhkannya.

> **PRINSIP YANG MEMBALIK RANCANGAN**
> Rancang setelan grup sebagai lantai terketat, bukan sebagai rata-rata. Dengan begitu, topik apa pun yang lahir tanpa konfigurasi tetap aman sampai seseorang sengaja melonggarkannya.

BAB 29 / LATIHAN

Latihan mandiri

1. Bangun grup forum uji dengan tiga topik dan jalankan keempat uji pada praktikum langkah tiga.
2. Buat topik keempat tanpa mengonfigurasinya, lalu periksa kewenangan apa yang ia warisi.
3. Rancang ulang setelan tingkat grup Anda sebagai lantai terketat, lalu longgarkan per topik seperlunya.
4. Tetapkan cakupan tombol inline untuk tiap topik dan tuliskan alasan masing-masing.

BAB 29 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Setelan topik tidak berlaku	Topik mewarisi grup; pastikan kunci threadId benar dan ditimpa dengan sengaja.
Pengiriman ke topik umum ditolak	message_thread_id dihilangkan di sana; itu perilaku yang benar.
Tombol inline muncul di grup pelanggan	Setel cakupan tombol ke dm atau allowlist.
Balasan menyebut pesan yang salah	Periksa replyToMode; first, all, dan batched berperilaku berbeda.
Semua topik berbagi sesi	Kunci sesi topik hanya terbentuk pada grup forum yang benar.

Tiga pertanyaan review

1. Mengapa pewarisan setelan dari grup ke topik lebih baik daripada menuntut setiap topik dikonfigurasi penuh?
2. Rancang tiga topik untuk tim operasi IT beserta tingkat kewenangan masing-masing.
3. Kapan cakupan tombol allowlist lebih tepat daripada dm?

Kunci pembahasan

Pewarisan membuat setelan aman berlaku secara bawaan ke seluruh topik, sehingga topik baru yang dibuat seseorang tidak lahir dalam keadaan terbuka; tanpa pewarisan, setiap topik baru menjadi celah sampai seseorang ingat mengonfigurasinya. Tiga topik untuk tim operasi: Tanya dengan skill baca saja tanpa mention wajib, Deploy dengan daftar izin sempit dan persetujuan wajib, dan Alarm yang hanya menerima dan tidak mengeksekusi apa pun. Cakupan allowlist lebih tepat ketika tombol memang diperlukan di grup, misalnya untuk persetujuan eksekusi, tetapi hanya sebagian anggota grup yang berhak menekannya.

Rujukan: dokumentasi OpenClaw, halaman channels/telegram/threads-and-sessions, channels/telegram/rich-messages, dan gateway/config-channels/personal-messaging, ditinjau 17 September 2026.

KANAL PERCAKAPAN

Bab 30 Slack, Discord, Signal, dan kanal kerja lain

Kanal pelanggan dan kanal internal menuntut sikap yang berbeda. Bab ini membahas kelompok kedua: tempat tim Anda sudah bekerja setiap hari, dan tempat agen paling mudah diterima karena ia tidak menuntut siapa pun mengubah kebiasaan.

Tujuan belajar

Memahami bentuk daftar izin yang berbeda di tiap kanal kerja, memilih transport Slack, dan mengetahui kapan Signal menjadi jawaban. Pada akhir bab, pembaca dapat menempatkan agen di ruang kerja tim dengan batas yang jelas.

Bentuk daftar izin berbeda-beda

Kanal	Bentuk allowlist grup
Discord	channels.discord.guilds.<id>.channels
Slack	channels.slack.channels
Matrix	channels.matrix.groups, memakai ID ruang, alias, atau nama
WhatsApp, Telegram, Signal, iMessage, Teams, Zalo	groupAllowFrom

DM grup dikendalikan terpisah lewat channels.discord.dm dan channels.slack.dm. Pembatasan tool per grup atau per kanal diterapkan sebagai tambahan atas kebijakan tool global dan kebijakan tool agen — dan penolakan tetap menang.

```
{
  channels: {
    slack: {
      groupPolicy: "allowlist",
      channels: { "#general": { enabled: true } },
    },
    discord: {
      groupPolicy: "allowlist",
      guilds: { GUILD_ID: { channels: { help: { enabled: true } } } },
    },
  },
}
```

Dua kanal kerja dengan bentuk penyarangan yang berbeda untuk hal yang sama.

BAB 30 / SLACK

Tiga transport Slack

Gambar 30.1 — Tiga transport Slack. Pilihannya mengikuti apakah Anda memiliki ingress HTTPS yang stabil, bukan preferensi.

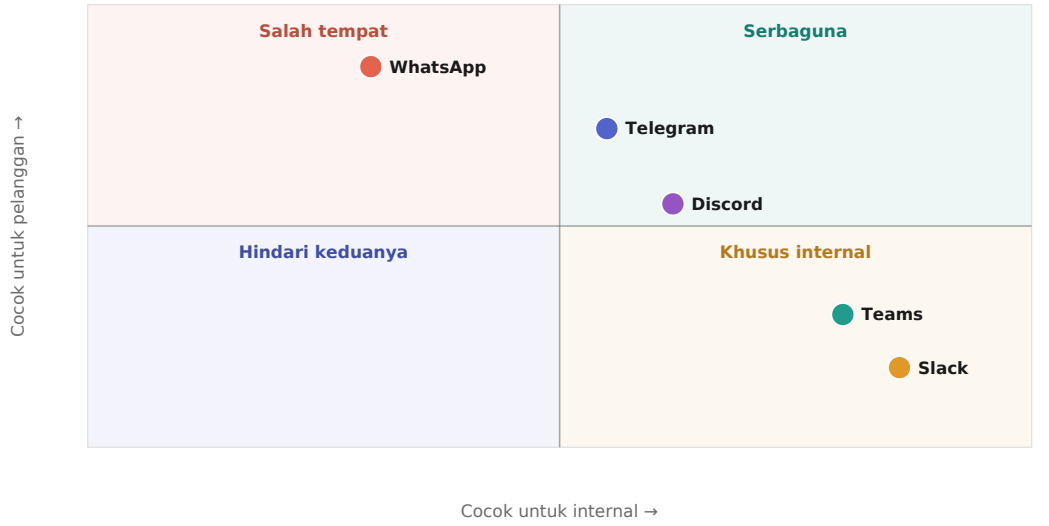
Slack juga membawa kemampuan yang tidak dimiliki kanal pesan pribadi: bagan dan tabel Block Kit native, pengiriman modal, dan tombol persetujuan. Untuk tim yang menuntut laporan terbaca rapi, ini alasan kuat memilih Slack dibanding kanal lain.

Discord

Discord menambahkan dimensi yang jarang dipakai perusahaan tetapi berharga untuk komunitas: kanal suara, transkrip rapat, dan komponen interaktif. Ia juga membawa kontrol akses yang paling berlapis — kebijakan DM, daftar izin guild, gerbang mention, perutean berbasis peran, otorisasi perintah, dan gerbang aksi.

BAB 30 / MEMILIH

Kanal internal versus kanal pelanggan



Gambar 30.2 — Penempatan kanal pada dua kebutuhan. Slack dan Teams nyaris tidak pernah tepat sebagai kanal pelanggan ritel.

Signal menempati kategorinya sendiri. Ia dipilih ketika privasi peserta adalah syarat utama — misalnya kanal pelaporan internal atau komunikasi dengan pihak yang membutuhkan jaminan kerahasiaan. Pemasangannya lewat signal-cli, dan modelnya berbasis nomor, bukan berbasis bot.

ATURAN YANG TIDAK BOLEH DILANGGAR

Satu prinsip yang berlaku di semua kanal kerja: agen yang berada di ruang internal hampir selalu punya kewenangan lebih besar daripada agen pelanggan. Ikat keduanya ke agen yang berbeda seperti pada Bab 9, dan jangan pernah membiarkan satu agen melayani keduanya.

BAB 30 / PRAKTIKUM

Menempatkan agen di ruang kerja tim

Kanal internal terasa lebih aman daripada kanal pelanggan, dan justru karena itu batasnya sering lebih longgar daripada seharusnya. Praktikum ini menyusunnya dengan kesungguhan yang sama.

Langkah 1 — pahami bentuk allowlist tiap kanal

```
{
  channels: {
    slack:   { groupPolicy: "allowlist",
               channels: { "#ops": { enabled: true } } },
    discord: { groupPolicy: "allowlist",
               guilds: { GUILD_ID: { channels: { help: { enabled: true } } } } },
  },
}
```

Bentuk penyarangannya berbeda untuk hal yang sama; menyalin antar kanal menghasilkan konfigurasi yang diam-diam tidak berlaku.

Langkah 2 — pilih transport Slack berdasarkan keadaan, bukan selera

Keadaan Anda	Transport	Alasan
Tidak punya ingress HTTPS	Socket Mode	Tidak butuh URL publik
Punya ingress dan volume naik	HTTP Request URL	Lebih mudah diskalakan
Topologi tidak mendukung keduanya	Relay	Jalur terakhir

Langkah 3 — pisahkan agen internal dari agen pelanggan

Ini langkah yang paling menentukan dan paling sering dilewati karena terasa berlebihan untuk tim kecil. Agen internal biasanya memiliki akses ke data dan tool yang tidak boleh dapat dijangkau dari kanal eksternal mana pun.

```
openclaw agents add internal --workspace ~/.openclaw/workspace-internal
openclaw agents bind --agent internal --bind slack:<akun>
openclaw agents bindings
```

Verifikasi pengikatan sesudahnya; lihat studi kasus Bab 9.

Langkah 4 — uji bahwa pembatasan hanya memperketat

Coba buka sebuah tool di tingkat kanal yang sudah ditolak di tingkat global. Ia tidak akan terbuka. Membuktikan ini sendiri sekali akan menghemat perdebatan panjang di kemudian hari tentang apa yang sebenarnya dapat dilakukan agen.

BAB 30 / STUDI KASUS

Satu agen yang melayani dua dunia

Sebuah perusahaan kecil menjalankan satu agen yang melayani Slack internal sekaligus WhatsApp pelanggan. Alasannya masuk akal pada saat itu: timnya enam orang, dan dua agen terasa seperti kerumitan yang tidak perlu.

Agen itu memiliki akses ke CRM karena tim internal membutuhkannya. Suatu hari seorang pelanggan menanyakan sesuatu yang membuat agen memanggil tool CRM dan mengutip informasi tentang pelanggan lain — nama perusahaan, nilai transaksi terakhir, dan catatan internal penjual.

Akar masalah	Mengapa konfigurasi tidak menolongnya
Satu agen, dua batas kepercayaan	Kebijakan tool berlaku per agen, bukan per pesan
Tool CRM dibutuhkan tim internal	Menolaknya akan mematikan nilai untuk tim
Instruksi prompt melarang	Prompt bukan kontrol; lihat Bab 51
Tidak ada pemisahan pengikatan	Kedua kanal jatuh ke agen yang sama

Perbaikannya adalah yang seharusnya dilakukan sejak awal: dua agen, dua workspace, dua daftar skill, dan pengikatan yang memastikan kanal pelanggan tidak pernah menyentuh agen yang memiliki tool CRM. Pekerjaannya setengah hari.

> **ATURAN YANG BERLAKU DI SELURUH BAGIAN IX**
> Menjalankan dua agen bukan kerumitan; menjalankan satu agen untuk dua batas kepercayaan adalah kerumitan yang ditunda. Biaya pemisahan dibayar sekali di awal; biaya tidak memisahkan dibayar pada hari yang tidak Anda pilih.

BAB 30 / LATIHAN

Latihan mandiri

1. Tuliskan bentuk allowlist grup untuk setiap kanal yang Anda pakai, lalu simpan sebagai rujukan supaya tidak menyalin yang keliru.
2. Pilih transport Slack Anda berdasarkan tabel praktikum langkah dua, bukan berdasarkan kemudahan.
3. Periksa apakah ada satu agen di penyiapan Anda yang melayani lebih dari satu batas kepercayaan. Bila ada, rancang pemisahannya.
4. Buktikan sendiri bahwa pembatasan tool tingkat kanal hanya dapat memperketat.

BAB 30 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Konfigurasi allowlist disalin antar kanal dan gagal	Bentuk penyarangannya berbeda. Periksa halaman kanal masing-masing.
Slack diam di kanal tertentu	channels.slack.channels belum memuat kanal itu.
Pembatasan tool per kanal tidak melonggarkan apa pun	Ia hanya menambah pembatasan; penolakan global tetap menang.
Socket Mode tidak cukup saat volume naik	Pindah ke HTTP Request URL bila ingress HTTPS tersedia.
Satu agen melayani kanal internal dan pelanggan	Pisahkan agennya. Kewenangannya tidak boleh sama.

Tiga pertanyaan review

1. Mengapa pembatasan tool per kanal hanya dapat memperketat, tidak pernah melonggarkan?
2. Kapan Socket Mode cukup, dan sinyal apa yang menandakan waktunya berpindah?
3. Rancang penempatan kanal untuk perusahaan dengan 40 karyawan, pelanggan ritel, dan kanal pelaporan pelanggaran.

Kunci pembahasan

Pembatasan per kanal hanya memperketat karena bila ia dapat melonggarkan, maka seseorang yang menguasai satu kanal dapat memberi dirinya kewenangan yang sengaja ditolak di tingkat global — kebijakan keamanan akan kehilangan artinya. Socket Mode cukup selama volume rendah dan Anda tidak memiliki ingress HTTPS; sinyal untuk berpindah adalah latensi yang naik atau kebutuhan menjalankan lebih dari satu instans. Untuk perusahaan itu: Slack untuk 40 karyawan karena pekerjaan sudah hidup di sana,

WhatsApp untuk pelanggan ritel karena di sanalah mereka berada, dan Signal untuk pelaporan pelanggaran karena kerahasiaan pelapor adalah syarat — dengan tiga agen terpisah dan tidak ada kewenangan yang dibagi di antaranya.

Rujukan: dokumentasi OpenClaw, halaman channels/slack, channels/discord, channels/signal, dan channels/groups, ditinjau 17 September 2026.

SKILLS DAN CLAWHUB

Bab 31 Anatomi skill dan format folder

Skill adalah berkas Markdown berisi instruksi. Ia tidak menambah kemampuan eksekusi apa pun; ia menambah pengetahuan tentang cara memakai kemampuan yang sudah ada. Perbedaan satu kalimat ini menentukan hampir semua keputusan desain berikutnya.

Tujuan belajar

Membaca struktur folder skill, menulis frontmatter yang benar, dan memahami mengapa deklarasi harus cocok dengan perilaku. Pada akhir bab, pembaca dapat menulis skill yang lolos analisis keamanan registri.

Bentuk dasarnya

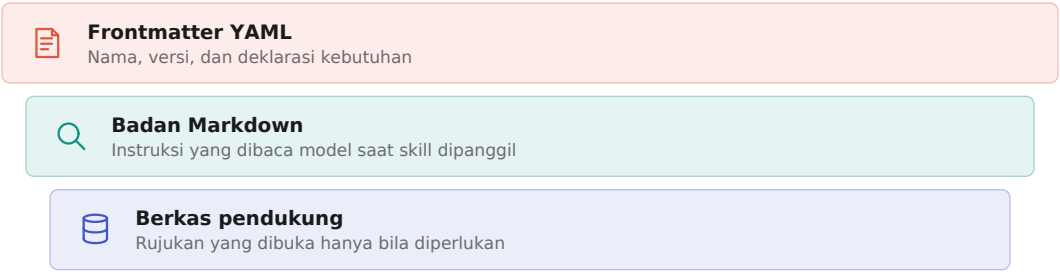
Sebuah skill adalah folder dengan satu berkas SKILL.md yang wajib ada, ditambah berkas pendukung yang opsional. Metadata skill dideklarasikan pada frontmatter YAML di bagian atas SKILL.md. Frontmatter itulah yang memberi tahu registri — dan analisis keamanannya — apa yang dibutuhkan skill Anda untuk berjalan.

```
---
name: todoist-cli
version: 1.2.0
metadata:
  openclaw:
    requires:
      env:
        - TODOIST_API_KEY
      bins:
        - todoist-cli
      install:
        node: todoist-cli
---
# Todoist CLI Skill

Ketika pengguna meminta mengelola tugas, pakai biner `todoist-cli`...
```

Blok metadata mendeklarasikan kebutuhan skill: variabel lingkungan, biner, dan perintah pemasangan.

OpenClaw memeriksa deklarasi ini pada saat muat dan memperingatkan Anda bila ada yang hilang. Persyaratan runtime dideklarasikan di bawah metadata.openclaw, dengan alias untuk nama-nama lama proyek ini.



Gambar 31.1 — Tiga lapis sebuah skill. Hanya lapis pertama yang selalu ada di konteks; dua lainnya dimuat sesuai kebutuhan.

BAB 31 / DEKLARASI

Mengapa deklarasi harus jujur

DEKLARASI DICOCOKKAN DENGAN PERILAKU
Analisis keamanan ClawHub memeriksa apakah yang dideklarasikan skill Anda cocok dengan apa yang benar-benar dilakukannya. Bila kode Anda merujuk TODOIST_API_KEY tetapi frontmatter tidak mendeklarasikannya di bawah requires.env, primaryEnv, atau envVars, analisis akan menandainya.

Pemeriksaan ini bukan formalitas birokrasi. Skill yang diam-diam membaca variabel lingkungan yang tidak dideklarasikan adalah pola serangan yang nyata: ia terlihat polos pada tinjauan cepat, lalu mengambil kredensial yang kebetulan ada di lingkungan agen.

Batas dan aturan penamaan

Aturan	Ketentuan
Ukuran bundel total	50 MB
Teks yang disematkan	SKILL.md ditambah sekitar 40 berkas UTF-8 terbatas
Nama skill portabel	Cocok dengan nama direktori induk, 1–64 karakter, huruf kecil, angka, atau tanda hubung
Scope paket	Harus cocok persis dengan handle penerbit ClawHub
Slug paket	Huruf kecil dan aman untuk npm
Lisensi	Seluruh skill yang diterbitkan di ClawHub berlisensi MIT-0

Baris terakhir layak diperhatikan sebelum Anda menerbitkan apa pun. MIT-0 berarti siapa pun boleh memakai, memodifikasi, dan mendistribusikan ulang skill yang Anda terbitkan — termasuk secara komersial — dan atribusi tidak diwajibkan. Jangan menerbitkan sesuatu yang Anda anggap sebagai keunggulan kompetitif.

BAB 31 / PRAKTIKUM

Menulis skill pertama yang lolos analisis

Praktikum ini menulis satu skill kecil dari nol, lengkap dengan deklarasi yang jujur. Skill yang deklarasinya tidak cocok dengan perilakunya akan ditandai, dan lebih baik Anda mengetahui bentuk pemeriksaannya sekarang.

Langkah 1 — buat struktur folder

```
mkdir -p ~/.openclaw/skills/ringkas-tiket
cd ~/.openclaw/skills/ringkas-tiket
touch SKILL.md
```

Nama folder dan medan name pada frontmatter sebaiknya sama.

Langkah 2 — tulis frontmatter yang jujur

```
---
name: ringkas-tiket
version: 1.0.0
description: Meringkas tiket dukungan menjadi tiga poin terstruktur
metadata:
  openclaw:
    requires:
      env:
        - TIKET_API_KEY
---
# Ringkas Tiket

Ketika pengguna meminta ringkasan tiket, ambil datanya lewat tool
yang tersedia, lalu susun tiga poin: masalah, yang sudah dicoba,
dan langkah berikutnya. Jangan menyimpulkan apa pun yang tidak ada
pada keluaran tool.
```

Setiap variabel lingkungan yang dipakai harus dideklarasikan; analisis keamanan memeriksa kecocokannya.

Langkah 3 — uji apakah ia terpicu

Gejala	Penyebab yang paling mungkin
Tidak pernah terpicu	Deskripsi terlalu kabur untuk dipilih model
Terpicu terlalu sering	Deskripsi terlalu luas
Terpicu tetapi tidak membantu	Instruksi tidak menyebut langkah konkret
Peringatan saat dimuat	Kebutuhan dideklarasikan tetapi tidak ada di mesin

Deskripsi adalah satu-satunya bagian skill yang dilihat model sebelum memutuskan memuatnya. Menulisnya sebagai kalimat yang menyebut kapan skill ini relevan jauh lebih efektif daripada menyebut apa isinya.

Langkah 4 — periksa sebelum berpikir menerbitkan

```
ls -la ~/.openclaw/skills/ringkas-tiket
grep -rn 'API_KEY\|token\|password' .
```

Penerbitan menyertakan seluruh berkas biasa di folder; periksa sebelum, bukan sesudah.

BAB 31 / STUDI KASUS

Berkas catatan yang ikut terbit

Seorang pengembang menulis skill untuk mengotomatiskan alur kerja internal. Selama pengembangan ia menyimpan berkas catatan di folder yang sama: contoh respons API, beberapa potongan konfigurasi, dan satu berkas berisi kredensial uji.

Ketika skill itu diterbitkan, seluruh berkas biasa di folder ikut. Kredensial uji itu memang untuk lingkungan pengembangan, tetapi ia juga memperlihatkan pola penamaan endpoint internal perusahaan dan struktur data pelanggannya.

- Menjalankan --dry-run dan membaca daftar berkas yang akan terbit
- Menyimpan catatan pengembangan di luar folder skill
- Memeriksa isi folder dengan grep sebelum menerbitkan
- Menganggap hanya SKILL.md yang ikut terbit
- Berasumsi penerbitan dapat dibatalkan

Gambar 31.2 — Lima kebiasaan. Dua yang terakhir adalah asumsi yang keliru, dan keduanya tidak dapat diperbaiki setelah terbit.

Skill itu ditarik dari katalog dalam beberapa jam. Tetapi lisensi MIT-0 berlaku pada salinan yang sudah diunduh, dan tidak ada cara mengetahui berapa banyak yang sudah mengambilnya dalam rentang itu.

> **KEBIASAAN YANG MENCEGAH SELURUH KELAS MASALAH INI**
> Perlakukan folder skill seperti folder yang akan Anda unggah ke repositori publik, karena secara efektif memang itu yang terjadi. Simpan catatan pengembangan di tempat lain sejak hari pertama.

BAB 31 / LATIHAN

Latihan mandiri

1. Tulis satu skill kecil untuk pekerjaan yang benar-benar Anda ulangi, lengkap dengan deklarasi kebutuhan yang jujur.
2. Uji apakah ia terpicu pada saat yang tepat, lalu perbaiki deskripsinya sampai pemicuannya akurat.
3. Periksa folder skill Anda dengan grep untuk memastikan tidak ada rahasia atau jalur internal di dalamnya.
4. Tuliskan tiga skill yang menurut Anda berguna untuk organisasi Anda, lalu tentukan mana yang layak diterbitkan publik dan mana yang tidak.

BAB 31 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Skill terpasang tetapi tidak pernah terpicu	Deskripsi frontmatter terlalu kabur. Model memilih berdasarkan deskripsi itu.
Analisis keamanan menandai skill Anda	Ada kebutuhan yang dipakai tetapi tidak dideklarasikan. Lengkapi requires.
Peringatan saat memuat skill	Biner atau variabel lingkungan yang dideklarasikan tidak ditemukan di mesin ini.
Penerbitan ditolak karena nama	Nama harus cocok direktori induk dan memakai huruf kecil.
Menyesal menerbitkan skill internal	MIT-0 tidak dapat ditarik. Periksa isi sebelum menerbitkan.

Tiga pertanyaan review

1. Mengapa skill tidak menambah kemampuan eksekusi, dan apa akibatnya bila Anda menulis skill untuk pekerjaan yang sebenarnya butuh tool?
2. Jelaskan pola serangan yang dicegah oleh pencocokan deklarasi dengan perilaku.
3. Perusahaan Anda ingin menerbitkan skill internal ke ClawHub agar mudah dipasang di banyak mesin. Apa yang harus dipertimbangkan lebih dulu?

Kunci pembahasan

Skill hanya berisi instruksi, sehingga menulis skill untuk pekerjaan yang butuh tool menghasilkan agen yang percaya diri mengarang — ia diberi tahu caranya, tetapi tidak punya alatnya. Pencocokan deklarasi mencegah skill yang diam-diam membaca kredensial dari lingkungan agen tanpa pernah menyebutnya, pola yang lolos tinjauan manual karena tidak terlihat mencurigakan. Sebelum menerbitkan skill internal, pertimbangkan bahwa lisensinya MIT-0 dan tidak dapat ditarik; bila isinya memuat proses bisnis khas perusahaan, pakailah distribusi internal, bukan registri publik.

Rujukan: dokumentasi OpenClaw, halaman clawhub/skill-format, ditinjau 17 September 2026.

SKILLS DAN CLAWHUB

Bab 32 Skills CLI: cari, pasang, verifikasi, perbarui

Perintah openclaw skills memeriksa skill lokal, mencari di ClawHub, memasang dari ClawHub, Git, atau direktori lokal, memverifikasi skill ClawHub, dan memperbarui pemasangan yang terlacak. Bab ini membahas urutan pemakaian yang aman.

Tujuan belajar

Menjalankan siklus penuh dari pencarian sampai pembaruan, membaca hasil verifikasi, dan mengetahui kapan sebuah pemasangan harus dihentikan. Pada akhir bab, pembaca tidak pernah memasang skill tanpa memeriksanya lebih dulu.

Gambar 32.1 — Urutan yang disarankan. Melompat langsung ke install adalah kebiasaan yang cepat sekali menjadi masalah.

```
openclaw skills search "calendar"
openclaw skills info <skill>
openclaw skills verify <skill>
openclaw skills install @openclaw/demo
openclaw skills list
openclaw skills check
openclaw skills update --all
```

OpenClaw mencatat asal skill sehingga pembaruan berikutnya tetap dapat diselesaikan lewat ClawHub.

Verifikasi membaca, bukan memindai ulang

APA ARTI LAPORAN KOSONG

Perintah verify membaca pemindaian yang tersimpan dan tidak memulai pemindaian baru. Sebuah laporan bernilai kosong ketika tidak tersedia, termasuk untuk pemindaian lama yang keluaran lengkapnya tidak dipertahankan — dan ringkasan tidak dipakai sebagai penggantinya.

Respons verifikasi dapat mencapai 64 MiB; respons yang lebih besar gagal secara eksplisit tanpa mencetak laporan sebagian. Permintaan JSON ClawHub lainnya mempertahankan batas 16 MiB.

BAB 32 / KEPERCAYAAN

Gerbang kepercayaan sebelum unduhan

Pemasangan dan pembaruan skill komunitas dari ClawHub memeriksa kepercayaan sebelum mengunduh. Rilis arsip komunitas yang berversi memakai metadata kepercayaan rilis yang persis. Skill berbasis GitHub yang melalui resolver bergantung pada resolver pemasangan ClawHub untuk menegakkan kebijakan pemindaian dan pemasangan paksa sebelum ia mengembalikan komit yang dipin.

```
openclaw skills install <skill> --force-install
```

Memasang skill berbasis GitHub yang pemindaiannya belum selesai. Pakai hanya bila Anda sudah meninjau sumbernya sendiri.

Rilis komunitas yang jahat atau diblokir ditolak. Hasil tinjauan mencetak ikhtisar audit ClawHub yang persis beserta tautan detailnya, lalu melanjutkan. Membaca tautan itu memakan tiga puluh detik dan sudah beberapa kali menyelamatkan orang dari memasang sesuatu yang tidak seharusnya dipasang.

Operasi pada skill terpasang

Perintah	Catatan
share, unshare, transfer	Menuntut <skill-id> dan --expected-revision <hash>
enable, disable, remove	Menuntut hash revisi yang diharapkan
rollback	Menuntut juga --revision <retained-hash>
--version <version>	Memilih versi ClawHub; menuntut --clawhub

Keharusan menyertakan hash revisi yang diharapkan bukan kerumitan tanpa alasan. Ia mencegah dua orang mengubah skill yang sama secara bersamaan dan saling menimpa tanpa menyadarinya — masalah yang sering muncul begitu sebuah tim mulai berbagi pustaka skill.

BAB 32 / PRAKTIKUM

Memasang skill pihak ketiga dengan disiplin

Praktikum ini menjalankan urutan lengkap pada satu skill nyata dari registri. Kerjakan seluruh langkahnya meski terasa lambat; kecepatan memasang adalah hal yang paling mudah Anda korbankan dan paling mahal akibatnya.

Langkah 1 — cari dan periksa sebelum memasang

```
openclaw skills search "<kata kunci>"
openclaw skills info <skill>
openclaw skills verify <skill>
```

verify membaca pemindaian tersimpan; ia tidak memulai pemindaian baru.

Langkah 2 — baca hasil verifikasi dengan benar

Yang Anda lihat	Artinya
Laporan lengkap	Pemindaian tersimpan dan dipertahankan
Laporan kosong	Tidak tersedia — bukan berarti aman
Peringatan kepercayaan	Tinjau sumbernya sendiri sebelum melanjutkan
Rilis ditolak	Jahat atau diblokir; jangan dipaksa

Baris kedua adalah yang paling sering disalahartikan. Laporan kosong berarti informasinya tidak tersedia, bukan bahwa pemeriksaan lulus. Ringkasan tidak dipakai sebagai penggantinya.

Langkah 3 — pasang, lalu periksa apa yang masuk

```
openclaw skills install <skill>
openclaw skills list
openclaw skills check

cat ~/.openclaw/skills/<skill>/SKILL.md
```

Perintah terakhir adalah yang paling jarang dijalankan orang dan paling berguna: bacalah instruksi yang baru saja Anda berikan kepada agen Anda.

Langkah 4 — tetapkan kebijakan pemasangan tim

- Hanya skill dengan badge audit yang boleh masuk produksi
- Setiap pemasangan ditinjau satu orang lain
- Versi dipin, dan pembaruan memicu peninjauan ulang
- Isi SKILL.md dibaca manusia sebelum dipakai
- Memasang karena jumlah unduhannya banyak

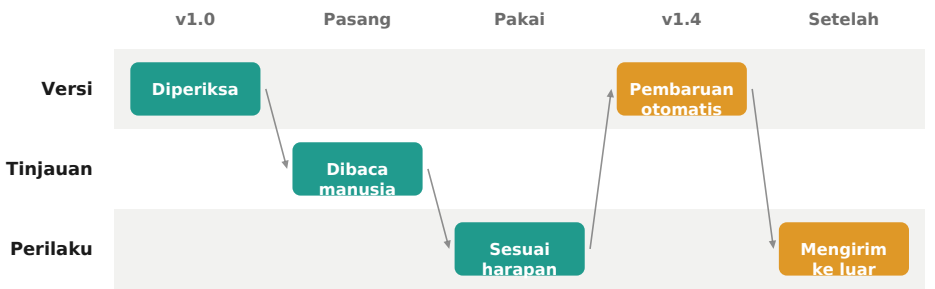
Gambar 32.2 — Lima butir kebijakan. Butir terakhir adalah sinyal popularitas, bukan sinyal keamanan.

BAB 32 / STUDI KASUS

Skill yang berubah perilakunya setelah diperbarui

Sebuah tim memasang skill komunitas untuk membantu pekerjaan riset. Mereka memeriksa isinya, memverifikasinya, dan memakainya selama beberapa bulan tanpa masalah.

Pembaruan rutin membawa versi baru. Tidak ada yang membacanya ulang, karena skill itu sudah dipercaya. Versi baru menambahkan instruksi yang meminta agen mengirim ringkasan hasil kerjanya ke sebuah endpoint — dibingkai sebagai fitur analitik.



Gambar 32.3 — Kepercayaan diberikan kepada versi, bukan kepada nama. Pembaruan adalah kode baru yang belum pernah ditinjau.

Tidak ada yang jahat secara teknis pada perubahan itu; ia dideklarasikan dan terlihat pada isinya. Yang salah adalah proses: pembaruan diperlakukan sebagai kelanjutan, padahal ia adalah kode baru yang belum pernah dibaca siapa pun di tim itu.

> **KEPERCAYAAN TIDAK DIWARISKAN OLEH PEMBARUAN**
> Pin versi skill yang dipakai di produksi, dan perlakukan setiap pembaruan sebagai pemasangan baru yang menuntut peninjauan. Kepercayaan melekat pada versi yang Anda baca, bukan pada nama skill-nya.

BAB 32 / LATIHAN

Latihan mandiri

1. Jalankan urutan lengkap cari, info, verify, pasang, periksa pada satu skill nyata.
2. Baca isi SKILL.md dari setiap skill yang sudah terpasang di mesin Anda. Catat bila ada yang mengejutkan.
3. Susun kebijakan pemasangan skill untuk tim Anda dalam lima butir yang dapat diperiksa.
4. Tetapkan cara Anda menangani pembaruan skill: apakah otomatis, dipin, atau ditinjau manual setiap kali.

BAB 32 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
verify tidak menampilkan laporan	Pemindaian lama tidak dipertahankan. Laporan kosong, bukan ringkasan pengganti.
Pemasangan ditolak	Rilis komunitas jahat atau diblokir. Jangan memaksanya.
Perintah enable ditolak	Hash revisi yang diharapkan tidak disertakan atau sudah berubah.
Respons verifikasi gagal tanpa isi	Melewati batas 64 MiB. Kegagalannya eksplisit, bukan laporan sebagian.
Pembaruan tidak menemukan sumber	Skill dipasang dari jalur yang tidak terlacak. Pasang ulang lewat ClawHub.

Tiga pertanyaan review

1. Mengapa verify sengaja tidak memulai pemindaian baru?
2. Kapan --force-install dapat dibenarkan, dan apa yang wajib Anda lakukan sebelum memakainya?
3. Rancang kebijakan pemasangan skill untuk tim berisi sepuluh insinyur.

Kunci pembahasan

verify membaca pemindaian tersimpan supaya hasilnya deterministik dan cepat; memindai ulang setiap kali akan membuat perintah itu lambat dan hasilnya berubah-ubah tanpa perubahan pada skillnya. --force-install dapat dibenarkan untuk skill internal berbasis GitHub yang Anda sendiri tulis, dan syaratnya adalah Anda benar-benar membaca komit yang dipin itu. Untuk tim sepuluh insinyur, tetapkan bahwa hanya skill yang sudah diverifikasi dan ditinjau satu orang lain yang boleh masuk ke mesin produksi, pin versinya, dan jadwalkan peninjauan ulang saat pembaruan.

Rujukan: dokumentasi OpenClaw, halaman cli/skills dan clawhub/quickstart, ditinjau 17 September 2026.

SKILLS DAN CLAWHUB

Bab 33 ClawHub: penemuan, pemindaian, dan audit keamanan

ClawHub adalah registri terbuka untuk skill dan plugin OpenClaw. Terbuka berarti siapa pun dapat menerbitkan ke sana, dan sejarah registri ini membuktikan bahwa sebagian memang melakukannya dengan niat buruk. Bab ini membahas cara memakainya tanpa menjadi korban.

Tujuan belajar

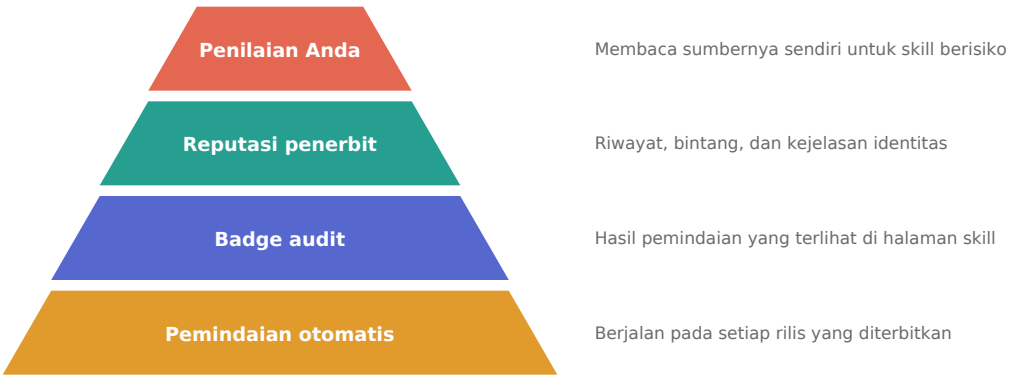
Membaca badge audit keamanan, memahami lapisan pemindaian yang berjalan, dan menetapkan kebiasaan pemeriksaan sendiri. Pada akhir bab, pembaca dapat memasang dari registri publik dengan risiko yang terukur.

Sejarah yang perlu diketahui

Registri ini pernah mengalami kampanye penyalahgunaan berskala besar, dan analisis pihak ketiga pada masa awal platform menemukan proporsi skill bermuatan jahat yang cukup tinggi. ClawHub memperketat prosesnya sebagai tanggapan. Skill yang diterbitkan kini disaring melalui beberapa lapis pemindaian, termasuk analisis semantik yang menandai instruksi tersembunyi serta skill yang perilaku sebenarnya tidak cocok dengan deskripsinya.

LAPISAN OTOMATIS TIDAK MENGGANTIKAN PENILAIAN ANDA

Penyaringan bukan jaminan. Periksa badge audit keamanan pada halaman skill, dan beri bobot pada penulis yang punya riwayat dan bintang. CLI bahkan akan menuntut Anda melewatkan tanda pengakuan risiko dalam kasus tertentu.



Gambar 33.1 — Empat lapis pertahanan. Lapis paling bawah berjalan sendiri; lapis paling atas hanya berjalan bila Anda menjalankannya.

BAB 33 / MODERASI

Apa yang terjadi pada rilis bermasalah

ClawHub menjalankan pemeriksaan otomatis pada skill dan rilis plugin yang diterbitkan. Rilis yang tertahan pemindaian atau diblokir dapat menghilang dari katalog publik dan permukaan pemasangan, sementara tetap terlihat oleh pemiliknya di dasbor.

Pengguna yang masuk dapat melaporkan skill dan paket. Moderator dapat meninjau laporan, menyembunyikan atau memulihkan konten, dan memblokir akun yang menyalahgunakan. Bagi Anda sebagai pemakai, artinya sebuah skill yang kemarin ada bisa hilang hari ini — dan itu kabar baik, bukan kerusakan.

Dua CLI, dua peran

OPENCLAW		CLAWHUB CLI
Memasang ke OpenClaw	**Peran**	Kerja registri
skills install, plugins install	**Contoh**	login, publish, rename
Tidak	**Butuh akun**	Ya
Tidak	**Menghapus dari registri**	Ya
Operator	**Dipakai sehari-hari oleh**	Penerbit

Gambar 33.2 — Pakai openclaw ketika memasang sesuatu ke dalam OpenClaw; pakai clawhub ketika bekerja dengan registrinya.

```
npm i -g clawhub
clawhub login
clawhub whoami
clawhub login --token clh_...     # lingkungan tanpa peramban
clawhub login --device            # remote atau headless
```

Masuk lewat GitHub, atau memakai token API dari antarmuka web ClawHub.

Ketika Anda menjalankan pemasangan sambil masuk, CLI dapat mengirim peristiwa pemasangan secara upaya-terbaik supaya ClawHub dapat menghitung jumlah pemasangan agregat. Ini dapat dimatikan bila Anda tidak menginginkannya.

BAB 33 / PRAKTIKUM

Menilai sebuah paket sebelum memercayainya

Praktikum ini memberi Anda urutan pemeriksaan yang dapat dijalankan dalam lima menit untuk paket apa pun dari registri terbuka. Lima menit itu adalah harga yang murah untuk menghindari kelas masalah yang tidak dapat dibatalkan.

Langkah 1 — periksa penerbitnya sebelum paketnya

Yang diperiksa	Sinyal baik	Sinyal yang menuntut kehati-hatian
Identitas penerbit	Nama atau organisasi yang dapat ditelusuri	Handle anonim tanpa jejak
Riwayat	Beberapa paket, dirawat berkelanjutan	Satu paket, baru dibuat
Respons terhadap isu	Ada dan wajar	Tidak ada sama sekali
Badge audit	Ada dan lulus	Tidak ada

Langkah 2 — baca isinya, bukan deskripsinya

```
openclaw skills info <skill>
openclaw skills verify <skill>

# setelah dipasang, yang paling penting:
cat ~/.openclaw/skills/<skill>/SKILL.md
```

Deskripsi ditulis untuk meyakinkan Anda; isi SKILL.md adalah yang benar-benar dibaca agen Anda.

Langkah 3 — cari pola yang layak dicurigai

- ⚠ Instruksi meminta mengirim data ke alamat di luar organisasi Anda
- ⚠ Instruksi meminta membaca berkas di luar cakupan tugasnya
- ⚠ Instruksi meminta menjalankan perintah yang tidak berhubungan
- ⚠ Deklarasi kebutuhan tidak cocok dengan isi instruksinya
- ⚠ Nada instruksi mendorong agen mengabaikan aturan lain

Gambar 33.3 — Lima pola. Menemukan salah satunya bukan bukti niat buruk, tetapi cukup alasan untuk tidak memasangnya di produksi.

Langkah 4 — pisahkan CLI sesuai perannya

```
openclaw skills install <skill>    # memasang ke OpenClaw
clawhub login                      # kerja registri
clawhub whoami
```

Pakai openclaw untuk memasang; pakai clawhub untuk menerbitkan dan mengelola listing.

BAB 33 / STUDI KASUS

Paket populer yang tidak pernah dibaca siapa pun

Sebuah tim memilih skill dari registri berdasarkan jumlah unduhan. Angkanya tinggi, sehingga mereka menyimpulkan banyak orang sudah memeriksanya. Skill itu dipasang di seluruh mesin tim dalam satu hari.

Beberapa bulan kemudian, seorang anggota tim membaca isi SKILL.md karena penasaran. Ia menemukan satu paragraf yang meminta agen menyertakan potongan konteks percakapan ketika memanggil sebuah layanan bantuan — dibingkai sebagai peningkatan kualitas jawaban.

ASUMSI TIM		KENYATAAN
Banyak yang memeriksa	**Jumlah unduhan tinggi**	Banyak yang berasumsi sama
Berarti aman	**Tidak ada laporan buruk**	Berarti belum ada yang membaca
Menjamin isinya	**Badge audit ada**	Memindai pola, bukan menilai maksud
Efisien	**Pemasangan cepat**	Menyebar sebelum ditinjau

Gambar 33.4 — Popularitas adalah sinyal adopsi, bukan sinyal pemeriksaan. Keduanya sering tertukar.

Tidak ada bukti data mereka benar-benar terkirim, karena tool yang dibutuhkan tidak pernah tersedia bagi agen mereka. Kebijakan tool yang ketat dari Bab 43 menyelamatkan mereka dari keputusan yang buruk di Bab 33 — persis cara pertahanan berlapis seharusnya bekerja.

> **SINYAL YANG PALING SERING DISALAHARTIKAN**
> Jumlah unduhan mengukur berapa banyak orang memasang, bukan berapa banyak yang membaca. Pada registri terbuka, keduanya adalah angka yang sangat berbeda.

BAB 33 / LATIHAN

Latihan mandiri

1. Pilih satu skill populer dari registri dan jalankan keempat langkah praktikum terhadapnya, termasuk membaca isi SKILL.md-nya.
2. Periksa setiap skill pihak ketiga yang sudah terpasang di mesin Anda terhadap lima pola pada Gambar 33.3.
3. Tuliskan kriteria penerbit yang Anda percayai, dalam bentuk yang dapat dipakai orang lain di tim Anda tanpa bertanya.
4. Rancang apa yang akan Anda lakukan bila sebuah skill yang sudah Anda pakai selama berbulan-bulan ternyata bermasalah.

BAB 33 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Skill yang dipakai tiba-tiba hilang dari katalog	Kemungkinan tertahan pemindaian atau diblokir. Periksa dasbor pemiliknya.
CLI menuntut tanda pengakuan risiko	Rilis itu membawa peringatan kepercayaan. Tinjau sebelum melanjutkan.
Badge audit tidak ada	Perlakukan sebagai belum terverifikasi, bukan sebagai aman.
Bingung memakai CLI yang mana	openclaw untuk memasang; clawhub untuk menerbitkan dan mengelola listing.
Ingin menghentikan telemetri pemasangan	Peristiwa pemasangan dapat dimatikan pada CLI ClawHub.

Tiga pertanyaan review

1. Mengapa registri terbuka menghadirkan kelas risiko yang tidak ada pada plugin bawaan?
2. Apa arti praktis dari 'skill yang perilakunya tidak cocok dengan deskripsinya' bagi Anda sebagai pemakai?
3. Susun kebijakan tiga baris untuk perusahaan yang ingin memakai skill komunitas.

Kunci pembahasan

Registri terbuka menerima kiriman dari siapa saja, sehingga tidak ada tinjauan manusia yang terjamin di balik setiap paket — berbeda dari plugin bawaan yang dikirim bersama produk. Skill yang perilakunya tidak cocok dengan deskripsi berarti instruksi yang dibaca model berbeda dari yang Anda kira Anda pasang, misalnya skill pencatat tugas yang juga menyuruh agen mengirim isi berkas ke suatu alamat. Kebijakan tiga baris: hanya skill dengan badge audit dan penerbit beridentitas jelas, wajib ditinjau satu insinyur lain sebelum masuk produksi, dan pin versi dengan peninjauan ulang pada setiap pembaruan.

Rujukan: dokumentasi OpenClaw, halaman clawhub, clawhub/security-audits, dan clawhub/moderation, ditinjau 17 September 2026. Catatan sejarah insiden berasal dari laporan pihak ketiga dan bukan dokumentasi resmi.

SKILLS DAN CLAWHUB

Bab 34 Menerbitkan skill dan mengelola namespace

Menerbitkan skill mengubah Anda dari pemakai menjadi penyedia, dan tanggung jawabnya ikut berpindah. Bab ini membahas mekanismenya sekaligus kewajiban yang menyertainya.

Tujuan belajar

Menerbitkan skill pertama, memahami penomoran versi otomatis, dan mengelola kepemilikan nama. Pada akhir bab, pembaca dapat merilis skill yang dapat dipasang orang lain dengan aman.

Menerbitkan

```
clawhub skill publish ./my-skill \
  --slug my-skill \
  --name "My Skill" \
  --changelog "Rilis pertama"

clawhub skill publish ./my-skill --dry-run
clawhub skill publish ./my-skill --version 2.0.0
```

Perintah melewati konten yang tidak berubah. Skill baru mulai di 1.0.0.

Perubahan berikutnya menerbitkan versi patch berikutnya secara otomatis. Pakai --dry-run untuk melihat pratinjau, atau --version untuk memilih versi eksplisit. Setiap penerbitan membuat versi baru mengikuti semver, dan tag adalah penunjuk string ke sebuah versi — latest adalah yang paling umum dipakai.

Sebelum menerbitkan

- ✓ Seluruh kebutuhan dideklarasikan di frontmatter
- ✓ Deskripsi menjelaskan perilaku sebenarnya, bukan yang diinginkan
- ✓ Tidak ada rahasia, token, atau jalur internal di dalam folder
- ✓ Sudah dijalankan --dry-run dan keluarannya dibaca
- ⚠ Isi memuat proses bisnis yang merupakan keunggulan kompetitif
- ⚠ Menerbitkan sambil berasumsi bisa ditarik nanti

Gambar 34.1 — Enam pemeriksaan sebelum menerbitkan. Dua yang terakhir adalah penyesalan yang tidak dapat dibatalkan.

SEMUA BERKAS IKUT TERBIT
Penerbitan menerima seluruh berkas biasa di dalam folder skill, apa pun ekstensinya. Artinya berkas yang tidak sengaja tertinggal di folder itu ikut terbit. Periksa isi direktori, bukan hanya SKILL.md.

BAB 34 / NAMESPACE

Nama, scope, dan kepemilikan

Scope paket harus cocok persis dengan handle penerbit ClawHub. Handle penerbit boleh memakai huruf kecil, angka, tanda hubung, titik, dan garis bawah; ia harus dimulai dan diakhiri dengan huruf kecil atau angka. ClawHub memisahkan slug yang dapat dirutekan dari nama tampilan katalog, sehingga nama yang sudah ada dari klien lain tetap dapat diterbitkan dan tidak diam-diam ditulis ulang.

```
clawhub skill rename <skill> <new-name>
clawhub skill merge <source> <target>

clawhub package explore --family code-plugin
clawhub package inspect <name>
clawhub package publish your-org/your-plugin --dry-run
```

Kanonikalisasi skill yang Anda miliki, dan jalur penerbitan untuk plugin.

Bila terjadi sengketa kepemilikan atas organisasi, merek, handle pemilik, scope paket, slug skill, atau namespace, tersedia jalur peninjauan ClawHub untuk menyelesaikannya. Bagi perusahaan, mendaftarkan handle lebih awal adalah langkah murah yang mencegah persoalan merek di kemudian hari.

Penghapusan dan batasnya

MENARIK KEMBALI TIDAK SEPENUHNYA MUNGKIN
Perintah uninstall pada CLI ClawHub hanya menghapus pemasangan lokal di mesin Anda. Menghapus sebuah listing dari registri adalah operasi yang berbeda, dan skill yang sudah tersebar tetap berlisensi MIT-0 di tangan orang yang sudah mengunduhnya.

BAB 34 / PRAKTIKUM

Menerbitkan tanpa menyesal

Praktikum ini menjalankan penerbitan sampai satu langkah sebelum titik tidak dapat kembali, lalu berhenti. Melihat apa yang akan terbit adalah bagian terpenting dari seluruh proses.

Langkah 1 — pisahkan folder terbit dari folder kerja

```
mkdir -p ~/publish/nama-skill
cp SKILL.md ~/publish/nama-skill/
# salin hanya berkas yang memang harus ikut
```

Menyalin yang perlu jauh lebih aman daripada menghapus yang tidak perlu.

Langkah 2 — jalankan pratinjau dan baca daftarnya

```
clawhub skill publish ~/publish/nama-skill --dry-run
```

Baca setiap nama berkas pada keluarannya. Ini satu-satunya kesempatan Anda.

Langkah 3 — periksa tiga hal yang tidak dapat ditarik

Yang diperiksa	Mengapa tidak dapat diperbaiki nanti
Rahasia atau kredensial	Salinan yang sudah diunduh tetap ada
Jalur atau nama internal	Memperlihatkan struktur organisasi Anda
Proses bisnis khas	MIT-0; siapa pun boleh memakainya komersial

Langkah 4 — terbitkan dan catat

```
clawhub skill publish ~/publish/nama-skill \
  --slug nama-skill \
  --name "Nama Skill" \
  --changelog "Rilis pertama"

clawhub whoami
```

Skill baru mulai di 1.0.0; perubahan berikutnya menerbitkan patch berikutnya otomatis.

Catat tanggal, versi, dan apa yang berubah pada setiap penerbitan. Penomoran otomatis nyaman, dan kenyamanan itu membuat orang kehilangan jejak versi mana yang memuat apa.

BAB 34 / STUDI KASUS

Keunggulan yang diberikan gratis

Sebuah perusahaan konsultan membangun skill yang mengodifikasi metodologi mereka: urutan pertanyaan diagnostik, kriteria penilaian, dan format laporan yang mereka kembangkan selama bertahun-tahun.

Untuk memudahkan pemasangan di mesin klien, mereka menerbitkannya ke registri publik. Itu memang memudahkan — satu perintah dan skill terpasang di mana pun. Tiga bulan kemudian, seorang pesaing memakai metodologi yang sama dalam proposalnya.

Yang mereka cari	Yang mereka dapat	Alternatif yang tersedia
Kemudahan pemasangan	Kemudahan pemasangan	Repositori git privat
Distribusi ke klien	Distribusi ke semua orang	Arsip yang dikirim langsung
Pembaruan otomatis	Pembaruan otomatis	Tap internal organisasi
Tidak ada	Lisensi MIT-0 permanen	—

Tidak ada yang melanggar apa pun. Seluruh skill yang diterbitkan di registri berlisensi MIT-0: siapa pun boleh memakai, memodifikasi, dan mendistribusikan ulang, termasuk secara komersial, dan atribusi tidak diwajibkan.

> **SATU PERTANYAAN SEBELUM MENERBITKAN**
> Sebelum menerbitkan, tanyakan satu hal: apakah Anda bersedia pesaing Anda memakainya besok pagi? Bila jawabannya tidak, pakailah jalur distribusi internal — kemudahannya hampir sama dan konsekuensinya jauh berbeda.

BAB 34 / LATIHAN

Latihan mandiri

1. Jalankan pratinjau penerbitan untuk satu skill Anda dan baca seluruh daftar berkas yang akan ikut.
2. Terapkan tiga pemeriksaan pada praktikum langkah tiga dan catat hasilnya.
3. Tentukan jalur distribusi untuk setiap skill internal Anda: publik, git privat, atau arsip langsung.
4. Bila organisasi Anda berencana menerbitkan, daftarkan handle penerbitnya sekarang sebelum orang lain memakainya.

BAB 34 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Berkas internal ikut terbit	Seluruh berkas biasa di folder ikut. Bersihkan direktori sebelum publish.
Versi naik sendiri padahal tidak diinginkan	Perubahan menerbitkan patch berikutnya otomatis. Pakai --version bila perlu.
Scope ditolak	Scope harus cocok persis dengan handle penerbit Anda.
Nama bentrok dengan penerbit lain	Ada jalur peninjauan sengketa namespace di ClawHub.
Ingin menarik skill yang sudah tersebar	MIT-0 berlaku pada salinan yang sudah diunduh. Tidak dapat ditarik penuh.

Tiga pertanyaan review

1. Mengapa --dry-run layak dijadikan kebiasaan wajib, bukan opsi?
2. Apa konsekuensi jangka panjang dari lisensi MIT-0 bagi strategi perusahaan Anda?
3. Rancang alur penerbitan internal supaya kesalahan tidak pernah mencapai registri publik.

Kunci pembahasan

--dry-run wajib karena penerbitan menyertakan seluruh berkas biasa di folder dan versinya naik otomatis; melihat pratinjau adalah satu-satunya kesempatan menangkap berkas yang tidak sengaja tertinggal sebelum ia menjadi publik dan berlisensi MIT-0. Konsekuensi MIT-0 adalah bahwa apa pun yang Anda terbitkan berhenti menjadi milik eksklusif perusahaan, sehingga skill yang mengandung proses bisnis khas sebaiknya didistribusikan lewat jalur internal. Alur penerbitan internal yang aman: repositori terpisah untuk skill yang akan dipublikasikan, pemeriksaan otomatis yang menolak berkas di luar daftar putih, tinjauan satu orang lain, lalu publish dari pipeline — bukan dari mesin pribadi seseorang.

Rujukan: dokumentasi OpenClaw, halaman clawhub/publishing, clawhub/namespace-claims, dan clawhub/skill-format, ditinjau 17 September 2026.

SKILLS DAN CLAWHUB

Bab 35 Workshop, library, dan skill yang dipelajari agen

Tidak semua skill datang dari registri. Sebagian lahir dari percakapan: agen mempelajari cara mengerjakan sesuatu, lalu menyimpannya supaya tidak perlu diajari lagi. Bab ini membahas mekanisme itu beserta batas kepercayaannya.

Tujuan belajar

Memahami perbedaan skill terpasang dan skill pustaka sesi, melampirkan skill ke sesi tertentu, dan menilai skill buatan agen. Pada akhir bab, pembaca dapat memakai pembelajaran agen tanpa kehilangan kendali atas perilakunya.

Dua asal skill

DARI REGISTRI		DARI WORKSHOP
ClawHub, Git, lokal	**Asal**	Dibuat dalam percakapan
Otomatis di registri	**Pemindaian**	Tidak ada
Penerbit dan moderator	**Peninjau**	Hanya Anda
Kemampuan umum	**Cocok untuk**	Proses khas Anda
Rantai pasok	**Risiko utama**	Perilaku yang tidak diaudit

Gambar 35.1 — Dua asal dengan profil risiko yang sama sekali berbeda. Skill workshop tidak lebih aman hanya karena dibuat sendiri.

Rilis v2026.9.4 membawa pembelajaran skill yang terlihat, dan v2026.9.3 membawa skill Workshop yang persisten. Keduanya mengarah ke tempat yang sama: agen yang mengingat cara mengerjakan sesuatu antar sesi, bukan hanya di dalam satu percakapan.

BAB 35 / LIBRARY

Melampirkan skill ke satu sesi

Pustaka skill memungkinkan sebuah skill dilampirkan ke sesi tertentu, bukan dinyalakan untuk seluruh agen. Ini berguna ketika sebuah pekerjaan menuntut pengetahuan khusus yang tidak seharusnya hadir di setiap percakapan.

```
openclaw skills library attach \
  --session <session-key> \
  --skill-id <skill-id> \
  --revision <revision-hash>

openclaw skills library detach --session <session-key> --skill-id <skill-id>
```

Detach memakai sesi dan skill id yang sama dengan attach.

Menghilangkan --skill-id pada operasi penyegaran akan menyegarkan seluruh skill yang terpilih, dan menuntut akses pustaka saat ini terhadap masing-masing. Operasi itu tidak pernah menggantikan pilihan dengan pustaka milik pemanggil saat ini. Setiap operasi melaporkan efeknya pada giliran berikutnya.

MENGAPA REVISI HARUS DISEBUT

Perhatikan bahwa revisi disebut secara eksplisit. Melampirkan skill berdasarkan nama saja akan membuat perilaku sesi berubah diam-diam ketika skill itu diperbarui. Menyebut revisi membuat sesi itu berperilaku tetap.

BAB 35 / TATA KELOLA

Menilai skill yang dibuat agen

Skill yang ditulis agen mudah sekali menumpuk. Setelah beberapa bulan, indeks skill bisa berisi puluhan entri yang tidak ada yang ingat asalnya. Perlakukan mereka seperti kode: mereka butuh pemilik, tinjauan, dan tanggal kedaluwarsa.

- Skill buatan agen ditinjau manusia sebelum dipakai di produksi
- Setiap skill punya pemilik bernama
- Indeks skill ditinjau berkala dan yang usang dihapus
- Revisi disebut eksplisit saat dilampirkan ke sesi penting
- Membiarkan agen menulis dan langsung memakai skill di kanal pelanggan

Gambar 35.2 — Lima aturan tata kelola. Yang terakhir adalah cara tercepat sebuah perubahan perilaku masuk produksi tanpa ada yang menyadarinya.

BAB 35 / PRAKTIKUM

Mengelola skill yang dibuat agen

Praktikum ini membangun disiplin untuk skill yang lahir dari percakapan. Tanpa disiplin ini, indeks skill Anda akan berisi puluhan entri yang tidak ada yang ingat asalnya dalam waktu kurang dari setahun.

Langkah 1 — daftar apa yang sudah ada

```
openclaw skills list
ls -lt ~/.openclaw/skills/
```

Urutkan berdasarkan waktu; skill yang lama tidak disentuh adalah kandidat peninjauan pertama.

Langkah 2 — beri tanda asal pada setiap skill

Asal	Perlakuan
Registri, terverifikasi	Pin versi; tinjau saat pembaruan
Registri, komunitas	Baca isi; tinjau setiap pembaruan
Ditulis tim Anda	Kendali versi; ada pemilik bernama
Dibuat agen di percakapan	Tinjau manusia sebelum dipakai produksi

Langkah 3 — pakai pustaka sesi untuk pengetahuan sempit

```
openclaw skills library attach \
  --session <session-key> \
  --skill-id <skill-id> \
  --revision <revision-hash>

openclaw skills library detach --session <session-key> --skill-id <skill-id>
```

Sebut revisi secara eksplisit supaya perilaku sesi itu tidak berubah ketika skill diperbarui.

Langkah 4 — jadwalkan peninjauan kuartalan

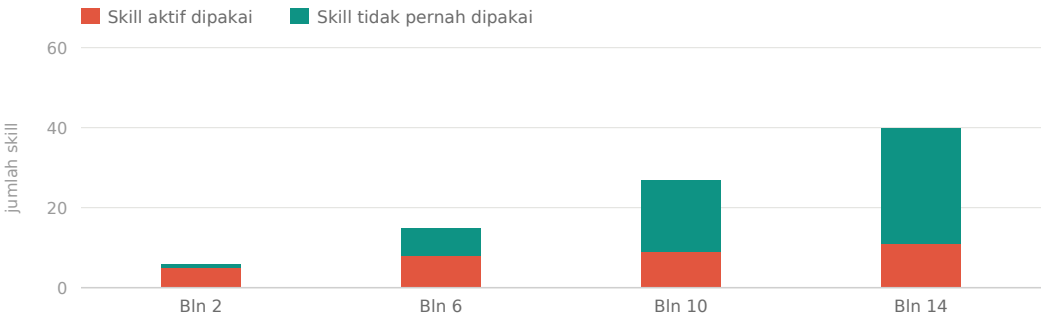
Tetapkan tanggal dan pelaksananya sekarang. Peninjauan yang tidak dijadwalkan tidak pernah terjadi, dan skill yang usang tidak akan menghilang sendiri — ia hanya terus menempati konteks dan menambah biaya pada setiap giliran.

BAB 35 / STUDI KASUS

Empat puluh skill dan tidak ada yang tahu asalnya

Sebuah tim memakai OpenClaw selama empat belas bulan. Sepanjang itu, setiap kali agen mempelajari cara mengerjakan sesuatu, mereka menyimpannya sebagai skill. Praktik yang terasa produktif dan memang produktif — pada awalnya.

Pada bulan keempat belas, indeks skill berisi empat puluh entri. Ketika biaya token diperiksa, sebagian besar kenaikannya berasal dari indeks yang membengkak: nama dan deskripsi keempat puluh skill itu masuk konteks pada setiap giliran, dipakai atau tidak.



Gambar 35.3 — Pertumbuhan sintetis indeks skill. Yang tumbuh adalah yang tidak dipakai, dan biayanya tetap dibayar setiap giliran.

Peninjauan pertama mereka menghapus dua puluh sembilan skill dalam satu sore. Tidak ada yang protes, karena tidak ada yang memakainya. Biaya token turun, dan kualitas pemilihan skill oleh agen justru membaik karena indeksnya menjadi lebih jelas.

MENGAPA MENGHAPUS ADALAH PEKERJAAN PRODUKTIF
Indeks skill yang sehat berisi puluhan entri, bukan ratusan. Setiap skill yang terlihat menambah biaya pada setiap percakapan dan menambah kemungkinan agen memilih yang keliru.

BAB 35 / LATIHAN

Latihan mandiri

1. Daftarkan seluruh skill di mesin Anda dan beri tanda asal masing-masing memakai kategori pada praktikum langkah dua.
2. Tandai skill yang tidak pernah dipakai dalam tiga bulan terakhir dan putuskan nasibnya.
3. Lampirkan satu skill ke sesi tertentu dengan revisi eksplisit, lalu buktikan sesi lain tidak melihatnya.
4. Tetapkan tanggal peninjauan kuartalan berikutnya beserta nama pelaksananya.

BAB 35 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Perilaku sesi berubah tanpa ada perubahan konfigurasi	Skill yang dilampirkan diperbarui. Sebut revisi secara eksplisit.
Indeks skill membengkak	Skill buatan agen menumpuk. Tinjau berkala dan hapus yang usang.
Skill workshop berperilaku tidak terduga	Ia tidak melalui pemindaian registri. Tinjau isinya seperti kode.
Penyegaran menyentuh skill yang tidak dimaksud	--skill-id dihilangkan, sehingga seluruh skill terpilih ikut disegarkan.
Tidak ada yang tahu asal sebuah skill	Tidak ada pemilik bernama. Tetapkan sebelum jumlahnya bertambah.

Tiga pertanyaan review

1. Mengapa skill buatan agen tidak otomatis lebih aman daripada skill dari registri?
2. Kapan melampirkan skill ke satu sesi lebih tepat daripada menyalakannya untuk seluruh agen?
3. Rancang siklus peninjauan skill untuk tim yang memakai OpenClaw selama satu tahun.

Kunci pembahasan

Skill buatan agen tidak melalui pemindaian registri maupun tinjauan pihak lain, sehingga satu-satunya pemeriksaan adalah yang Anda lakukan sendiri — dan biasanya tidak ada,

justru karena terasa buatan sendiri. Melampirkan ke satu sesi tepat ketika pengetahuannya khusus untuk satu pekerjaan dan kehadirannya di percakapan lain hanya menambah biaya token serta risiko perilaku yang tidak diinginkan. Siklus peninjauan setahun yang masuk akal: setiap kuartal periksa daftar skill aktif, hapus yang tidak dipakai tiga bulan terakhir, verifikasi pemilik masih ada di tim, dan periksa ulang skill yang menyentuh kanal pelanggan.

Rujukan: dokumentasi OpenClaw, halaman cli/skills bagian library dan workshop, serta catatan rilis v2026.9.3 dan v2026.9.4, ditinjau 17 September 2026.

SKILLS DAN CLAWHUB

Bab 36 Visibilitas skill per agen

Skill yang terpasang belum tentu terlihat. Bab pendek ini menutup Bagian V dengan mekanisme yang menentukan skill mana yang hadir pada agen mana — lapis terakhir dari pemisahan kewenangan yang dimulai di Bab 9.

Tujuan belajar

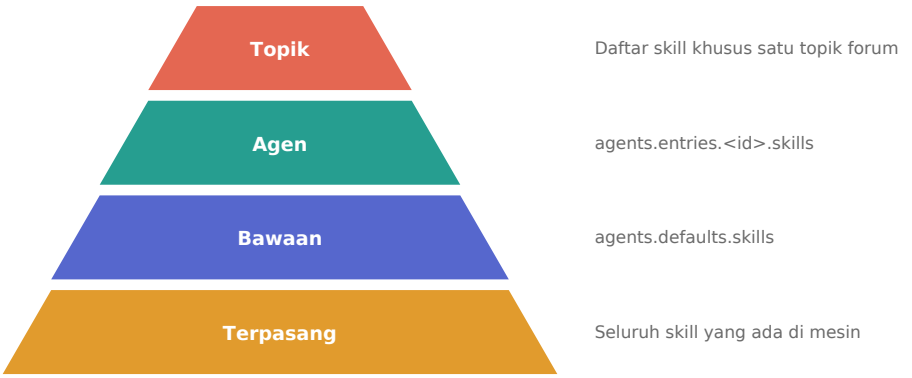
Mengatur daftar skill per agen dan per topik, memahami interaksinya dengan kebijakan tool, dan mengendalikan ukuran indeks skill. Pada akhir bab, pembaca dapat memberi tiap agen persis pengetahuan yang ia butuhkan.

Tiga tingkat pengaturan

```
{
  agents: {
    defaults: { skills: ["search", "docs"] },
    entries: {
      cs:  { skills: ["cs-playbook"] },
      ops: { skills: ["runbook", "shell-notes"] },
    },
  },
}
```

agents.defaults.skills berlaku umum; entri per agen menimpanya.

Telegram menambahkan tingkat ketiga: konfigurasi topik dapat menyebut daftar skill sendiri, dan ia mewarisi setelan grup kecuali ditimpa. Gabungan ketiganya memungkinkan satu grup operasi memiliki topik yang benar-benar berbeda pengetahuannya, seperti dibahas di Bab 29.



Gambar 36.1 — Empat tingkat penyempitan. Dari bawah ke atas, setiap tingkat memilih sebagian dari tingkat di bawahnya.

BAB 36 / BIAYA

Indeks yang sehat

Nama dan deskripsi setiap skill yang terlihat berada di konteks pada setiap giliran. Indeks yang membengkak menaikkan biaya token pada setiap percakapan, bahkan ketika tidak ada satu pun skill yang dipakai.

UKURAN INDEKS ADALAH BIAYA BERJALAN
Indeks yang sehat berisi puluhan entri, bukan ratusan. Bila agen Anda memiliki ratusan skill yang terlihat, sebagian besar tidak pernah dipakai dan Anda membayar untuk kehadiran mereka pada setiap pesan.

Skill bukan pengganti kebijakan tool

Menyembunyikan skill dari sebuah agen tidak mencabut kemampuannya. Agen yang tidak melihat skill runbook tetap memiliki tool shell bila kebijakan tool mengizinkannya; ia hanya tidak tahu prosedur perusahaan. Untuk mencabut kemampuan, Anda perlu kebijakan tool yang dibahas di Bab 43.

VISIBILITAS SKILL		KEBIJAKAN TOOL
Pengetahuan	**Mengatur**	Kemampuan
Agen tidak tahu caranya	**Bila dicabut**	Agen tidak bisa
Ya, bila agen tahu sendiri	**Dapat dilewati**	Tidak
Fokus dan biaya	**Dipakai untuk**	Keamanan

Gambar 36.2 — Dua mekanisme yang sering tertukar. Hanya kolom kanan yang merupakan kontrol keamanan.

BAB 36 / PRAKTIKUM

Memberi tiap agen persis yang ia butuhkan

Praktikum ini mempersempit indeks skill per agen, lalu mengukur dampaknya terhadap biaya. Mengukur sebelum dan sesudah mengubah pekerjaan ini dari dugaan menjadi bukti.

Langkah 1 — ukur biaya sebelum mengubah apa pun

```
openclaw status
openclaw gateway usage-cost
```

Catat angkanya. Tanpa garis dasar, Anda tidak dapat membuktikan perbaikan.

Langkah 2 — daftar skill yang benar-benar dipakai tiap agen

Untuk tiap agen, tuliskan pekerjaan yang benar-benar ia kerjakan, lalu skill mana yang dibutuhkan pekerjaan itu. Daftar yang disusun dari pekerjaan selalu lebih pendek daripada daftar yang disusun dari apa yang tersedia.

Langkah 3 — persempit dan uji

```
{
  agents: {
    defaults: { skills: [] },
    entries: {
      cs:  { skills: ["cs-playbook", "katalog"] },
      ops: { skills: ["runbook", "status"] },
    },
  },
}
```

Mulai dari daftar bawaan yang kosong, lalu berikan per agen seperlunya.

Langkah 4 — buktikan pembedaannya

Uji	Cara	Hasil yang benar
Agen melihat skill sendiri	Minta ia menyebutkan yang tersedia	Hanya yang didaftarkan
Agen tidak melihat milik lain	Minta ia memakai skill agen lain	Tidak dikenali
Kemampuan tetap ada	Minta tindakan yang tool-nya masih terbuka	Tetap dapat dilakukan
Biaya turun	Bandingkan dengan garis dasar	Token masuk berkurang

Uji ketiga adalah yang paling penting untuk dipahami. Menyembunyikan skill tidak mencabut kemampuan; ia hanya menghapus pengetahuan tentang prosedurnya. Untuk mencabut kemampuan, Anda memerlukan kebijakan tool di Bab 43.

BAB 36 / STUDI KASUS

Agen yang tetap bisa meski tidak tahu caranya

Sebuah tim ingin memastikan agen layanan pelanggan tidak dapat menyentuh sistem produksi. Mereka menghapus seluruh skill operasi dari daftar agen itu, mengujinya dengan bertanya apakah ia tahu cara merestart layanan, dan mendapat jawaban tidak.

Merasa aman, mereka melanjutkan. Beberapa minggu kemudian, seorang pelanggan yang kebetulan berlatar teknis menuliskan perintah lengkap di dalam pesannya. Agen menjalankannya, karena tool eksekusi tidak pernah ditolak untuk agen itu.

YANG DIUJI		YANG SEHARUSNYA DIUJI
Apakah kamu tahu caranya	**Pertanyaan**	Apakah kamu bisa melakukannya
Tidak	**Jawaban**	Bisa
Visibilitas skill	**Mekanisme**	Kebijakan tool
Pengetahuan	**Sifatnya**	Kemampuan
Ya, bila diberi tahu	**Dapat dilewati**	Tidak

Gambar 36.3 — Dua pertanyaan yang terdengar mirip dan menguji hal yang sama sekali berbeda.

Pengujian mereka mengukur hal yang salah. Agen memang tidak tahu prosedurnya, tetapi pengetahuan dapat diberikan oleh siapa pun yang mengirim pesan — termasuk orang di luar organisasi.

CARA MENGUJI BATAS YANG BENAR

Untuk menguji batas keamanan, jangan bertanya kepada agen apakah ia tahu. Perintahkan ia melakukannya dan lihat apakah ia bisa. Hanya penolakan pada tingkat kebijakan tool yang merupakan bukti.

BAB 36 / LATIHAN

Latihan mandiri

1. Ukur biaya garis dasar Anda, persempit indeks skill per agen, lalu ukur lagi setelah satu minggu.
2. Jalankan keempat uji pada praktikum langkah empat untuk setiap agen Anda.
3. Untuk satu agen, buktikan bahwa menyembunyikan skill tidak mencabut kemampuan, lalu tutup celahnya dengan kebijakan tool.
4. Tuliskan daftar skill final tiap agen beserta satu kalimat alasan untuk setiap entri.

BAB 36 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Skill terpasang tidak muncul di agen tertentu	Daftar skill per agen menyempitkannya. Itu perilaku yang benar.
Biaya token tinggi tanpa penambahan pengguna	Indeks skill membengkak. Persempit daftar per agen.
Agen tetap bisa menjalankan perintah berbahaya	Menyembunyikan skill tidak mencabut tool. Pakai kebijakan tool.
Setelan topik tidak mengubah skill	Topik mewarisi grup; timpa secara eksplisit di konfigurasi topik.
Agen memakai skill milik agen lain	Daftar skill tidak disempitkan. Setel agents.entries per agen.

Tiga pertanyaan review

1. Mengapa visibilitas skill bukan kontrol keamanan?
2. Bagaimana ukuran indeks skill memengaruhi biaya, dan mengapa dampaknya terasa bahkan saat skill tidak dipakai?
3. Rancang daftar skill untuk agen CS dan agen operasi, lalu jelaskan satu skill yang sengaja tidak Anda berikan ke keduanya.

Kunci pembahasan

Visibilitas skill hanya mengatur pengetahuan, dan model yang sudah mengetahui cara melakukan sesuatu tetap dapat melakukannya selama tool-nya tersedia — sehingga ia tidak pernah menjadi batas yang dapat ditegakkan. Ukuran indeks berpengaruh pada biaya karena nama dan deskripsi seluruh skill yang terlihat ikut masuk konteks pada setiap giliran, dipakai atau tidak. Untuk agen CS, berikan skill playbook layanan dan pencarian katalog; untuk agen operasi, berikan runbook dan catatan shell; dan skill yang sengaja tidak diberikan ke keduanya adalah skill yang dapat mengubah konfigurasi Gateway, karena perubahan itu harus lewat manusia.

Rujukan: dokumentasi OpenClaw, halaman tools/skills-config, gateway/config-agents, dan channels/telegram/threads-and-sessions, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

Bab 37 Model kapabilitas dan pemasangan plugin

Bila skill menambah pengetahuan, plugin menambah kemampuan. Ia dapat mendaftarkan tool, hook, perintah, kanal, penyedia model, server MCP, dan rute HTTP. Karena itulah pemasangan plugin melewati layar persetujuan yang tidak dimiliki skill.

Tujuan belajar

Membaca layar persetujuan kapabilitas, memilih sumber pemasangan secara deterministik, dan memeriksa plugin sebelum menyalakannya. Pada akhir bab, pembaca dapat memasang plugin pihak ketiga dengan risiko yang dipahami.

Persetujuan kapabilitas

OpenClaw meminta Anda meninjau kapabilitas yang dideklarasikan sebuah plugin pihak ketiga sebelum memasang atau menyalakannya. Layar persetujuan mengidentifikasi plugin, versi dan sumbernya, integritas artefak, serta informasi kepercayaan yang tersedia.

Gambar 37.1 — Apa yang tercantum di layar persetujuan. Layar ini adalah satu-satunya tempat Anda melihat seluruhnya sekaligus sebelum memutuskan.

Layar itu juga mencantumkan skill yang dibawa plugin serta pemberian operator yang berlaku untuk hook, akses model, dan subagen. Plugin bawaan dan plugin pihak pertama yang terverifikasi dari katalog resmi OpenClaw tidak menuntut tinjauan kapabilitas ini saat pemasangan, penyalaan, pembaruan, atau perbaikan Doctor.

BAB 37 / SUMBER

Memilih sumber secara deterministik

Spesifikasi paket telanjang memiliki perilaku kompatibilitas khusus, dan memahaminya mencegah Anda memasang paket yang berbeda dari yang Anda kira.

Bentuk spesifikasi	Yang terjadi
Nama telanjang yang cocok id plugin bawaan	Memakai sumber bawaan
Nama telanjang yang cocok plugin eksternal resmi	Memakai katalog paket resmi
Nama telanjang lainnya	Dipasang lewat npm
Spesifikasi @openclaw/* mentah	Menuju salinan bawaan sebelum fallback npm

```
openclaw plugins install clawhub:<package>
openclaw plugins install npm:@openclaw/<plugin>@<version>
openclaw plugins install git:<repo>
openclaw plugins install ./my-plugin
openclaw plugins install -l ./my-plugin    # tautan lokal, tanpa menyalin
```

Pakai awalan clawhub:, npm:, git:, atau npm-pack: untuk pemilihan sumber yang deterministik.

TAUTAN LOKAL UNTUK PENGEMBANGAN

Opsi --link tidak didukung bersama --marketplace atau pemasangan git:, dan ia menuntut jalur lokal yang sudah ada. Untuk tautan lokal noninteraktif, lewatkan --force setelah meninjau sumbernya; ia mengonfirmasi asal tetapi tidak menyalin maupun menimpa direktori yang ditautkan.

Pemasangan plugin memvalidasi kompatibilitas pluginApi dan minGatewayVersion yang diiklankan sebelum pemasangan arsip berjalan. Ketika sebuah versi paket menerbitkan artefak ClawPack, OpenClaw lebih memilih berkas npm-pack yang diunggah persis, memverifikasi header digest ClawHub beserta byte yang diunduh, dan mencatat metadata artefak untuk pembaruan berikutnya.

BAB 37 / MEMERIKSA

Memeriksa sebelum dan sesudah

```
openclaw plugins list
openclaw plugins list --enabled
openclaw plugins list --verbose

openclaw plugins inspect <id>
openclaw plugins inspect <id> --runtime
openclaw plugins inspect <id> --json
openclaw plugins inspect --all

openclaw plugins doctor
```

inspect menampilkan identitas, status muat, sumber, kapabilitas manifest, flag kebijakan, diagnostik, dan dukungan MCP atau LSP yang terdeteksi.

> **TERDAFTAR BUKAN BERARTI TERMUAT**
> plugins list adalah pemeriksaan inventaris dingin: apa yang dapat ditemukan OpenClaw dari konfigurasi, manifest, dan registri plugin yang tersimpan. Ia tidak membuktikan bahwa Gateway yang sedang berjalan sudah mengimpor runtime plugin itu.

Tambahkan --runtime untuk memuat modul plugin dan menyertakan hook, tool, perintah, layanan, metode Gateway, dan rute HTTP yang terdaftar. Namun inspect --runtime pun memuat plugin di dalam proses CLI yang memeriksa, bukan di Gateway. Untuk benar-benar yakin, picu kapabilitasnya dan verifikasi efek nyatanya.

Keluaran JSON menyertakan kontrak manifest plugin sehingga operator dapat mengaudit deklarasi permukaan tepercaya sebelum menyalakan atau merestart sebuah plugin. Bagi perusahaan, inilah bahan yang layak disimpan sebagai bukti audit.

BAB 37 / PRAKTIKUM

Membaca layar persetujuan seperti auditor

Layar persetujuan kapabilitas adalah satu-satunya tempat Anda melihat seluruh yang diminta plugin sekaligus. Praktikum ini melatih Anda membacanya sebagai daftar kewenangan, bukan sebagai dialog yang harus dilewati.

Langkah 1 — periksa sebelum memasang, bukan sesudah

```
openclaw plugins search "<kata kunci>"
openclaw plugins inspect <id> --json | head -60
```

Inspect bekerja tanpa mengimpor runtime plugin secara bawaan.

Langkah 2 — terjemahkan tiap kapabilitas menjadi pertanyaan

Kapabilitas yang tercantum	Pertanyaan yang harus Anda jawab
Kanal	Apakah saya ingin plugin ini menerima dan mengirim pesan?
Tool	Apa yang dapat dilakukan tool itu pada mesin saya?
Hook	Titik mana dalam alur yang akan ia cegat?
Server MCP	Proses apa yang akan dijalankan, dan dari mana?
Perintah CLI	Perintah baru apa yang muncul di permukaan openclaw?
Tanda berbahaya	Mengapa plugin ini membutuhkannya?

Bila Anda tidak dapat menjawab salah satu pertanyaan itu, jawabannya bukan menekan setuju sambil berharap. Jawabannya adalah membaca dokumentasi plugin itu, atau tidak memasangnya.

Langkah 3 — pilih sumber secara deterministik

```
openclaw plugins install clawhub:<package>
openclaw plugins install npm:@openclaw/<plugin>@<version>
openclaw plugins install git:<repo>
```

Nama telanjang diselesaikan berjenjang dan dapat memasang sumber yang berbeda dari dugaan Anda.

Langkah 4 — buktikan yang berjalan sama dengan yang terdaftar

```
openclaw plugins list --enabled
openclaw plugins inspect <id> --runtime
# lalu picu kapabilitasnya dan amati efek nyatanya
```

list adalah inventaris dingin; hanya memicu kapabilitas yang membuktikan.

BAB 37 / STUDI KASUS

Nama telanjang yang memasang paket berbeda

Seorang insinyur memasang plugin dengan menuliskan namanya saja, seperti yang ia lihat di sebuah contoh. Pemasangan berhasil, plugin muncul di daftar, dan fungsinya berjalan.

Beberapa minggu kemudian ia mencoba memakai fitur yang disebutkan dokumentasi resmi dan tidak menemukannya. Setelah setengah hari, ia menyadari bahwa yang terpasang adalah paket npm dengan nama serupa, bukan plugin bawaan yang ia maksud.

YANG IA KETIK		YANG TERPASANG
Nama telanjang	**Spesifikasi**	Diselesaikan berjenjang
Plugin bawaan	**Dugaan**	Paket npm pihak ketiga
Berjalan	**Fungsi dasar**	Berjalan
Ada di dokumentasi	**Fitur lanjutan**	Tidak ada di paket ini
Diasumsikan	**Sumber terverifikasi**	Tidak pernah diperiksa

Gambar 37.2 — Pemasangan yang berhasil dengan sumber yang keliru. Tidak ada galat karena tidak ada yang salah secara teknis.

Nama telanjang yang cocok dengan id plugin bawaan memakai sumber bawaan; yang cocok dengan plugin eksternal resmi memakai katalog resmi; sisanya dipasang lewat npm. Tanpa awalan eksplisit, Anda tidak mengendalikan mana yang menang.

> **KEBIASAAN YANG MURAH DAN MENYELAMATKAN**
> Untuk apa pun yang akan berjalan di mesin produksi, pakai awalan sumber yang eksplisit. Mengetik enam karakter tambahan menghilangkan seluruh kelas ketidakpastian ini.

BAB 37 / LATIHAN

Latihan mandiri

1. Jalankan inspect --json pada setiap plugin yang aktif di mesin Anda dan simpan keluarannya sebagai bukti audit.
2. Terjemahkan kapabilitas satu plugin menjadi enam pertanyaan pada praktikum langkah dua, lalu jawab semuanya.
3. Periksa apakah ada plugin di mesin Anda yang dipasang dengan nama telanjang, dan pastikan sumbernya memang yang Anda maksud.
4. Susun kebijakan pemasangan plugin untuk tim Anda, termasuk siapa yang boleh menyetujui kapabilitas berbahaya.

BAB 37 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Plugin terdaftar tetapi tidak berfungsi	list hanya inventaris dingin. Picu kapabilitasnya dan verifikasi efeknya.
Versi yang terpasang bukan yang diminta	Nama telanjang mungkin menuju salinan bawaan. Pakai awalan npm: eksplisit.
Pemasangan ditolak sebelum arsip dibuka	Validasi pluginApi atau minGatewayVersion gagal.
Tidak ada layar persetujuan	Plugin bawaan atau pihak pertama terverifikasi memang melewatinya.
--link ditolak	Tidak didukung bersama --marketplace atau git:, dan jalurnya harus sudah ada.

Tiga pertanyaan review

1. Mengapa plugin menuntut persetujuan kapabilitas sementara skill tidak?
2. Jelaskan mengapa nama paket telanjang dapat memasang sesuatu yang berbeda dari yang Anda duga.
3. Rancang prosedur audit plugin untuk perusahaan yang memasang plugin pihak ketiga.

Kunci pembahasan

Plugin mendaftarkan kode yang benar-benar berjalan — tool, hook, rute HTTP, proses MCP — sementara skill hanya menambahkan teks yang dibaca model, sehingga kelas risikonya berbeda secara mendasar. Nama telanjang diselesaikan berjenjang: ia dapat menuju plugin bawaan, katalog resmi, atau npm, sehingga tanpa awalan eksplisit Anda tidak mengendalikan sumber mana yang menang. Prosedur audit yang layak: jalankan inspect

--json sebelum menyalakan, simpan keluarannya sebagai bukti, tinjau kontrak permukaan tepercaya, picu setiap kapabilitas di lingkungan uji, dan baru kemudian nyalakan di produksi dengan versi yang dipin.

Rujukan: dokumentasi OpenClaw, halaman plugins/manage-plugins, cli/plugins/install, dan cli/plugins/inspect-and-diagnose, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

Bab 38 Manifest plugin, skema, dan server MCP bawaan

Manifest adalah kontrak antara plugin dan Gateway. Ia dideklarasikan sebelum kode apa pun berjalan, divalidasi dengan ketat, dan menjadi dasar layar persetujuan yang dibahas di bab sebelumnya.

Tujuan belajar

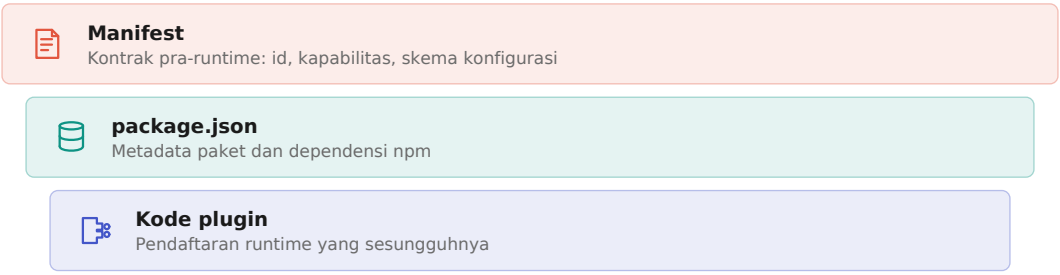
Membaca medan manifest yang penting, memahami pemisahan manifest dari package.json, dan mengendalikan server MCP yang dibawa plugin. Pada akhir bab, pembaca dapat mengaudit sebuah manifest tanpa membaca kodenya.

Bentuk minimal

```
{
  "id": "voice-call",
  "configSchema": {
    "type": "object",
    "additionalProperties": false,
    "properties": {}
  }
}
```

additionalProperties bernilai false: validasi konfigurasi plugin bersifat ketat, sama seperti konfigurasi Gateway.

Manifest tidak dipakai untuk mendaftarkan hook runtime native, mendeklarasikan titik masuk runtime plugin secara penuh, atau menampung metadata npm install. Ketiganya termasuk dalam kode plugin dan package.json.



Gambar 38.1 — Tiga berkas dengan peran berbeda. Audit dimulai dari lapis paling atas karena ia dapat dibaca tanpa menjalankan apa pun.

BAB 38 / PERMUKAAN

Medan permukaan host

Manifest mendeklarasikan permukaan yang disentuh plugin pada host: ikon, perintah CLI, MCP, Control UI, dasbor, QA, kanal, dan cadangan. Masing-masing memberi tahu operator di mana plugin ini akan muncul.

Kelompok medan	Isinya
Kapabilitas	Kepemilikan kapabilitas, metadata ketersediaan tool, rencana aktivasi
Konfigurasi dan rahasia	Tanda berbahaya, migrasi SecretRef, preset penyedia rahasia
Model	Katalog model, keluarga singkat, normalisasi id, harga
Penyedia	Metadata generasi, pemahaman media, endpoint, permintaan
Penyiapan dan auth	Deskriptor penyiapan, pilihan auth, penemuan percakapan
Permukaan host	Ikon, CLI, MCP, Control UI, dasbor, QA, kanal, cadangan

Server MCP yang dibawa plugin

Medan mcpServers memungkinkan plugin native mengirimkan server MCP — termasuk sebuah MCP App — tanpa menuntut operator menduplikasi definisi proses statisnya di openclaw.json.

```
{
  "mcpServers": {
    "example": {
      "transport": "stdio",
      "command": "node",
      "args": ["./mcp-server.js"]
    }
  }
}
```

Jalur command, args, cwd, dan workingDirectory yang relatif diselesaikan dari akar plugin.

ANDA TETAP MEMEGANG KENDALI AKHIR

OpenClaw menyertakan server ini hanya selama plugin pemiliknya menyala. Konfigurasi pengguna tetap otoritatif: mcp.servers.<name> dapat menggantikan default plugin atau menyetel enabled bernilai false untuk menghilangkannya.

Perenderan MCP App dan panggilan tool server tetap menuntut setelan MCP Apps yang normal beserta kebijakan tool yang efektif. Mendeklarasikan sebuah server tidak melewati salah satu batas itu — deklarasi bukan izin.

BAB 38 / BUNDEL

Bundel dari ekosistem lain

OpenClaw dapat memasang paket dari format lain sebagai bundel yang kompatibel. Bundel dipasang ke akar plugin yang normal dan ikut dalam alur list, info, enable, dan disable yang sama.

Jalur skill dan hook-pack harus tetap berada di dalam akar plugin dan diperiksa batasnya. Berkas setelan dibaca dengan pemeriksaan batas yang sama. Server MCP stdio yang didukung dapat dijalankan sebagai subproses. Ini membuat bundel lebih aman secara bawaan, tetapi Anda tetap harus memperlakukan bundel pihak ketiga sebagai konten tepercaya untuk fitur yang memang ia paparkan.

> **TERCANTUM TIDAK SELALU BERARTI AKTIF**
> Bila sebuah kapabilitas tercantum tetapi ditandai belum tersambung, itu batas produk, bukan pemasangan yang rusak. Jalankan plugins inspect untuk melihat statusnya.

Bila sebuah paket membawa penanda khusus klien sekaligus plugin.json di akarnya, format khusus klien yang menang supaya pemetaannya yang lebih kaya tetap terjaga. Bila sebuah direktori memuat manifest native sekaligus penanda bundel, OpenClaw memakai jalur native. Ini mencegah paket berformat ganda terpasang setengah-setengah sebagai bundel.

BAB 38 / PRAKTIKUM

Mengaudit manifest tanpa menjalankan kodenya

Manifest dapat dibaca sepenuhnya tanpa mengeksekusi apa pun. Praktikum ini memanfaatkan sifat itu: seluruh audit dikerjakan sebelum satu baris kode plugin pernah berjalan di mesin Anda.

Langkah 1 — baca manifest sebagai dokumen

```
openclaw plugins inspect <id> --json > audit-<id>.json
jq '.manifest.capabilities' audit-<id>.json
jq '.manifest.mcpServers' audit-<id>.json
jq '.contracts' audit-<id>.json
```

Keluaran JSON memuat kontrak permukaan tepercaya yang dapat diaudit sebelum plugin dinyalakan.

Langkah 2 — periksa server MCP yang dibawanya

Medan	Yang Anda periksa
transport	stdio berarti subproses akan dijalankan di mesin Anda
command dan args	Program apa yang dijalankan, dengan argumen apa
cwd	Jalur relatif diselesaikan dari akar plugin
Nama server	Apakah ia bentrok dengan mcp.servers milik Anda

Langkah 3 — timpa atau matikan bila perlu

```
{
  mcp: { servers: {
    "<nama-server-plugin>": { enabled: false },
  } },
}
```

Konfigurasi pengguna tetap otoritatif; ia dapat mengganti default plugin atau menghilangkannya.

Langkah 4 — pahami batas deklarasi

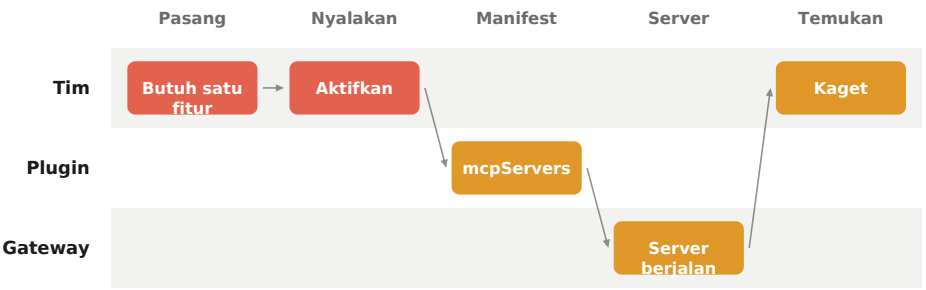
Mendeklarasikan sebuah server tidak melewati setelan MCP Apps maupun kebijakan tool yang efektif. Deklarasi mendaftarkan; ia tidak memberi izin. Menguji sendiri sekali akan membuat perbedaan ini melekat.

BAB 38 / STUDI KASUS

Server MCP yang muncul tanpa diminta

Sebuah tim memasang plugin untuk satu fitur kecil yang mereka butuhkan. Beberapa hari kemudian, seorang anggota tim memeriksa daftar server MCP dan menemukan entri yang tidak pernah ia tambahkan.

Server itu dibawa oleh plugin lewat medan manifest, dan OpenClaw menyertakannya selama plugin pemiliknya menyala. Perilaku ini terdokumentasi dan disengaja, tetapi tim itu tidak mengetahuinya karena tidak ada yang membaca manifest saat memasang.



Gambar 38.2 — Rantai yang seluruhnya terdokumentasi, dan tetap mengejutkan karena tidak ada yang membaca manifest.

Setelah diperiksa, server itu sah dan berguna. Yang bermasalah adalah prosesnya: sebuah proses tambahan berjalan di mesin mereka selama beberapa hari tanpa ada yang memutuskan bahwa ia boleh berjalan.

> **MENGAPA AUDIT MANIFEST LAYAK DIBIASAKAN**
> Manifest adalah tempat Anda melihat apa yang akan terjadi sebelum ia terjadi. Membacanya memakan dua menit; menemukan kejutan berhari-hari kemudian memakan jauh lebih banyak.

BAB 38 / LATIHAN

Latihan mandiri

1. Audit manifest setiap plugin aktif Anda dan daftarkan seluruh server MCP yang mereka bawa.
2. Periksa medan command dan args tiap server tersebut, lalu pastikan seluruhnya diselesaikan dari akar plugin.
3. Matikan satu server yang dibawa plugin lewat konfigurasi, lalu buktikan ia benar-benar berhenti.
4. Tuliskan prosedur audit manifest tim Anda dalam lima langkah yang dapat dijalankan tanpa menjalankan kode plugin.

BAB 38 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Konfigurasi plugin ditolak	Skema plugin ketat; additionalProperties bernilai false.
Server MCP muncul tanpa Anda menambahkannya	Dibawa plugin lewat mcpServers. Timpa atau matikan lewat mcp.servers.
Kapabilitas bundel tidak berjalan	Ditandai belum tersambung; itu batas produk, bukan kerusakan.
Paket berformat ganda terpasang aneh	Jalur native menang atas penanda bundel. Perilaku yang disengaja.
Ingin mengaudit tanpa menjalankan kode	Baca manifest lewat plugins inspect --json.

Tiga pertanyaan review

1. Mengapa skema konfigurasi plugin dibuat ketat dengan additionalProperties false?
2. Apa arti kalimat 'mendeklarasikan sebuah server tidak melewati batas kebijakan tool'?
3. Anda diminta menyetujui plugin pihak ketiga yang membawa dua server MCP. Langkah apa saja yang Anda ambil sebelum menyetujuinya?

Kunci pembahasan

Skema ketat membuat kesalahan konfigurasi gagal saat pemasangan, bukan diam-diam terabaikan pada saat runtime; kunci yang salah ketik akan ditolak alih-alih membuat setelan keamanan yang Anda kira aktif ternyata tidak pernah terbaca. Kalimat itu berarti plugin dapat menyatakan ia membawa server MCP, tetapi tool dari server itu tetap harus lolos kebijakan tool yang berlaku — deklarasi hanya mendaftarkan, tidak memberi izin. Sebelum menyetujui plugin dengan dua server MCP, baca manifest lewat inspect --json,

periksa command dan args kedua server itu, pastikan keduanya berasal dari akar plugin, uji di lingkungan terpisah, lalu putuskan apakah salah satunya perlu dimatikan lewat mcp.servers.

Rujukan: dokumentasi OpenClaw, halaman plugins/manifest, plugins/manifest/surfaces, dan plugins/bundles, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

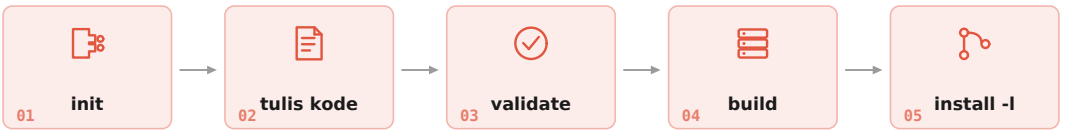
Bab 39 Membangun plugin pertama

Menulis plugin sendiri adalah jalan keluar ketika tidak ada skill maupun server MCP yang cocok. Bab ini membangun satu plugin dari perancah sampai siap dipasang, dengan penekanan pada bagian yang membuatnya layak produksi.

Tujuan belajar

Membuat perancah plugin, memvalidasinya, dan mengemasnya. Pada akhir bab, pembaca memiliki plugin yang lolos validasi dan dapat dipasang di mesin lain.

Siklus pengembangan



Gambar 39.1 — Siklus pengembangan. Tautan lokal di langkah terakhir membuat Anda dapat menyunting tanpa memasang ulang.

```
openclaw plugins init
openclaw plugins validate
openclaw plugins build

openclaw plugins install -l ./my-plugin
openclaw plugins inspect my-plugin --runtime
```

Opsi -l menambahkan direktori ke plugins.load.paths tanpa menyalinnya.

Perintah init membuat perancah dengan manifest, skema konfigurasi, dan titik masuk runtime. Perintah validate memeriksa manifest terhadap skema sebelum Anda membuang waktu memasangnya. Perintah build menyiapkan artefak yang dapat didistribusikan.

Memisahkan logika dari pendaftaran

Satu pola yang layak diikuti sejak awal: pisahkan logika bisnis dari lapisan yang mendaftarkannya ke OpenClaw. Logika murni dapat diuji tanpa menjalankan Gateway sama sekali, dan itu mengubah kecepatan pengembangan Anda secara drastis.

Gambar 39.2 — Empat berkas inti. Lapis ketiga tidak boleh mengimpor apa pun dari OpenClaw; itulah yang membuatnya mudah diuji.

> **MODUL NATIVE MENUNTUT DOKUMENTASI**
> Bila plugin Anda bergantung pada modul native, dokumentasikan langkah build beserta kebutuhan allowlist pengelola paket. Plugin yang gagal dibangun di mesin orang lain adalah plugin yang tidak dapat dipakai siapa pun selain Anda.

BAB 39 / DISTRIBUSI

Menyebarkan ke mesin lain

Jalur	Cocok untuk
Tautan lokal dengan -l	Pengembangan di mesin Anda sendiri
Direktori atau arsip	Distribusi internal tanpa registri
git:	Tim yang sudah memakai repositori bersama
npm: atau npm-pack:	Distribusi lewat registri paket perusahaan
clawhub:	Publikasi terbuka; tunduk pada aturan Bab 33 dan 34

Untuk plugin internal perusahaan, jalur git: atau registri npm privat hampir selalu lebih tepat daripada ClawHub — bukan karena alasan teknis, melainkan karena publikasi di ClawHub berlisensi MIT-0 dan tidak dapat ditarik.

BAB 39 / PRAKTIKUM

Plugin pertama yang dapat diuji

Praktikum ini membangun plugin kecil dengan struktur yang membuatnya dapat diuji tanpa menjalankan Gateway. Struktur itu adalah perbedaan antara siklus umpan balik detik dan menit.

Langkah 1 — buat perancah dan pahami isinya

```
openclaw plugins init
ls -la
cat openclaw.plugin.json
```

Baca manifest yang dihasilkan sebelum menambahkan apa pun.

Langkah 2 — pisahkan logika sejak baris pertama

Berkas	Boleh mengimpor OpenClaw?	Dapat diuji sendiri?
src/logic.ts	Tidak	Ya, dengan runner uji biasa
src/index.ts	Ya	Hanya lewat integrasi
openclaw.plugin.json	—	Lewat plugins validate
package.json	—	—

Aturan pada kolom kedua adalah yang menentukan. Berkas logika yang tidak mengimpor apa pun dari OpenClaw dapat dijalankan oleh runner uji mana pun dalam hitungan detik, tanpa Gateway, tanpa kanal, tanpa model.

Langkah 3 — validasi sebelum memasang

```
openclaw plugins validate
openclaw plugins build
openclaw plugins install -l ./my-plugin
```

Opsi -l menautkan direktori tanpa menyalinnya, sehingga suntingan Anda langsung terlihat.

Langkah 4 — buktikan perubahan Anda benar-benar berjalan

```
openclaw plugins inspect my-plugin --runtime
# lalu picu tool atau perintahnya dan amati efeknya
```

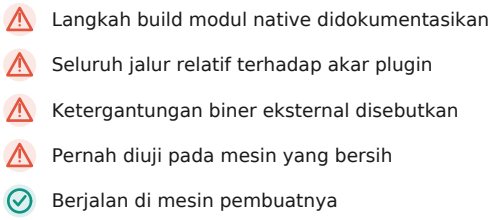
inspect --runtime memuat plugin di proses CLI, bukan di Gateway; memicu kapabilitas adalah bukti sesungguhnya.

BAB 39 / STUDI KASUS

Plugin yang hanya berjalan di mesin pembuatnya

Seorang insinyur membangun plugin internal yang bekerja sempurna di laptopnya. Ia membagikannya ke tiga rekan kerja. Tidak ada satu pun yang berhasil menjalankannya.

Penyebabnya bertumpuk: plugin bergantung pada modul native yang menuntut langkah build tambahan, memakai jalur absolut yang hanya ada di mesinnya, dan mengandaikan sebuah biner terpasang secara global. Tidak satu pun didokumentasikan, karena semuanya sudah ada di mesin pembuatnya sejak lama.



Gambar 39.3 — Empat butir gagal dan satu berhasil. Butir terakhir adalah satu-satunya yang diuji sebelum dibagikan.

Perbaikannya memakan tiga hari, sebagian besar untuk menemukan asumsi yang tidak pernah disadari pembuatnya. Menguji sekali pada mesin bersih sejak awal akan menemukan seluruhnya dalam satu jam.

> **UJI PADA MESIN YANG BUKAN MILIK ANDA**
> Sebelum membagikan plugin apa pun, jalankan pada mesin yang bersih atau pengguna macOS baru. Asumsi yang tidak terlihat hanya muncul di lingkungan yang tidak Anda bangun sendiri.

BAB 39 / LATIHAN

Latihan mandiri

1. Bangun satu plugin kecil dengan logika terpisah, lalu tulis satu uji yang berjalan tanpa menjalankan Gateway.
2. Pasang dengan tautan lokal, sunting satu baris, dan buktikan perubahan langsung terlihat.
3. Uji plugin Anda pada pengguna macOS baru atau mesin lain, dan catat setiap asumsi yang terungkap.
4. Tentukan jalur distribusi plugin internal Anda dan tuliskan alasan memilihnya.

BAB 39 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
validate gagal tanpa penjelasan jelas	Skema manifest ketat. Periksa kunci tak dikenal dan tipe yang salah.
Perubahan kode tidak terlihat	Plugin disalin, bukan ditautkan. Pasang ulang dengan -l.
Plugin gagal dibangun di mesin lain	Dependensi native. Dokumentasikan langkah build-nya.
inspect --runtime berbeda dari perilaku nyata	Ia memuat di proses CLI, bukan di Gateway. Picu kapabilitasnya.
Logika sulit diuji	Logika tercampur dengan pendaftaran. Pisahkan ke berkas sendiri.

Tiga pertanyaan review

1. Mengapa memisahkan logika bisnis dari lapisan pendaftaran mempercepat pengembangan?
2. Kapan plugin lebih tepat daripada server MCP untuk kebutuhan yang sama?
3. Rancang jalur distribusi plugin internal untuk perusahaan dengan lima mesin Gateway.

Kunci pembahasan

Logika murni dapat diuji dengan runner uji biasa dalam hitungan detik, sementara menguji lewat Gateway menuntut memuat plugin, menjalankan agen, dan memicu tool — perbedaan antara siklus umpan balik detik dan menit. Plugin lebih tepat daripada MCP ketika Anda butuh hook, perintah CLI, atau kanal baru, karena MCP hanya memaparkan tool. Untuk lima Gateway, jalur yang paling sederhana adalah repositori git internal yang dipin pada commit tertentu, sehingga pembaruan menjadi tindakan sadar di tiap mesin dan bukan sesuatu yang terjadi diam-diam.

Rujukan: dokumentasi OpenClaw, halaman plugins/building-plugins, cli/plugins/authoring, dan cli/plugins/install, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

Bab 40 Hook: mencegat agen, tool, pesan, dan siklus hidup

Hook adalah titik cegat. Ia memungkinkan kode Anda berjalan sebelum atau sesudah peristiwa tertentu — sebuah tool dipanggil, sebuah pesan dikirim, Gateway dimulai. Di sinilah kebijakan perusahaan berubah dari dokumen menjadi sesuatu yang ditegakkan.

Tujuan belajar

Membedakan hook internal dari hook plugin, memilih peristiwa yang tepat, dan memahami kapan sebuah hook boleh membatalkan. Pada akhir bab, pembaca dapat menegakkan aturan yang tidak dapat dilewati prompt.

Dua keluarga hook

HOOK INTERNAL		HOOK PLUGIN
Skrip di direktori hook	**Bentuk**	Kode dalam proses
Peristiwa siklus hidup	**Dipicu oleh**	Peristiwa runtime
/new, /reset, /stop, startup	**Contoh**	Pemanggilan tool, pengiriman pesan
openclaw hooks	**Dikelola dengan**	Manifest plugin
Otomasi operator	**Cocok untuk**	Penegakan kebijakan

Gambar 40.1 — Dua keluarga. Untuk pencegatan panggilan tool dalam proses, hook plugin adalah jawabannya.

Hook internal adalah skrip berbasis peristiwa yang dipicu peristiwa siklus hidup agen seperti /new, /reset, dan /stop, pemadatan sesi, startup Gateway, dan aliran pesan. Mereka ditemukan dari direktori hook dan dikelola dengan openclaw hooks.

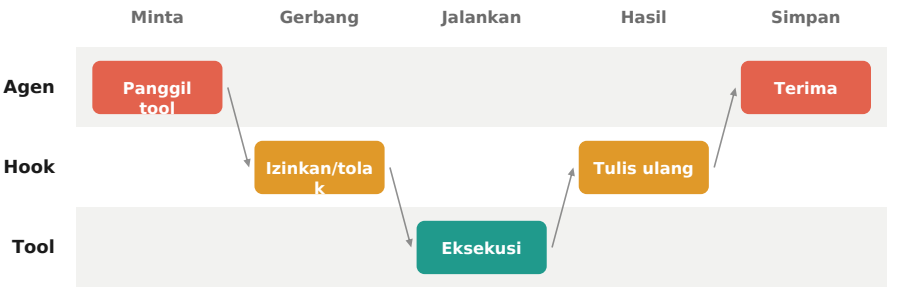
```
openclaw hooks list
openclaw hooks enable <name>
openclaw hooks disable <name>
```

Hook internal ditemukan dari direktori dan dikelola lewat CLI.

BAB 40 / KEBIJAKAN TOOL

Hook yang menggerbangi panggilan tool

Kelompok hook yang paling berharga bagi perusahaan adalah yang menggerbangi panggilan tool. Ia dapat menggerbangi, menulis ulang, dan menyetujui panggilan tool, serta menulis ulang hasil tool sebelum disimpan.



Gambar 40.2 — Hook duduk di antara agen dan tool, pada arah berangkat maupun pulang.

Kemampuan menulis ulang hasil sebelum disimpan layak diperhatikan. Ia memungkinkan penyuntingan data sensitif dari transkrip — misalnya menghapus nomor kartu yang terbaca dari sebuah dokumen — sehingga data itu tidak pernah masuk ke riwayat sesi.

Keluarga hook plugin

Keluarga	Yang dapat dilakukan
Prompt dan sesi	Menimpa model, membentuk prompt, menyimpan keadaan plugin
Kebijakan panggilan tool	Menggerbangi, menulis ulang, dan menyetujui panggilan
Pesan dan pengiriman	Mengklaim pesan masuk, menulis ulang atau membatalkan kiriman
Siklus hidup	Pemeriksaan pemasangan, start dan stop Gateway, peristiwa cron

BATAS YANG MENENTUKAN

Hook menuntut runtime tertanam OpenClaw. Ketika sebuah model dikonfigurasi dengan autentikasi OAuth Claude CLI, OpenClaw mendelegasikan loop model dan tool ke proses luar — dan hook dalam proses tidak berjalan di sana. Periksa ini sebelum mengandalkan hook sebagai kontrol keamanan.

BAB 40 / PRAKTIKUM

Menegakkan satu aturan yang tidak dapat dibujuk

Praktikum ini memindahkan satu aturan dari instruksi menjadi kode. Setelah mengerjakan ini sekali, perbedaan antara kebijakan yang tertulis dan kebijakan yang ditegakkan akan terasa jelas.

Langkah 1 — pilih aturan yang dapat diperiksa mesin

Aturan	Dapat ditegakkan	Mengapa
Jangan jalankan perintah destruktif	Ya	Pola perintah dapat diperiksa
Jangan kirim data ke luar organisasi	Ya	Tujuan dapat diperiksa
Selalu sopan kepada pelanggan	Tidak	Tidak ada kriteria mesin
Sunting nomor identitas dari hasil	Ya	Pola dapat dikenali

Baris ketiga penting untuk dipahami. Tidak semua aturan dapat ditegakkan, dan aturan semacam itu memang tempatnya di prompt — asalkan Anda tidak menganggapnya sebagai kontrol keamanan.

Langkah 2 — pasang hook dan buktikan ia berjalan

```
openclaw hooks list
openclaw hooks enable <name>
# lalu picu peristiwanya dan periksa jejaknya
```

Hook yang tidak pernah terpicu sama tidak bergunanya dengan hook yang tidak dipasang.

Langkah 3 — periksa apakah runtime Anda mendukungnya

Hook dalam proses menuntut runtime tertanam OpenClaw. Pada konfigurasi yang mendelegasikan loop model dan tool ke proses luar, hook itu tidak berjalan. Periksa ini sebelum mengandalkan hook sebagai kontrol keamanan — bukan sesudahnya.

Langkah 4 — uji hook Anda dengan upaya yang sengaja melanggar

```
# Kirim pesan yang secara eksplisit meminta agen
# melakukan hal yang dilarang hook Anda.
# Hook harus menolak, dan penolakannya harus terlihat di jejak.
```

Menguji dengan upaya melanggar adalah satu-satunya bukti bahwa gerbang bekerja.

BAB 40 / STUDI KASUS

Kontrol yang tidak berjalan pada jalur yang dipakai

Sebuah perusahaan membangun hook untuk menyunting nomor identitas dari hasil tool sebelum disimpan. Mereka mengujinya, ia bekerja, dan tim kepatuhan menerimanya sebagai kontrol.

Beberapa bulan kemudian, konfigurasi model diubah untuk alasan biaya. Konfigurasi baru mendelegasikan loop model dan tool ke proses luar. Hook dalam proses berhenti berjalan, dan tidak ada yang menyadarinya karena tidak ada galat — hook yang tidak berjalan tidak melaporkan apa pun.

SEBELUM PERUBAHAN		SESUDAH
Tertanam	**Runtime**	Didelegasikan ke luar
Ya	**Hook berjalan**	Tidak
Aktif	**Penyuntingan data**	Tidak aktif
Tidak ada	**Galat**	Tidak ada
Terpenuhi	**Status kepatuhan**	Terpenuhi di atas kertas

Gambar 40.3 — Perubahan konfigurasi yang tidak berhubungan mematikan kontrol kepatuhan tanpa satu pun sinyal.

Nomor identitas tersimpan di transkrip selama hampir lima bulan. Yang menemukannya bukan pemantauan, melainkan audit tahunan yang kebetulan memeriksa isi transkrip secara acak.

KONTROL HARUS MELAPORKAN KETIKA IA TIDAK BERJALAN
Ketika sebuah kontrol bergantung pada kondisi runtime, tuliskan ketergantungan itu sebagai bagian dari kontrolnya, dan periksa kondisinya secara berkala. Kontrol yang diam ketika mati adalah kontrol yang tidak dapat dipercaya.

BAB 40 / LATIHAN

Latihan mandiri

1. Pilih satu aturan organisasi Anda yang dapat diperiksa mesin, lalu pindahkan penegakannya dari prompt ke hook.
2. Uji hook itu dengan upaya yang sengaja melanggar, dan pastikan penolakannya terlihat di jejak.
3. Periksa apakah runtime Anda saat ini menjalankan hook dalam proses, dan tuliskan hasilnya.
4. Rancang pemeriksaan berkala yang akan memberi tahu Anda bila hook berhenti berjalan.

BAB 40 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Hook tidak pernah berjalan	Periksa penemuan dan kelayakannya lewat openclaw hooks.
Hook berjalan di satu konfigurasi, tidak di konfigurasi lain	Loop model mungkin didelegasikan ke proses luar. Hook menuntut runtime tertanam.
Data sensitif tetap masuk transkrip	Penulisan ulang hasil tool belum dipasang.
Hook memperlambat setiap giliran	Hook sinkron pada jalur panas. Pindahkan pekerjaan berat ke latar.
Kebijakan perusahaan dilanggar lewat prompt	Prompt bukan kontrol. Tegakkan lewat hook kebijakan tool.

Tiga pertanyaan review

1. Mengapa menegakkan kebijakan lewat hook lebih kuat daripada menuliskannya di AGENTS.md?
2. Berikan satu contoh penulisan ulang hasil tool yang bernilai bagi kepatuhan.
3. Apa yang harus Anda periksa sebelum mengandalkan hook sebagai kontrol keamanan utama?

Kunci pembahasan

Instruksi di AGENTS.md hanya memengaruhi kecenderungan model dan dapat kalah oleh prompt lain atau oleh isi dokumen yang tidak tepercaya, sedangkan hook berjalan sebagai kode yang tidak dapat dibujuk. Contoh penulisan ulang yang bernilai: menyunting nomor identitas atau nomor kartu dari hasil pembacaan dokumen sebelum ia tersimpan di transkrip, sehingga data itu tidak pernah ada di basis data sesi. Sebelum mengandalkan hook, pastikan runtime yang dipakai adalah runtime tertanam OpenClaw — pada konfigurasi yang mendelegasikan loop ke proses luar, hook dalam proses tidak berjalan dan kontrol Anda menjadi ilusi.

Rujukan: dokumentasi OpenClaw, halaman plugins/hooks, automation/hooks, dan plugins/hooks/tool-policy, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

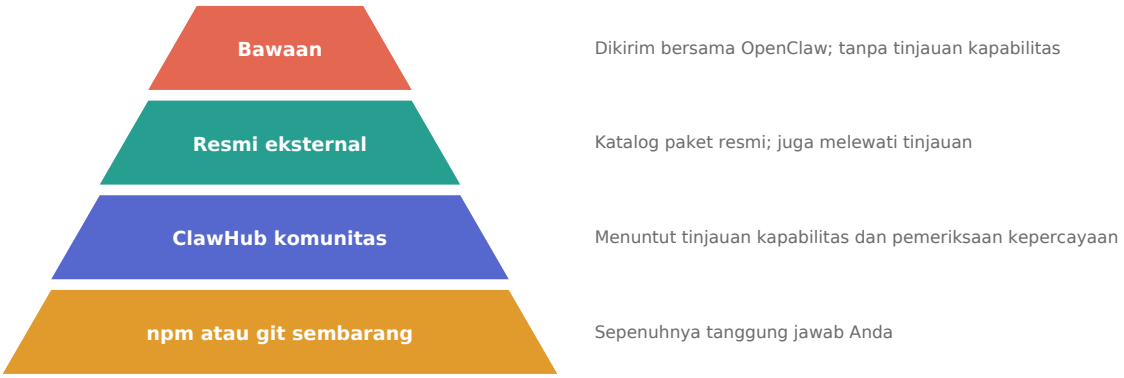
Bab 41 Plugin bawaan dan katalog resmi

Sebelum mencari di registri, periksa apa yang sudah ada. OpenClaw mengirimkan sejumlah plugin dan menawarkan katalog resmi yang tidak menuntut tinjauan kapabilitas saat dipasang — keuntungan yang nyata dari sisi keamanan.

Tujuan belajar

Menemukan plugin yang sudah tersedia, memahami mengapa sebagian melewati layar persetujuan, dan memilih yang layak dinyalakan. Pada akhir bab, pembaca dapat memperluas kemampuan tanpa menambah risiko rantai pasok.

Tiga kelas sumber

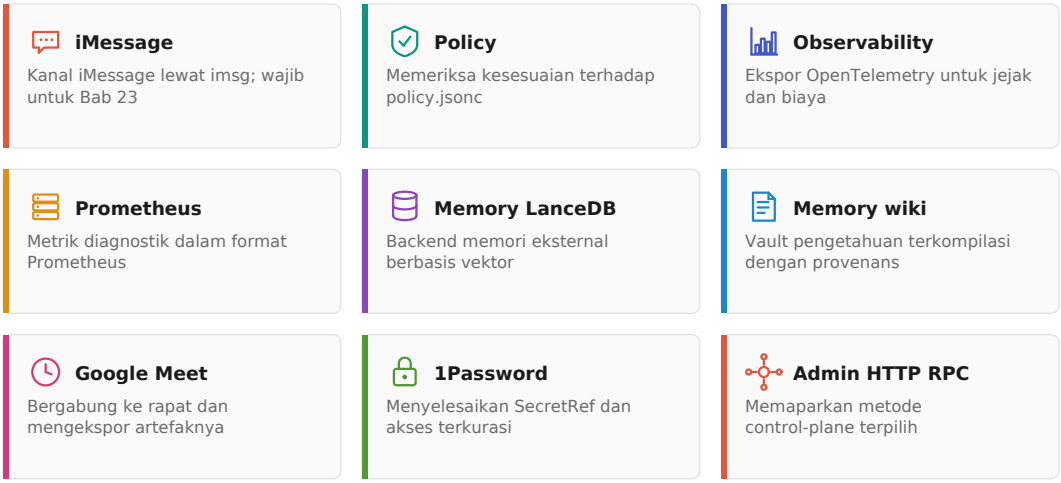


Gambar 41.1 — Empat kelas sumber dengan tingkat jaminan menurun. Naik satu tingkat hanya bila tingkat di atasnya tidak menyediakan yang Anda butuhkan.

Plugin bawaan dan plugin pihak pertama terverifikasi dari katalog resmi tidak menuntut tinjauan kapabilitas saat pemasangan, penyalaan, pembaruan, maupun perbaikan Doctor. Itu bukan kelonggaran; itu konsekuensi dari fakta bahwa keduanya sudah melalui proses rilis yang sama dengan produk intinya.

BAB 41 / KATALOG

Yang tersedia dan kapan dipakai



Gambar 41.2 — Sembilan plugin yang paling sering relevan untuk penerapan perusahaan. Semuanya opt-in.

Tiga yang layak dipertimbangkan lebih dulu

- Policy — Mengubah kebijakan tertulis menjadi pemeriksaan yang dapat dijalankan dan dibandingkan. Dibahas tuntas di Bab 54.
- Observability — Menjawab pertanyaan yang selalu muncul di bulan kedua: ke mana biaya token pergi, dan langkah mana yang paling lambat.
- 1Password — Menghapus alasan terakhir untuk menaruh kunci API sebagai teks biasa di berkas konfigurasi.

TIDAK ADA YANG GRATIS
Menyalakan plugin menambah permukaan. Plugin observability mengirim data ke luar, plugin admin HTTP memaparkan metode control-plane, dan plugin memori mengubah tempat ingatan Anda tersimpan. Masing-masing keputusan yang sah, dan masing-masing harus disengaja.

BAB 41 / PRAKTIKUM

Memilih plugin dengan urutan yang benar

Praktikum ini menghemat waktu sekaligus menurunkan risiko dengan satu kebiasaan: periksa katalog resmi sebelum mencari di registri terbuka.

Langkah 1 — lihat apa yang sudah tersedia

```
openclaw plugins list
openclaw plugins list --verbose
openclaw plugins search "<kebutuhan Anda>"
```

Sebagian kebutuhan sudah terjawab oleh plugin yang dikirim bersama produk.

Langkah 2 — pilih berdasarkan kelas sumber

Urutan pencarian	Tinjauan kapabilitas	Kapan naik ke tingkat berikut
Plugin bawaan	Tidak diperlukan	Bila kebutuhan tidak terjawab
Katalog resmi	Tidak diperlukan	Bila kebutuhan tidak terjawab
ClawHub komunitas	Diperlukan	Bila tidak ada pilihan resmi
npm atau git	Sepenuhnya tanggung jawab Anda	Jalur terakhir

Langkah 3 — nyalakan satu per satu, bukan sekaligus

Menyalakan tiga plugin dalam satu sesi membuat Anda tidak dapat mengaitkan perubahan perilaku atau biaya dengan plugin tertentu. Nyalakan satu, amati satu siklus kerja, baru lanjutkan.

Langkah 4 — catat apa yang berubah setelah tiap plugin

Yang dicatat	Mengapa
Biaya token harian	Sebagian plugin menambah konteks pada setiap giliran
Latensi balasan	Hook sinkron berada di jalur panas
Permukaan baru	Perintah CLI, rute HTTP, atau server MCP
Data yang keluar	Plugin observability memang mengirim ke luar

BAB 41 / STUDI KASUS

Plugin yang mengirim lebih banyak daripada yang diduga

Sebuah tim menyalakan plugin observability untuk menjawab pertanyaan biaya yang berulang dari manajemen. Pemasangannya mudah, dasbornya berguna, dan pertanyaan biaya akhirnya punya jawaban.

Beberapa bulan kemudian, tim hukum menanyakan data apa saja yang meninggalkan perusahaan. Tidak ada yang dapat menjawab dengan pasti untuk plugin itu, karena tidak ada yang memeriksa apa yang dikirim — hanya apa yang ditampilkan.

YANG DIPERIKSA		YANG SEHARUSNYA
Apa yang ditampilkan	**Dasbor**	Apa yang dikirim
Apakah menyala	**Konfigurasi**	Medan apa yang disertakan
Cara memasang	**Dokumentasi**	Apa yang dikumpulkan
Apakah berguna	**Pertanyaan**	Apakah boleh keluar
Berguna	**Hasil**	Tidak diketahui

Gambar 41.3 — Dua kolom yang menguji hal berbeda. Kolom kiri menjawab pertanyaan produk; kolom kanan menjawab pertanyaan kepatuhan.

Setelah diperiksa, sebagian besar yang dikirim adalah metadata dan bukan isi percakapan. Tetapi jawaban itu datang tiga minggu setelah pertanyaannya, dan selama tiga minggu itu tim hukum tidak dapat memberi kepastian kepada siapa pun.

> **PERTANYAAN YANG PASTI DATANG**
> Sebelum menyalakan plugin yang mengirim data ke luar, tuliskan apa yang dikirim, ke mana, dan siapa yang menyetujuinya. Pertanyaan itu pasti datang; yang dapat Anda pilih hanyalah kapan Anda menjawabnya.

BAB 41 / LATIHAN

Latihan mandiri

1. Daftarkan seluruh plugin aktif di mesin Anda beserta kelas sumbernya.
2. Untuk tiap plugin yang mengirim data ke luar, tuliskan apa yang dikirim dan ke mana.
3. Matikan satu plugin yang tidak dipakai, lalu amati apakah ada yang berubah selama satu minggu.
4. Susun urutan pencarian plugin untuk tim Anda dan tuliskan siapa yang boleh naik ke tingkat komunitas.

BAB 41 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Mencari fitur di registri padahal sudah bawaan	Periksa katalog resmi lebih dulu; ia melewati tinjauan kapabilitas.
Plugin bawaan tidak aktif	Semuanya opt-in. Nyalakan secara eksplisit.
Nama telanjang memasang paket npm, bukan yang bawaan	Sebagian nama memang menuju npm. Pakai awalan eksplisit.
Data keluar ke layanan pihak ketiga	Plugin observability memang mengirim ke luar. Keputusan itu harus disengaja.
Kunci API masih teks biasa di konfigurasi	Pertimbangkan plugin penyedia rahasia seperti 1Password.

Tiga pertanyaan review

1. Mengapa plugin bawaan dan resmi tidak menuntut tinjauan kapabilitas?
2. Plugin mana yang akan Anda nyalakan lebih dulu pada penerapan perusahaan, dan mengapa?
3. Apa risiko menyalakan plugin admin HTTP RPC, dan dalam kondisi apa ia dapat dibenarkan?

Kunci pembahasan

Keduanya melalui proses rilis yang sama dengan produk intinya, sehingga tinjauan sudah terjadi di hulu; menuntut operator meninjau ulang hanya akan melatih orang untuk menekan setuju tanpa membaca. Pada penerapan perusahaan, Policy layak dinyalakan lebih dulu karena ia mengubah kebijakan menjadi sesuatu yang dapat diverifikasi berulang, dan tanpa itu seluruh keputusan keamanan di buku ini hanya bertahan selama ingatan orang yang membuatnya. Plugin admin HTTP RPC memaparkan metode control-plane lewat HTTP, sehingga ia memperluas permukaan serang secara signifikan; ia dapat dibenarkan hanya di belakang autentikasi yang kuat dan jaringan yang tertutup, misalnya untuk integrasi dasbor internal.

Rujukan: dokumentasi OpenClaw, halaman plugins/plugin-inventory, plugins/manage-plugins, dan halaman plugin masing-masing, ditinjau 17 September 2026.

PLUGIN DAN EKSTENSI

Bab 42 MCP: server, transport, dan OAuth

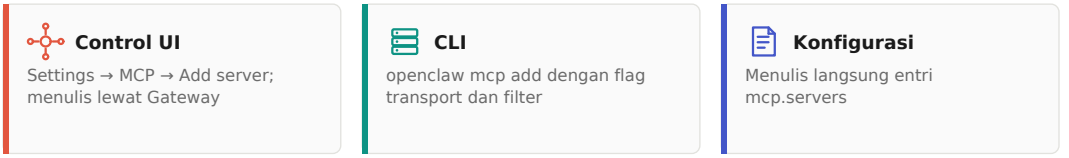
Model Context Protocol adalah cara sebuah agen meminjam tool dari program lain. Server MCP memaparkan tool, sumber daya, dan prompt; OpenClaw menyambung kepadanya dan membuat tool itu tersedia bagi agen Anda. Bab ini menutup Bagian VI.

Tujuan belajar

Menambahkan server MCP dari tiga permukaan, memilih transport, dan menangani server yang menuntut OAuth. Pada akhir bab, pembaca dapat menyambungkan layanan eksternal tanpa melewati kebijakan tool.

> **KALIMAT YANG PALING PENTING DI BAB INI**
> Definisi server tinggal di bawah mcp.servers dalam konfigurasi, dan tool yang mereka paparkan melewati kontrol profil tool dan kebijakan tool yang sama dengan yang lain — menyambungkan sebuah server tidak melewati kebijakan Anda.

Tiga permukaan untuk menambahkan



Gambar 42.1 — Tiga jalan menuju entri yang sama. Untuk header, TLS, timeout, dan filter tool, pakai editor konfigurasi berlingkup atau berkasnya.

```
openclaw mcp add docs \
  --url https://mcp.example.com/mcp \
  --transport streamable-http \
  --include 'search,read_*'

openclaw mcp doctor docs --probe
openclaw mcp status --verbose
openclaw mcp probe <name>
```

Menyimpan definisi tidak membuktikan apa pun tentang keterjangkauan; probe yang membuktikannya.

Proses Gateway atau agen yang sudah berjalan mungkin menuntut restart atau muat ulang runtime sebelum mereka mengambil definisi baru. Bila probe berhasil tetapi agen tetap tidak melihat tool-nya, inilah penjelasan yang paling sering benar.

BAB 42 / TRANSPORT

Tiga transport

Transport	Bentuk	Cocok untuk
stdio	Perintah beserta argumennya	Server lokal yang dijalankan sebagai subproses
sse	URL http atau https	Server jarak jauh gaya lama
streamable-http	URL http atau https	Server jarak jauh modern; streaming dua arah

Untuk transport HTTP, masukkan URL server. Untuk stdio, masukkan perintah diikuti argumennya — dan perhatikan bahwa argumen memang termasuk di medan argumen, bukan digabung ke dalam perintah.

BAB 42 / OAUTH

Server yang menuntut OAuth

Server MCP jarak jauh yang dilindungi OAuth tidak dapat dipakai dengan header statis, karena token akses berumur pendek. OpenClaw menyediakan alur OAuth yang menyimpan dan menyegarkan token.

```
openclaw mcp login docs
openclaw mcp logout docs
```

Logout menghapus kredensial tersimpan tetapi mempertahankan definisi servernya.

Bila penyedia memutar token atau keadaan otorisasi tersangkut, jalankan logout lalu ulangi login. Perintah logout dapat membersihkan kredensial untuk server HTTP tersimpan bahkan setelah mode OAuth dihapus dari konfigurasi, selama nama dan URL server masih mengidentifikasi entri penyimpanan kredensialnya.

Identitas bersama atau per pemanggil

```
{
  mcp: {
    servers: {
      docs: {
        url: "https://mcp.example.com/mcp",
        transport: "streamable-http",
        auth: "oauth",
        oauth: { identity: "per-requester", scope: "docs.read" },
      },
    },
  },
}
```

Nilai bawaan identity adalah shared, yang berperilaku persis seperti sebelumnya.

Pada kanal bersama, identitas bersama berarti setiap pemanggil memakai token OAuth operator. Mode per pemanggil membalik itu: pemanggil yang memanggil tool sebelum menyambungkan akun akan menerima tautan masuk untuk dirinya sendiri, bukan kredensial pemanggil lain. Setelah callback berhasil, ia mengulang panggilan tool dengan akunnya sendiri.

> **SYARAT YANG MUDAH TERLEWAT**
> Mode per pemanggil hanya aktif untuk pengirim tepercaya. Model kanal bersama OpenClaw mengandaikan daftar anggota yang dikurasi operator dan saling percaya. Bila gateway.publicOrigin tidak ada, hasil masuk akan menyebut setelan itu dan doctor melaporkan perbaikan yang sama.

BAB 42 / PRAKTIKUM

Menyambungkan server MCP dengan permukaan sesempit mungkin

Praktikum ini menyambungkan satu server MCP dan membatasi tool-nya sejak awal. Menambah tool nanti mudah; mencabut tool yang sudah dipakai orang jauh lebih sulit.

Langkah 1 — lihat dulu apa yang dipaparkan server itu

```
openclaw mcp add sementara --url <url> --transport streamable-http
openclaw mcp probe sementara
openclaw mcp doctor sementara --probe
```

Probe memperlihatkan kapabilitas nyata; menyimpan definisi tidak membuktikan apa pun.

Langkah 2 — pilih hanya yang Anda butuhkan

```
openclaw mcp add docs \
  --url <url> \
  --transport streamable-http \
  --include 'search,read_*'
```

Setiap tool yang terlihat menambah konteks pada setiap giliran dan memperbesar permukaan yang dapat dipanggil keliru.

Langkah 3 — putuskan model identitas

Pertanyaan	Bila ya	Bila tidak
Tindakan harus tercatat atas nama peminta?	per-requester	shared
Kanal dipakai bersama beberapa orang?	Pertimbangkan per-requester	shared

Pertanyaan	Bila ya	Bila tidak
gateway.publicOrigin sudah disetel?	per-requester dapat bekerja	Setel dulu sebelum mencoba
Anggota kanal saling percaya?	Syarat terpenuhi	Jangan pakai kanal bersama

Langkah 4 — uji bahwa kebijakan tool tetap berlaku

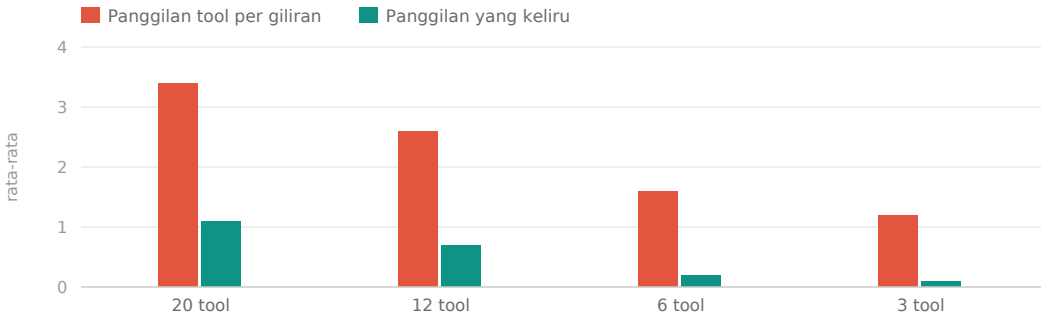
Tolak salah satu tool MCP lewat kebijakan tool, lalu minta agen memakainya. Ia harus ditolak. Membuktikan ini sendiri sekali menutup perdebatan tentang apakah menyambungkan server melewati kebijakan Anda — ia tidak.

BAB 42 / STUDI KASUS

Dua puluh tool yang membuat agen bingung

Sebuah tim menyambungkan server MCP internal yang memaparkan dua puluh tool. Mereka tidak memfilter apa pun, dengan alasan bahwa agen akan memilih yang tepat sendiri.

Kualitas jawaban menurun. Agen kadang memanggil tool pencarian ketika seharusnya memanggil tool status, kadang memanggil tiga tool berturut-turut untuk pertanyaan sederhana. Biaya naik dan latensi ikut naik.



Gambar 42.2 — Angka sintetis. Menyempitkan permukaan tool menurunkan panggilan keliru lebih cepat daripada menurunkan biaya.

Setelah difilter menjadi enam tool yang benar-benar dipakai, kualitas jawaban membaik dan biaya turun. Tidak ada satu pun kemampuan yang hilang, karena empat belas tool sisanya memang tidak pernah relevan untuk agen itu.

> **LEBIH BANYAK TOOL BUKAN LEBIH BAIK**
> Memberi agen lebih banyak pilihan tidak membuatnya lebih mampu; ia membuat pemilihannya lebih sulit. Perlakukan daftar tool seperti indeks skill di Bab 36 — sempit dan sengaja.

BAB 42 / LATIHAN

Latihan mandiri

1. Sambungkan satu server MCP dengan filter, lalu bandingkan jumlah tool yang terlihat sebelum dan sesudahnya.
2. Jawab keempat pertanyaan identitas pada praktikum langkah tiga untuk situasi Anda.
3. Buktikan sendiri bahwa kebijakan tool tetap berlaku untuk tool yang berasal dari MCP.
4. Daftarkan seluruh server MCP di penyiapan Anda beserta jumlah tool yang dipaparkan masing-masing, lalu persempit yang berlebihan.

BAB 42 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Server tersimpan tetapi tool tidak muncul	Jalankan mcp doctor --probe; lalu restart Gateway bila perlu.
Token kedaluwarsa terus-menerus	Header statis tidak cukup untuk server OAuth. Pakai auth oauth.
Otorisasi tersangkut	Jalankan mcp logout lalu login ulang.
Tautan masuk per pemanggil tidak terbit	gateway.publicOrigin belum disetel. Doctor menyebut perbaikan yang sama.
Tool MCP melewati kebijakan	Tidak mungkin. Periksa ulang kebijakan tool Anda sendiri.

Tiga pertanyaan review

1. Mengapa menyimpan definisi server tidak membuktikan apa pun, dan perintah apa yang membuktikannya?
2. Kapan identitas per pemanggil lebih tepat daripada identitas bersama?
3. Sebuah server MCP memaparkan dua puluh tool, padahal agen Anda hanya butuh tiga. Apa yang Anda lakukan dan mengapa?

Kunci pembahasan

Definisi hanyalah teks di konfigurasi; keterjangkauan dan daftar tool baru terbukti ketika probe menghubungi server, dan itulah fungsi mcp doctor --probe. Identitas per pemanggil tepat pada kanal bersama ketika tindakan harus tercatat atas nama peminta — misalnya tool yang membuat tiket — sehingga jejak auditnya benar. Untuk server dengan dua puluh tool, pakai --include supaya hanya tiga yang masuk.

Rujukan: dokumentasi OpenClaw, halaman tools/mcp, cli/mcp, dan cli/mcp/transports, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

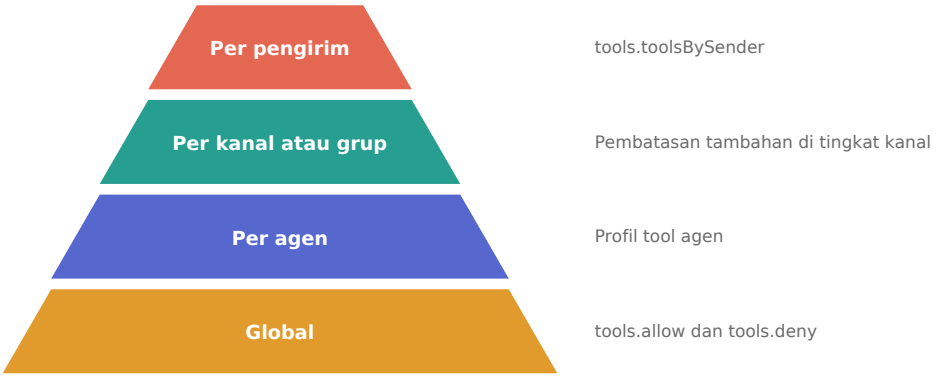
Bab 43 Kebijakan tool: profil, allow, deny, dan mode kode

Kebijakan tool adalah kontrol keamanan yang sesungguhnya. Bab 36 sudah menegaskan bahwa menyembunyikan skill tidak mencabut kemampuan; di sinilah kemampuan itu benar-benar dicabut.

Tujuan belajar

Menyusun lapisan kebijakan tool, memahami urutan menang, dan membatasi tool per pengirim maupun per penyedia. Pada akhir bab, pembaca dapat membuktikan bahwa sebuah agen tidak mungkin menjalankan perintah tertentu.

Lapisan kebijakan



Gambar 43.1 — Empat lapis. Setiap lapis hanya dapat memperketat lapis di bawahnya; penolakan selalu menang.

Pembatasan tool per grup atau per kanal diterapkan sebagai tambahan atas kebijakan tool global dan kebijakan tool agen. Penolakan tetap menang. Aturan satu kalimat ini adalah yang membuat seluruh sistem dapat dipercaya: tidak ada lapis yang dapat memberi kembali apa yang sudah ditolak di atasnya.

```
{
  tools: {
    allow: ["web", "sessions"],
    deny: ["exec"],
    byProvider: {},
    toolsBySender: {
      "*": ["web"],
    },
    elevated: {},
```

```
    codeMode: {},
  },
}
```

Kerangka blok tools. Periksa rujukan konfigurasi untuk medan lengkapnya.

BAB 43 / PERMUKAAN

Permukaan kebijakan yang tersedia

Kunci	Mengatur
`tools.allow dan tools.deny`	Lapis kebijakan dasar untuk seluruh tool
`tools.byProvider`	Pembatasan berdasarkan penyedia tool
`tools.toolsBySender`	Tool yang tersedia per pengirim pesan
`tools.elevated`	Gerbang eksekusi berhak istimewa
`tools.codeMode`	Mode kode dan batasnya
`tools.exec`	Pengaturan tool eksekusi; dibahas di Bab 44
`tools.web dan tools.media`	Pencarian, pengambilan web, dan pemahaman media
`tools.agentToAgent dan tools.sessions`	Komunikasi antar agen dan lintas sesi
`tools.loopDetection`	Deteksi pengulangan yang tidak produktif

Tool MCP dan tool plugin berada di dalam kebijakan tool yang sama, termasuk di dalam kebijakan tool sandbox. Tidak ada kategori tool yang hidup di luar sistem ini, dan itu disengaja.

> **ELEVATED ADALAH KEPUTUSAN SADAR**
> Tool elevated mati secara bawaan — tidak ada eksekusi berhak istimewa tanpa opt-in eksplisit. Bila Anda menemukan konfigurasi yang menyalakannya, pastikan ada orang yang dapat menjelaskan mengapa.

BAB 43 / PRAKTIK

Menyusun kebijakan untuk agen nyata

AGEN PELANGGAN		AGEN OPERASI
Ditolak	**exec**	Diizinkan dengan persetujuan
Diizinkan	**Pencarian web**	Diizinkan
Ditolak	**Tool sesi lintas**	Terbatas
Hanya baca	**Tool MCP internal**	Baca dan tulis
Ditolak	**elevated**	Ditolak

Gambar 43.2 — Dua profil yang berbeda tajam. Baris terakhir sama untuk keduanya, dan itu bukan kebetulan.

Prinsip yang layak dipegang: mulai dari menolak semuanya, lalu buka satu per satu berdasarkan kebutuhan yang dapat dinamai. Kebijakan yang disusun terbalik — membuka semuanya lalu menutup yang menakutkan — selalu meninggalkan sesuatu yang terlewat, karena Anda tidak dapat menutup apa yang tidak Anda ketahui ada.

BAB 43 / PRAKTIKUM

Menyusun kebijakan tool dari nol

Praktikum ini menyusun kebijakan dengan cara yang benar: mulai dari menolak semuanya, lalu membuka satu per satu berdasarkan kebutuhan yang dapat dinamai.

Langkah 1 — daftar kemampuan yang benar-benar dibutuhkan

Kembali ke empat baris yang Anda tulis pada praktikum Bab 1. Baris Perlu adalah daftar pembuka Anda; baris Tidak adalah daftar penolakan Anda. Bila Anda melewatkan latihan itu, kerjakan sekarang — seluruh bab ini menjadi jauh lebih mudah dengan keduanya di tangan.

Langkah 2 — mulai dari tertutup

```
{
  tools: {
    deny: ["exec", "process"],
    allow: ["web"],
    elevated: {},
  },
}
```

Tool elevated mati secara bawaan; biarkan begitu sampai ada alasan yang dapat dinamai.

Langkah 3 — buktikan penolakan benar-benar berlaku

Uji	Cara	Hasil yang benar
Tool ditolak global	Minta agen memakainya	Ditolak, terlihat di jejak
Kanal tidak dapat melonggarkan	Buka tool itu di tingkat kanal	Tetap ditolak
Agen lain tidak terpengaruh	Uji agen kedua	Kebijakannya sendiri berlaku
Tool MCP ikut tunduk	Minta tool dari server MCP yang ditolak	Ditolak

Langkah 4 — tuliskan kebijakan tiap agen dalam satu baris

```
# Contoh isian
agen cs   : web; tanpa exec; tanpa tool sesi lintas; MCP hanya baca
agen ops  : web, exec dengan allowlist dan persetujuan; MCP baca-tulis
agen riset: web; tanpa exec; MCP hanya baca; sandbox penuh
```

Satu baris per agen yang dapat dibaca orang non-teknis adalah dokumen keamanan paling berguna yang dapat Anda buat.

BAB 43 / STUDI KASUS

Kebijakan yang disusun terbalik

Sebuah tim menyusun kebijakan tool dengan cara yang terasa alami: mereka membuka semuanya supaya agen dapat bekerja, lalu menutup tool yang terasa menakutkan satu per satu ketika mereka memikirkannya.

Setelah beberapa bulan, daftar penolakan mereka berisi sembilan tool. Ketika seorang auditor bertanya apa saja yang dapat dilakukan agen itu, tidak ada yang dapat menjawab — jawabannya adalah segala sesuatu kecuali sembilan hal, dan tidak ada yang tahu berapa panjang daftar segala sesuatu itu.

MULAI TERBUKA		MULAI TERTUTUP
Penolakan	**Daftar yang dipelihara**	Pembukaan
Sulit dijawab	**Pertanyaan auditor**	Dijawab langsung
Otomatis terbuka	**Tool baru dari pembaruan**	Tetap tertutup
Otomatis terbuka	**Plugin baru**	Tetap tertutup
Yang tidak Anda ketahui ada	**Yang terlewat**	Tidak ada

Gambar 43.3 — Baris ketiga dan keempat adalah alasan sesungguhnya: kebijakan terbuka membuka hal-hal yang belum ada saat Anda menulisnya.

Baris ketiga yang akhirnya menggigit mereka. Sebuah pembaruan menambahkan tool baru, dan tool itu langsung tersedia bagi seluruh agen — termasuk agen layanan pelanggan — karena kebijakan mereka hanya menolak yang sudah mereka pikirkan.

MENGAPA ARAHNYA MENENTUKAN

Kebijakan yang dimulai dari tertutup tidak dapat dilampaui oleh hal yang belum ada. Itu sifat yang tidak dimiliki kebijakan yang dimulai dari terbuka, dan itulah alasan sesungguhnya memilih arah ini.

BAB 43 / LATIHAN

Latihan mandiri

1. Susun ulang kebijakan tool Anda dari tertutup, memakai daftar Perlu dari praktikum Bab 1.
2. Jalankan keempat uji pada praktikum langkah tiga dan simpan jejaknya sebagai bukti.
3. Tuliskan kebijakan tiap agen Anda dalam satu baris yang dapat dibaca orang non-teknis.
4. Periksa apakah kebijakan Anda saat ini dimulai dari terbuka atau tertutup, lalu hitung berapa banyak tool yang sebenarnya tersedia.

BAB 43 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Tool tetap tersedia meski ditolak di kanal	Periksa lapis global dan agen; kanal hanya menambah pembatasan.
Membuka tool di kanal tidak berpengaruh	Lapis kanal hanya memperketat. Longgarkan di lapis yang tepat.
Agen pelanggan dapat menjalankan perintah	exec belum ditolak untuk agen itu. Pisahkan profilnya.
Tool MCP tidak tunduk kebijakan	Ia tunduk. Periksa ulang kebijakan yang benar-benar aktif.
Bingung tool mana yang aktif	Periksa kebijakan efektif, bukan hanya berkas konfigurasi.

Tiga pertanyaan review

1. Mengapa lapis yang lebih spesifik hanya boleh memperketat, tidak pernah melonggarkan?
2. Apa perbedaan praktis antara toolsBySender dan kebijakan per agen?
3. Buktikan, dalam tiga langkah, bahwa agen layanan pelanggan Anda tidak mungkin menjalankan perintah shell.

Kunci pembahasan

Bila lapis spesifik dapat melonggarkan, maka siapa pun yang menguasai satu kanal atau satu grup dapat mengembalikan kemampuan yang sengaja dicabut di tingkat global, dan kebijakan keamanan berhenti menjadi jaminan. toolsBySender membedakan berdasarkan siapa yang mengirim pesan dalam satu agen yang sama, sedangkan kebijakan per agen berlaku untuk seluruh percakapan agen itu tanpa memandang pengirim. Tiga langkah pembuktian: tunjukkan exec ada di tools.deny pada lapis global atau profil agen itu, tunjukkan tidak ada lapis lain yang dapat melonggarkannya karena penolakan menang, lalu picu percobaan nyata di lingkungan uji dan tunjukkan penolakannya.

Rujukan: dokumentasi OpenClaw, halaman gateway/config-tools/tool-policy dan gateway/sandbox-vs-tool-policy-vs-elevated, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

Bab 44 Exec, proses latar, dan persetujuan perintah

Tool exec adalah tempat agen berhenti menjadi penasihat dan mulai menjadi pelaku. Bab ini membahas gerbang yang mengelilinginya, dan mengapa gerbang itu berlapis.

Tujuan belajar

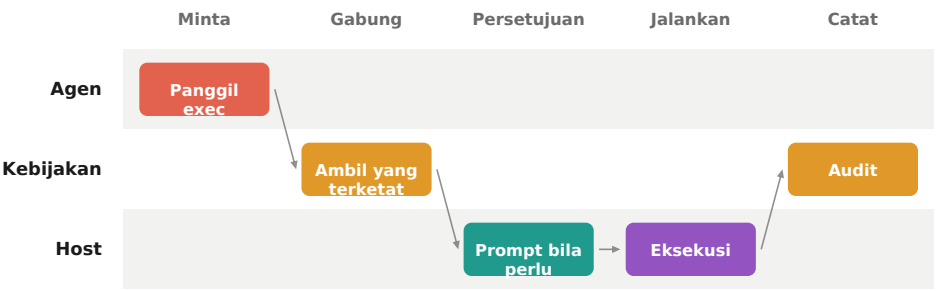
Menyusun kebijakan eksekusi, memahami bahwa persetujuan hanya dapat memperketat, dan mengetahui di mana persetujuan ditegakkan. Pada akhir bab, pembaca dapat memberi agen kemampuan menjalankan perintah tanpa kehilangan kendali.

Dua setelan yang berpasangan

Setelan	Mengatur
security	Seberapa bebas perintah boleh dipilih: full atau allowlist atau off
ask	Kapan operator dimintai persetujuan: always, on-miss, atau off

ATURAN PENGGABUNGAN
Kebijakan efektif adalah yang lebih ketat di antara tools.exec dan default approvals: approvals hanya dapat memperketat security dan ask yang berasal dari konfigurasi, tidak pernah melonggarkannya. Bila sebuah medan approvals dihilangkan, nilai tools.exec yang dipakai.

Menyetel tools.exec.mode ke full saja tidak melewati setelan ini. Pada Gateway, sesi berizin penuh dengan security efektif full dan ask off melewatkan evaluasi persetujuan host. Eksekusi elevated-full yang diizinkan juga melewatkannya ketika kebijakan exec dan kebijakan persetujuan host sama-sama mengizinkan full atau off.



Gambar 44.1 — Persetujuan ditegakkan secara lokal pada host eksekusi, bukan di tempat permintaan berasal.

BAB 44 / HOST

Di mana persetujuan ditegakkan

Persetujuan eksekusi ditegakkan secara lokal pada host eksekusi. Untuk host Gateway, itu berarti proses openclaw pada mesin Gateway. Eksekusi di node tetap memeriksa kebijakan target sebelum dispatch: allowlist atau off pada pemanggil menolak perintah yang tidak cocok, dan ask always pada target menuntut persetujuan meskipun pemanggil meminta full atau off.

```
{
  tools: {
    exec: {
      host: "node",
      security: "allowlist",
      node: "build-node",
    },
  },
}
```

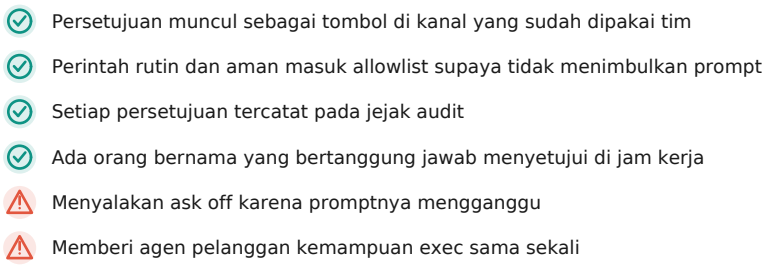
Mengarahkan seluruh panggilan exec ke node tertentu dengan mode allowlist.

Host exec juga memakai keadaan persetujuan lokal pada mesin itu. Sebuah ask always yang tersimpan pada dokumen persetujuan host eksekusi akan terus meminta persetujuan meskipun default sesi atau konfigurasi meminta on-miss. Node yang tidak dikonfigurasi memakai dasar full atau off yang sama dengan Gateway.

BAB 44 / PRAKTIK

Membuat persetujuan benar-benar dipakai

Gerbang persetujuan yang merepotkan akan dimatikan orang dalam seminggu. Kuncinya adalah menempatkan persetujuan di tempat orang sudah berada — yaitu di kanal percakapan, sebagai tombol, seperti dibahas di Bab 29 untuk Telegram.



Gambar 44.2 — Enam praktik. Dua yang terakhir adalah cara paling umum gerbang persetujuan berhenti bermakna.

Proses latar layak disebut terpisah. Eksekusi latar dan tool proses memungkinkan pekerjaan panjang berjalan tanpa memblokir giliran, dan pekerjaan itu tetap tunduk pada kebijakan yang sama. Bedanya, kegagalannya lebih mudah luput dari perhatian — jadi pastikan ada jalur pelaporan ketika sebuah proses latar gagal.

BAB 44 / PRAKTIKUM

Membuat gerbang persetujuan yang benar-benar dipakai

Gerbang persetujuan yang merepotkan akan dimatikan orang dalam seminggu. Praktikum ini menyeimbangkan keamanan dengan kelayakan pakai, karena gerbang yang dimatikan adalah gerbang yang tidak ada.

Langkah 1 — pahami kebijakan efektif Anda

```
openclaw approvals
openclaw exec-policy
```

Kebijakan efektif adalah yang lebih ketat di antara tools.exec dan default approvals.

Langkah 2 — pisahkan perintah rutin dari perintah berisiko

Kelas	Contoh	Perlakuan
Rutin dan aman	Membaca status, memeriksa log	Masuk allowlist; tanpa prompt
Rutin dan mengubah	Merestart layanan yang terdaftar	Prompt sekali per sesi
Jarang dan berisiko	Mengubah konfigurasi produksi	Prompt setiap kali
Destruktif	Menghapus data	Tidak pernah di allowlist

Sebagian besar kelelahan persetujuan berasal dari baris pertama yang tidak dipisahkan. Memindahkan perintah baca ke allowlist menghilangkan mayoritas prompt tanpa mengurangi keamanan sedikit pun.

Langkah 3 — tempatkan prompt di tempat orang sudah berada

Persetujuan yang muncul sebagai tombol di kanal yang sudah dipakai tim akan ditekan; persetujuan yang menuntut membuka terminal tidak akan. Bab 29 membahas tombol persetujuan pada Telegram, dan Slack punya padanannya.

Langkah 4 — uji penegakan di host eksekusi

```
# Pada penyiapan node, uji keduanya:
1. Pemanggil meminta full, target menyetel ask always
2. Pemanggil menyetel allowlist, target menerima perintah di luar daftar
```

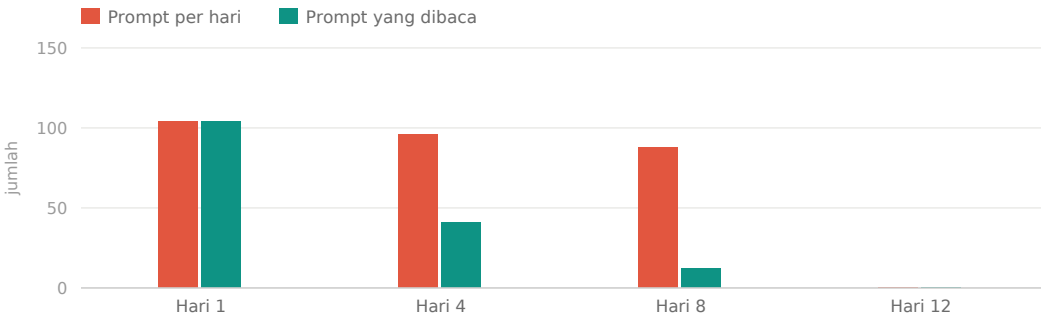
Keduanya harus ditolak. Kebijakan target diperiksa sebelum dispatch.

BAB 44 / STUDI KASUS

Gerbang yang dimatikan pada minggu kedua

Sebuah tim operasi memasang persetujuan untuk setiap perintah yang dijalankan agen. Niatnya benar dan rancangannya ketat: tidak ada satu pun perintah yang berjalan tanpa seorang manusia menekan setuju.

Pada hari pertama, jumlah prompt mencapai lebih dari seratus. Sebagian besar untuk perintah membaca status yang dijalankan agen puluhan kali sehari. Pada hari kesepuluh, seseorang menyetel mode tanya menjadi mati supaya pekerjaan dapat berjalan.



Gambar 44.3 — Angka sintetis. Persetujuan berhenti dibaca jauh sebelum ia dimatikan; grafik kedua adalah peringatan yang terlewat.

Yang paling berbahaya bukan hari kedua belas melainkan hari kedelapan, ketika prompt masih muncul tetapi hampir tidak ada yang benar-benar dibaca. Gerbang yang ditekan tanpa dibaca memberi ilusi kontrol tanpa kontrolnya.

METRIK YANG LAYAK DIPANTAU

Pantau bukan hanya apakah persetujuan menyala, melainkan berapa lama rata-rata orang memutuskan. Waktu keputusan yang turun di bawah beberapa detik adalah tanda gerbang Anda sudah berhenti bermakna.

BAB 44 / LATIHAN

Latihan mandiri

1. Klasifikasikan seluruh perintah yang dijalankan agen Anda memakai empat kelas pada praktikum langkah dua.
2. Pindahkan perintah rutin dan aman ke allowlist, lalu hitung penurunan jumlah prompt.
3. Tempatkan persetujuan sebagai tombol di kanal yang sudah dipakai tim Anda.
4. Tetapkan cara memantau apakah persetujuan masih benar-benar dibaca, bukan hanya ditekan.

BAB 44 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Prompt persetujuan tidak pernah muncul	Sesi mungkin berizin penuh dengan ask off. Periksa kebijakan efektif.
Prompt muncul terus meski disetel on-miss	Ada ask always pada dokumen persetujuan host eksekusi.
Perintah ditolak di node meski pemanggil full	Kebijakan target diperiksa sebelum dispatch. Itu benar.
Tim mematikan persetujuan karena merepotkan	Pindahkan ke tombol di kanal dan perluas allowlist perintah rutin.
Proses latar gagal tanpa ada yang tahu	Tambahkan jalur pelaporan kegagalan untuk pekerjaan latar.

Tiga pertanyaan review

1. Mengapa persetujuan hanya dapat memperketat, tidak pernah melonggarkan konfigurasi?
2. Mengapa persetujuan ditegakkan pada host eksekusi, bukan di tempat permintaan berasal?
3. Rancang kebijakan exec untuk agen operasi yang harus dapat merestart layanan tetapi tidak boleh menghapus data.

Kunci pembahasan

Bila approvals dapat melonggarkan, maka sebuah sesi dapat memberi dirinya kewenangan yang ditolak konfigurasi, sehingga konfigurasi berhenti menjadi batas. Penegakan di host eksekusi penting karena mesin itulah yang menanggung akibatnya; mesin yang meminta tidak tahu apa arti sebuah perintah pada sistem berkas mesin tujuan. Untuk agen operasi, pakai security allowlist dengan daftar perintah restart yang eksplisit, ask on-miss supaya

perintah di luar daftar memicu persetujuan manusia, dan pastikan perintah destruktif tidak pernah masuk allowlist — sehingga penghapusan data selalu menuntut persetujuan sadar, bukan sekadar tidak dianjurkan.

Rujukan: dokumentasi OpenClaw, halaman tools/exec-approvals, gateway/background-process, dan nodes/command-policy, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

Bab 45 Menjalankan perintah di node dan memindahkan berkas

Bab 12 memperkenalkan node sebagai periferal. Bab ini memakainya sebagai tempat kerja: menjalankan perintah di mesin lain, memindahkan berkas, dan mengetahui Mac mana yang sedang Anda pakai.

Tujuan belajar

Mengarahkan eksekusi ke node tertentu, menyusun allowlist perintah, dan memakai kehadiran untuk merutekan peringatan. Pada akhir bab, pembaca dapat memisahkan tempat Gateway berjalan dari tempat pekerjaan dilakukan.

Mengapa memisahkan tempat eksekusi

Pakai host node ketika Gateway Anda berjalan di satu mesin dan Anda ingin perintah dijalankan di mesin lain. Pola ini muncul secara alami begitu Gateway dipindahkan ke server, sementara pekerjaan yang sesungguhnya — membangun proyek, menyentuh berkas desain, memakai aplikasi berlisensi — harus terjadi di Mac seseorang.

```
{
  tools: {
    exec: {
      host: "node",
      security: "allowlist",
      node: "build-node",
    },
  },
  gateway: {
    nodes: {
      allowCommands: ["system.run"],
      denyCommands: ["camera.clip"],
    },
  },
}
```

host node merutekan seluruh panggilan exec ke node; node memilih node tertentu.

NAMA HARUS PERSIS

Pakai nama perintah node yang persis. Penolakan memblokir nama yang persis meskipun default atau allowCommands menyertakannya, sehingga Anda dapat membuka satu keluarga lalu mengunci satu anggotanya.

BAB 45 / BERKAS

Memindahkan berkas

Pemindahan berkas node mencakup unggahan lewat terminal node serta plugin File Transfer yang menyediakan tool untuk mendaftar direktori, mengambil, dan menulis. Karena ini menyentuh sistem berkas mesin lain, ia tunduk pada kebijakan yang sama dengan eksekusi.

```
openclaw nodes status
openclaw nodes describe --node <idOrNameOrIp>
openclaw file-transfer
```

Perintah file-transfer juga meninjau dan memigrasikan persetujuan berdiri.

Kehadiran: Mac mana yang sedang dipakai

OpenClaw dapat mendeteksi Mac yang paling terakhir Anda pakai dan merutekan peringatan node ke sana. Untuk seseorang dengan MacBook dan Mac mini, ini perbedaan antara notifikasi yang muncul di depan mata dan notifikasi yang berbunyi di ruangan kosong.

Gambar 45.1 — Kehadiran memilih tujuan. Tanpa itu, peringatan mendarat di tempat yang kebetulan terdaftar lebih dulu.

BAB 45 / PRAKTIKUM

Memindahkan pekerjaan ke mesin yang tepat

Praktikum ini memisahkan tempat Gateway berjalan dari tempat pekerjaan dilakukan. Pemisahan ini muncul secara alami begitu Gateway dipindahkan ke server, dan lebih baik Anda menyiapkannya sebelum dibutuhkan.

Langkah 1 — pastikan node sehat dan kapabilitasnya jelas

```
openclaw nodes status
openclaw nodes describe --node <idOrNameOrIp>
```

Daftarkan kapabilitas yang muncul; itu yang benar-benar dapat dipanggil.

Langkah 2 — arahkan eksekusi dan persempit daftarnya

```
{
  tools: { exec: {
    host: "node",
    security: "allowlist",
    node: "build-node",
  } },
  gateway: { nodes: {
    allowCommands: ["system.run"],
    denyCommands: ["camera.clip", "screen.record"],
  } },
}
```

Pakai nama perintah yang persis; penolakan menang atas allowCommands.

Langkah 3 — uji kedua arah kebijakan

Uji	Cara	Hasil yang benar
Pemanggil dibatasi	Minta perintah di luar allowlist pemanggil	Ditolak
Target dibatasi	Setel ask always di node, minta dari pemanggil full	Tetap menuntut persetujuan
Penolakan menang	Masukkan satu perintah ke kedua daftar	Ditolak
Node yang dipin	Panggil ketika node itu mati	Gagal jelas, tidak berpindah

Langkah 4 — atur kehadiran supaya peringatan mendarat di tempat yang benar

Bila Anda memakai lebih dari satu Mac, pastikan peringatan node mendarat pada mesin yang sedang Anda pakai. Peringatan yang berbunyi di ruangan kosong sama tidak bergunanya dengan peringatan yang tidak dikirim.

BAB 45 / STUDI KASUS

Perintah yang berjalan di mesin yang salah

Sebuah tim memindahkan Gateway ke server Linux dan memasangkan dua Mac sebagai node: satu Mac build dan satu Mac desain. Mereka menyetel host eksekusi ke node tanpa memin node tertentu.

Sebuah pekerjaan build berjalan di Mac desain, karena node itulah yang kebetulan tersedia saat perintah dikirim. Toolchain yang berbeda menghasilkan artefak yang berbeda pula, dan artefak itu sempat masuk ke rilis sebelum ketahuan.

TANPA PIN NODE		DENGAN PIN NODE
Mana saja yang tersedia	**Node yang dipakai**	Yang Anda sebut
Bergantung mesin	**Hasil build**	Dapat direproduksi
Berpindah diam-diam	**Bila node mati**	Gagal dengan jelas
Sulit; hasil berubah-ubah	**Diagnosis**	Langsung terlihat
Pekerjaan yang setara di mana pun	**Cocok untuk**	Pekerjaan yang butuh lingkungan tertentu

Gambar 45.2 — Kegagalan yang jelas lebih berharga daripada keberhasilan yang tidak dapat direproduksi.

Mereka kemudian memin node untuk seluruh pekerjaan build. Ketika node itu mati, pekerjaan gagal dengan pesan yang jelas alih-alih berpindah diam-diam ke mesin yang salah — dan itu perbaikan, bukan kemunduran.

> **KETAHANAN YANG SEBENARNYA MERUGIKAN**
> Untuk pekerjaan yang hasilnya bergantung pada lingkungan, pin node-nya. Berpindah otomatis terdengar seperti ketahanan, tetapi ia menukar kegagalan yang terlihat dengan kesalahan yang tidak terlihat.

BAB 45 / LATIHAN

Latihan mandiri

1. Pasangkan satu node dan arahkan eksekusi kepadanya dengan mode allowlist.
2. Jalankan keempat uji pada praktikum langkah tiga dan catat hasil masing-masing.
3. Tentukan pekerjaan mana di penyiapan Anda yang harus dipin ke node tertentu dan mengapa.
4. Uji apa yang terjadi ketika node yang dipin sedang mati, lalu pastikan pesannya cukup jelas bagi orang lain di tim Anda.

BAB 45 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
exec berjalan di Gateway padahal ingin di node	Setel tools.exec.host ke node.
Perintah node ditolak meski ada di allowCommands	Ada entri denyCommands yang cocok persis. Penolakan menang.
Peringatan mendarat di Mac yang salah	Kehadiran belum dipakai, atau Mac itu memang yang terakhir aktif.
Pemindahan berkas gagal	Tunduk kebijakan yang sama dengan eksekusi. Periksa izin dan allowlist.
Node tidak muncul di status	Belum dipasangkan. Periksa devices dan nodes pending.

Tiga pertanyaan review

1. Kapan memisahkan host eksekusi dari host Gateway memberi keuntungan nyata?
2. Mengapa pemindahan berkas tunduk pada kebijakan yang sama dengan eksekusi?
3. Rancang penyiapan untuk tim yang Gateway-nya di server Linux tetapi pekerjaannya di Mac.

Kunci pembahasan

Pemisahan menguntungkan ketika pekerjaan menuntut lingkungan yang hanya ada di mesin tertentu — aplikasi berlisensi, toolchain khusus, atau akses jaringan yang berbeda — sementara Gateway sendiri lebih baik berada di mesin yang selalu hidup. Pemindahan berkas tunduk kebijakan yang sama karena menulis berkas ke mesin lain memiliki dampak yang setara dengan menjalankan perintah di sana; membedakannya akan menciptakan celah. Untuk tim itu: Gateway di server Linux, Mac dipasangkan sebagai node, tools.exec.host disetel ke node dengan security allowlist, dan perintah yang diizinkan dibatasi pada yang benar-benar dibutuhkan alur kerja mereka.

Rujukan: dokumentasi OpenClaw, halaman nodes/node-exec, nodes/file-transfers, nodes/presence, dan nodes/command-policy, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

Bab 46 Automations: jadwal, webhook, dan heartbeat

Agen yang hanya menjawab ketika ditanya adalah setengah dari yang dijanjikan. Bab ini membahas mekanisme yang membuatnya bekerja tanpa diminta — dan cara memilih mekanisme yang tepat, karena memilih yang salah menghasilkan biaya atau keterlambatan.

Tujuan belajar

Membedakan automations dari heartbeat, membuat pekerjaan terjadwal, dan memahami risiko eksekusi tanpa pengawasan. Pada akhir bab, pembaca dapat menjadwalkan pekerjaan tanpa membayar untuk keheningan.

Memilih mekanisme

Kebutuhan	Pilih	Alasan
Laporan harian tepat pukul sembilan	Automation terjadwal	Waktu persis, eksekusi terisolasi
Ingatkan saya dua puluh menit lagi	Automation sekali jalan	Satu tembakan dengan waktu presisi
Periksa kotak masuk tiap tiga puluh menit	Heartbeat	Digabung dengan pemeriksaan lain, sadar konteks
Pantau kalender untuk acara mendatang	Heartbeat	Cocok untuk kesadaran berkala
Lacak pekerjaan terlepas	Background task	Ledger tugas mencatat seluruh pekerjaan terlepas
Orkestrasi alur bertahap	Task Flow	Alur multi-langkah yang tahan lama
Wewenang operasi permanen	Standing order	Dimuat setiap sesi lewat berkas workspace

Automations adalah penjadwal bawaan OpenClaw untuk seluruh pekerjaan berulang dan sekali jalan, termasuk monitor heartbeat. Penjadwal mempertahankan pekerjaan, membangunkan agen pada waktu yang tepat, dan dapat mengirimkan keluaran kembali ke kanal percakapan atau endpoint webhook.

BAB 46 / HEARTBEAT

Heartbeat sebagai monitor sistem

Heartbeat adalah automation monitor yang dimiliki sistem, menjalankan giliran sesi utama secara berkala — bawaannya setiap tiga puluh menit. Ia dapat memakai konteks catatan monitor yang kecil untuk memunculkan hal yang perlu perhatian tanpa membuat catatan tugas terlepas atau memperpanjang kesegaran sesi.

HEARTBEAT		AUTOMATION
Setiap N menit	**Pemicu**	Jadwal spesifik
HEARTBEAT.md	**Yang dibaca**	Pesan atau perintah sendiri
Melanjutkan sesi utama	**Sesi**	Dapat terisolasi
Berjalan meski menganggur	**Biaya**	Hanya saat terjadwal
Pemantauan reaktif	**Cocok untuk**	Laporan terjadwal

Gambar 46.1 — Aturan praktis: heartbeat untuk pemantauan, automation untuk pekerjaan yang waktunya pasti.

HEARTBEAT TIDAK MEMOTONG ANTREAN

Catatan monitor yang kosong dilewati sebagai berkas heartbeat kosong. Giliran monitor terjadwal ditunda selama antrean utama atau pekerjaan automation sedang sibuk, selama ada jalannya yang lain untuk agen yang sama, atau ketika sesi target punya pekerjaan aktif maupun mengantre.

BAB 46 / RISIKO

Eksekusi tanpa pengawasan

INI PERMUKAAN PALING BERISIKO DI BAGIAN INI
Skrip pemicu kondisi dan payload skrip berjalan tanpa pengawasan secara bawaan dengan kebijakan tool penuh milik agen pemiliknya, termasuk exec. Jadwal stream juga membiarkan perintah yang ditulis operator berjalan tanpa pengawasan. Perlakukan permukaan ini sebagai eksekusi kode tanpa pengawasan dengan izin agen tersebut.

Kalimat itu layak dibaca ulang. Sebuah pekerjaan terjadwal yang salah tulis dapat menjalankan perintah pada pukul tiga pagi tanpa seorang pun melihatnya sampai pagi. Karena itu, batasi kebijakan tool agen yang memiliki pekerjaan terjadwal, bukan hanya isi pekerjaannya.

Menulis pemantau yang jujur

Tulis pemantau di sekitar keadaan yang dapat ditindaklanjuti, bukan hanya di sekitar keberhasilan. Pemantau yang diam ketika pemeriksaannya gagal atau kehabisan waktu akan terlihat sehat padahal rusak. Bandingkan pengamatan dengan keadaan pemicu dan kembalikan keadaan yang segar untuk deduplikasi; jangan mengandalkan ingatan model atau proses.

Ketika sebuah pekerjaan menyala, buat pesannya mandiri, karena pesan itu menjadi seluruh konteks peristiwa bagi jalannya pekerjaan tersebut. Pekerjaan yang mengandalkan konteks percakapan sebelumnya akan berperilaku berbeda ketika dijalankan terisolasi.

```
openclaw automations list
openclaw automations get <id>
openclaw automations edit <id>
openclaw automations remove <id>

openclaw tasks list
openclaw tasks audit
```

Automations mengelola jadwal; tasks adalah ledger yang mencatat, bukan penjadwal.

Operator yang butuh penghentian total dapat mematikan pemicu automation lewat konfigurasi. Menghapus atau menonaktifkan sebuah pekerjaan selama evaluasi kondisi akan membatalkan evaluasi itu sebelum payload-nya sempat mulai.

BAB 46 / PRAKTIKUM

Menjadwalkan pekerjaan tanpa membayar untuk keheningan

Praktikum ini memilih mekanisme yang tepat untuk tiap pekerjaan, lalu menulis pemantau yang jujur tentang kegagalannya sendiri.

Langkah 1 — klasifikasikan pekerjaan Anda

Pekerjaan Anda	Mekanisme	Alasan
Laporan pada jam tertentu	Automation terjadwal	Waktu persis, terisolasi
Pengingat sekali jalan	Automation sekali jalan	Satu tembakan presisi
Pemantauan berkala	Heartbeat	Digabung, sadar konteks
Melacak pekerjaan terlepas	Background task	Ledger, bukan penjadwal
Alur bertahap tahan lama	Task Flow	Orkestrasi multi-langkah

Langkah 2 — hitung biaya heartbeat Anda

Heartbeat berjalan meski tidak ada apa-apa. Kalikan frekuensinya dengan biaya satu giliran, lalu bandingkan dengan nilai yang benar-benar ia hasilkan. Bagi sebagian pembaca, perhitungan ini akan mengubah pengaturannya hari itu juga.

Langkah 3 — tulis pemantau yang melaporkan kegagalannya sendiri

```
# Buruk : lapor bila ada anomali
# Baik  : lapor bila ada anomali,
#         DAN lapor bila pemeriksaan gagal atau kehabisan waktu
```

Pemantau yang diam ketika pemeriksaannya gagal akan terlihat sehat padahal rusak.

Langkah 4 — batasi agen yang memiliki pekerjaan terjadwal

```
openclaw automations list
openclaw automations get <id>
openclaw tasks list
openclaw tasks audit
```

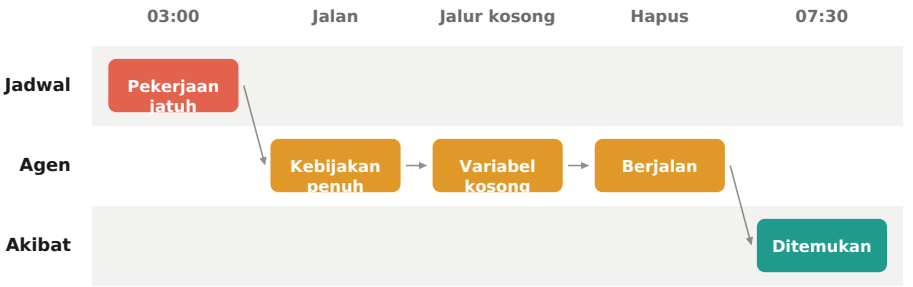
Payload skrip berjalan tanpa pengawasan dengan kebijakan tool penuh milik agen pemiliknya, termasuk exec.

BAB 46 / STUDI KASUS

Pekerjaan pukul tiga pagi yang tidak ada yang lihat

Sebuah tim membuat pekerjaan terjadwal untuk membersihkan berkas sementara setiap malam. Skripnya sederhana dan sudah diuji manual. Mereka memasangnya pada agen yang juga dipakai untuk pekerjaan operasi lain.

Sebuah perubahan kecil pada skrip — satu variabel jalur yang tidak terisi karena lingkungan berbeda — membuat perintah pembersihan berjalan pada direktori yang salah. Ia berjalan pukul tiga pagi, selesai dalam hitungan detik, dan tidak ada yang melihatnya sampai pagi.



Gambar 46.2 — Empat setengah jam antara kejadian dan penemuan. Tidak ada manusia di jalur mana pun.

Dua hal yang seharusnya berbeda. Pertama, agen pemilik pekerjaan itu seharusnya punya kebijakan tool yang jauh lebih sempit daripada agen operasi umum. Kedua, pesan pekerjaan seharusnya mandiri dan memvalidasi asumsinya sendiri sebelum bertindak.

> **BATASI AGENNYA, BUKAN HANYA PEKERJAANNYA**
> Payload skrip dan pemicu kondisi berjalan tanpa pengawasan dengan izin penuh agen pemiliknya. Perlakukan permukaan itu sebagai eksekusi kode tanpa pengawasan, dan batasi agennya — bukan hanya isi pekerjaannya.

BAB 46 / LATIHAN

Latihan mandiri

1. Klasifikasikan seluruh pekerjaan otomatis Anda memakai tabel praktikum langkah satu.
2. Hitung biaya heartbeat harian Anda dan bandingkan dengan nilai yang dihasilkannya.
3. Periksa setiap pemantau Anda dan pastikan ia melaporkan kegagalan pemeriksaannya sendiri, bukan hanya temuannya.
4. Daftarkan agen mana yang memiliki pekerjaan terjadwal, lalu periksa apakah kebijakan tool-nya sesempit yang seharusnya.

BAB 46 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Biaya token tinggi tanpa percakapan	Heartbeat berjalan meski menganggur. Pertimbangkan automation terjadwal.
Pekerjaan terjadwal terlambat	Giliran monitor ditunda saat antrean sibuk; periksa kepadatan pekerjaan.
Pemantau terlihat sehat padahal rusak	Ia hanya melapor saat berhasil. Tulis di sekitar keadaan yang dapat ditindaklanjuti.
Pekerjaan berperilaku beda saat terisolasi	Pesannya tidak mandiri. Sertakan seluruh konteks yang dibutuhkan.
Perintah berjalan tanpa persetujuan di malam hari	Payload skrip berjalan tanpa pengawasan dengan izin penuh agen. Batasi agennya.

Tiga pertanyaan review

1. Mengapa pesan pada pekerjaan terjadwal harus mandiri?
2. Jelaskan mengapa pemantau yang hanya melapor saat berhasil berbahaya.
3. Rancang pembatasan untuk agen yang memiliki lima pekerjaan terjadwal, dua di antaranya menyentuh sistem produksi.

Kunci pembahasan

Pesan pekerjaan terjadwal menjadi seluruh konteks peristiwa bagi jalannya pekerjaan itu, sehingga pekerjaan yang mengandalkan percakapan sebelumnya akan kehilangan informasi ketika dijalankan terisolasi. Pemantau yang hanya melapor saat berhasil menghasilkan keheningan yang ambigu: diam bisa berarti tidak ada masalah, atau berarti pemantaunya sendiri mati — dan Anda tidak dapat membedakannya. Untuk agen dengan lima pekerjaan, pisahkan dua pekerjaan produksi ke agen tersendiri dengan kebijakan tool yang jauh lebih ketat, karena payload skrip berjalan dengan izin penuh agen pemiliknya dan tanpa pengawasan.

Rujukan: dokumentasi OpenClaw, halaman automation, automation/cron-jobs, automation/cron-jobs/schedules, dan automation/standing-orders, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

Bab 47 Standing order, Task Flow, dan ledger tugas

Automations menjadwalkan pekerjaan. Standing order memberi wewenang. Task Flow mengorkestrasi langkahnya. Ledger tugas mencatat apa yang benar-benar terjadi. Empat mekanisme yang saling melengkapi, dan sering tertukar.

Tujuan belajar

Menulis standing order yang dapat ditegakkan, memahami ledger tugas sebagai catatan bukan penjadwal, dan mengetahui kapan Task Flow diperlukan. Pada akhir bab, pembaca dapat memberi agen wewenang operasi permanen dengan batas yang jelas.

Standing order

Standing order memberi agen wewenang operasi permanen untuk program yang didefinisikan. Ia tinggal di berkas workspace — biasanya AGENTS.md — dan disuntikkan ke setiap sesi, lalu digabungkan dengan pekerjaan terjadwal untuk penegakan berbasis waktu.

```
## Program: Laporan Status Mingguan
**Wewenang:** Kumpulkan data, susun laporan, kirim ke pemangku kepentingan
**Pemicu:** Setiap Jumat pukul 16.00 (ditegakkan lewat automation)
**Gerbang persetujuan:** Tidak ada untuk laporan standar.
                        Tandai anomali untuk tinjauan manusia.
**Eskalasi:** Bila sumber data tidak tersedia atau metrik tampak tidak wajar

### Langkah eksekusi
1. Tarik metrik dari sumber yang dikonfigurasi
2. ...
```

Bentuk standing order: wewenang, pemicu, gerbang persetujuan, eskalasi, langkah.

> **LETAKKAN DI TEMPAT YANG DIMUAT**
> Letakkan standing order di AGENTS.md untuk menjamin ia dimuat setiap sesi. Bootstrap workspace otomatis menyuntikkan AGENTS.md, SOUL.md, TOOLS.md, IDENTITY.md, USER.md, HEARTBEAT.md, BOOTSTRAP.md, dan MEMORY.md — tetapi bukan berkas sembarangan di subdirektori.

Empat medan pertama pada contoh itu layak dijadikan pola tetap. Wewenang menyatakan apa yang boleh, gerbang persetujuan menyatakan kapan manusia dilibatkan, dan eskalasi menyatakan apa yang terjadi ketika asumsinya salah. Standing order tanpa medan eskalasi akan menghasilkan agen yang memaksakan langkah pada data yang rusak.

BAB 47 / LEDGER

Tugas latar adalah catatan, bukan penjadwal

Ledger tugas latar melacak seluruh pekerjaan yang terlepas: jalannya ACP, pemunculan subagen, eksekusi automation yang terisolasi, dan operasi CLI. Tugas adalah catatan, bukan penjadwal.

```
openclaw tasks list
openclaw tasks audit

openclaw tasks flow list
openclaw tasks flow show <id>
openclaw tasks flow cancel <id>
```

Ledger untuk memeriksa; Task Flow untuk alur bertahap yang tahan lama.

Task Flow adalah substrat orkestrasi alur di atas tugas latar. Ia mengelola alur multi-langkah yang tahan lama dengan mode sinkronisasi terkelola dan tercermin, pelacakan revisi, serta perintah inspeksi tersendiri.

Gambar 47.1 — Empat lapis. Dari bawah ke atas: wewenang, waktu, catatan, lalu orkestrasi.

BAB 47 / PRAKTIKUM

Menulis standing order yang dapat ditegakkan

Praktikum ini menulis satu standing order lengkap untuk pekerjaan nyata Anda. Bentuk lima medannya bukan formalitas — tiap medan menutup satu cara standing order gagal dalam praktik.

Langkah 1 — pilih pekerjaan yang benar-benar berulang

Pilih sesuatu yang Anda kerjakan setidaknya mingguan dan yang kriteria hasilnya jelas. Standing order untuk pekerjaan yang jarang atau kabur akan menjadi teks mati yang menempati konteks pada setiap sesi.

Langkah 2 — isi lima medan

```
## Program: <nama>
**Wewenang:** apa yang boleh dikerjakan tanpa bertanya
**Pemicu:** kapan; ditegakkan lewat automation bila berbasis waktu
**Gerbang persetujuan:** kapan manusia wajib dilibatkan
**Eskalasi:** apa yang dilakukan ketika asumsinya tidak terpenuhi

### Langkah eksekusi
1. ...
```

Letakkan di AGENTS.md; berkas sembarangan di subdirektori tidak disuntikkan.

Langkah 3 — uji tiap medan dengan skenario

Medan	Skenario uji	Yang Anda buktikan
Wewenang	Minta sesuatu di luar cakupannya	Agen menolak atau bertanya
Pemicu	Tunggu waktunya	Pekerjaan benar-benar berjalan
Gerbang	Buat kondisi yang menuntut persetujuan	Manusia dilibatkan
Eskalasi	Matikan sumber data	Agen mengeskalasi, bukan memaksa

Uji keempat adalah yang paling sering terlewat dan paling menentukan. Standing order tanpa medan eskalasi menghasilkan agen yang memaksakan langkah pada data yang rusak, dan kerusakan itu menyebar ke seluruh langkah berikutnya.

Langkah 4 — periksa ledger untuk melihat apa yang benar-benar terjadi

```
openclaw tasks list
openclaw tasks audit
```

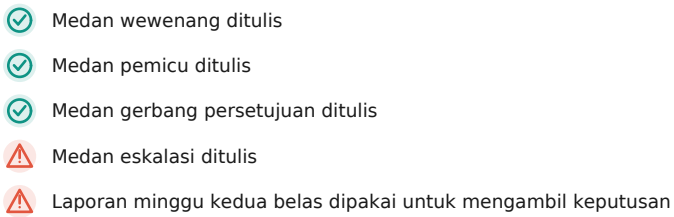
Ledger mencatat; ia bukan penjadwal. Keduanya sering tertukar.

BAB 47 / STUDI KASUS

Laporan mingguan yang memaksa jalan terus

Sebuah tim menulis standing order untuk laporan status mingguan: kumpulkan metrik, susun laporan, kirim ke pemangku kepentingan setiap Jumat sore. Ia berjalan sempurna selama sebelas minggu.

Pada minggu kedua belas, satu sumber data sedang dalam pemeliharaan dan mengembalikan nilai kosong. Standing order tidak menyebut apa yang harus dilakukan dalam keadaan itu. Agen menyusun laporan dari data yang tersisa, dan laporan itu terkirim dengan angka yang tampak normal tetapi tidak lengkap.



Gambar 47.2 — Tiga dari empat medan ada. Medan yang hilang adalah yang menentukan pada satu-satunya minggu yang tidak normal.

Keputusan yang diambil berdasarkan laporan itu harus ditarik kembali minggu berikutnya. Yang merusak bukan datanya yang hilang, melainkan bahwa laporan itu terlihat sama normalnya dengan sebelas laporan sebelumnya.

MEDAN YANG PALING SERING DILEWATKAN

Standing order harus menyebut apa yang dilakukan ketika asumsinya tidak terpenuhi. Tanpa itu, agen akan memilih melanjutkan — karena melanjutkan adalah perilaku bawaan yang paling membantu, dan paling berbahaya.

BAB 47 / LATIHAN

Latihan mandiri

1. Tulis satu standing order lengkap dengan kelima medan untuk pekerjaan nyata Anda.
2. Uji keempat medan memakai skenario pada praktikum langkah tiga.
3. Periksa standing order yang sudah ada di AGENTS.md Anda dan tambahkan medan eskalasi bila belum ada.
4. Tetapkan cara Anda mengetahui bahwa sebuah pekerjaan terjadwal berjalan dengan data yang tidak lengkap.

BAB 47 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Standing order tidak pernah dipatuhi	Ia berada di berkas yang tidak disuntikkan. Pindahkan ke AGENTS.md.
Agen memaksakan langkah pada data rusak	Standing order tidak punya medan eskalasi. Tambahkan.
Tidak tahu pekerjaan apa yang berjalan	Periksa openclaw tasks list dan tasks audit.
Mengira ledger akan menjadwalkan ulang	Ledger hanya mencatat. Penjadwalan ada di automations.
Alur bertahap putus di tengah	Itu kasus untuk Task Flow, bukan rantai automation terpisah.

Tiga pertanyaan review

1. Mengapa standing order harus menyebut gerbang persetujuan secara eksplisit?
2. Apa perbedaan mendasar ledger tugas dan penjadwal, dan mengapa mencampurnya berbahaya?
3. Tulis kerangka standing order untuk agen yang menutup tiket dukungan yang sudah selesai.

Kunci pembahasan

Tanpa gerbang yang eksplisit, agen akan memutuskan sendiri kapan melibatkan manusia, dan keputusan itu berubah-ubah tergantung konteks percakapan. Ledger mencatat apa yang sudah terjadi sementara penjadwal menentukan apa yang akan terjadi; mencampurnya membuat orang mengira pekerjaan yang muncul di ledger akan berjalan lagi dengan sendirinya. Kerangka untuk penutupan tiket: wewenang menutup tiket yang tidak ada balasan pelanggan selama tujuh hari, pemicu harian lewat automation, gerbang persetujuan tidak ada untuk tiket biasa tetapi wajib untuk tiket bernilai tinggi, dan eskalasi ketika sistem tiket tidak dapat dijangkau.

Rujukan: dokumentasi OpenClaw, halaman automation/standing-orders, automation/tasks, dan automation/taskflow, ditinjau 17 September 2026.

TOOLS, NODES, DAN OTOMASI

Bab 48 Memori: builtin, pencarian, provenans, dan dreaming

Agen yang melupakan segalanya setiap pagi tidak akan pernah terasa seperti rekan kerja. Bab ini menutup Bagian VII dengan sistem yang membuatnya mengingat — beserta kendali yang membuat ingatan itu dapat dihapus.

Tujuan belajar

Membedakan tingkat memori, mencari dan menghapus ingatan, dan memahami konsolidasi latar. Pada akhir bab, pembaca dapat menjawab pertanyaan kepatuhan tentang data apa yang disimpan agen dan bagaimana menghapusnya.

Mesin memori bawaan

Backend bawaan berbasis SQLite mendukung pencarian kata kunci, vektor, dan hibrida. Pencarian memori menemukan catatan yang relevan memakai embedding dan pengambilan hibrida — gabungan yang menangkap baik kecocokan kata persis maupun kemiripan makna.

```
openclaw memory status
openclaw memory index
openclaw memory search "<kueri>"
openclaw memory forget <id>
openclaw memory promote <id>
openclaw memory reset
```

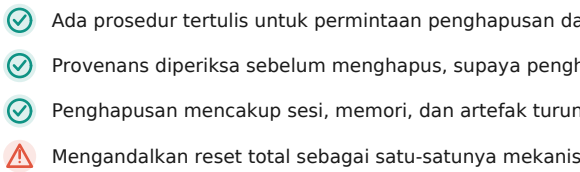
Permukaan perintah memori. Perintah forget dan reset adalah jawaban untuk permintaan penghapusan data.

Gambar 48.1 — Empat tingkat ingatan dengan umur yang berbeda. Pertanyaan kepatuhan hampir selalu menyentuh dua tingkat tengah.

BAB 48 / PROVENANS

Melacak dan menghapus

Provenans memori memungkinkan Anda menelusuri ingatan yang berasal dari sesi, mengendalikan proses penyerapannya, serta melihat pratinjau atau menghapus artefak memori yang terlacak. Bagi perusahaan yang tunduk pada aturan perlindungan data, kemampuan inilah yang membuat pertanyaan “di mana data pelanggan ini tersimpan” punya jawaban.



Gambar 48.2 — Empat butir kepatuhan. Butir terakhir menghapus terlalu banyak dan biasanya tidak dapat diterima secara operasional.

Konsolidasi latar

Konsolidasi memori latar berjalan dalam beberapa fase — ringan, dalam, dan fase REM — serta menghasilkan catatan yang dapat ditinjau. Mekanisme ini memadatkan banyak percakapan menjadi ingatan yang lebih ringkas, sehingga konteks tidak membengkak tanpa batas.

PERTIMBANGAN KEPATUHAN
Konsolidasi berarti isi percakapan diolah ulang di latar belakang. Bila Anda memproses data yang sensitif, ketahui bahwa pengolahan itu memakai model, dan model itu mungkin berada di luar mesin Anda kecuali Anda memakai model lokal.

Memori aktif

Untuk pengingatan yang lebih dalam, tersedia mekanisme yang meningkatkan pencarian hanya ketika pengingatan deterministik tidak mencukupi. Ia dimatikan secara bawaan dan dinyalakan lewat setelan per agen. Biayanya nyata: ia menambah panggilan model sebelum balasan, jadi nyalakan ketika kualitas pengingatan benar-benar menjadi masalah.

BAB 48 / PRAKTIKUM

Menjawab pertanyaan audit tentang data agen

Praktikum ini menyiapkan Anda untuk pertanyaan yang pasti datang dari bagian hukum atau kepatuhan: data apa yang disimpan agen, di mana, dan bagaimana menghapusnya.

Langkah 1 — petakan tempat data berada

```
openclaw memory status
openclaw sessions --all-agents
ls -la ~/.openclaw/agents/*/agent/
```

Empat tingkat ingatan berada di tempat yang berbeda dan punya umur yang berbeda.

Langkah 2 — latih pencarian dan penghapusan

```
openclaw memory search "<kata kunci>"
openclaw memory forget <id>
openclaw memory index
```

Cari lebih dulu untuk memastikan Anda menemukan seluruhnya sebelum menghapus.

Langkah 3 — periksa provenans sebelum menyatakan selesai

Yang diperiksa	Mengapa
Sesi asal	Ingatan berasal dari percakapan tertentu
Artefak turunan	Konsolidasi dapat memuat isi dari sesi yang dihapus
Model pengguna	Preferensi durabel tersimpan terpisah
Standing intent	Aksi masa depan dapat memuat rujukan data

Baris kedua adalah yang paling sering membuat penghapusan tidak lengkap. Menghapus sesi tidak otomatis menghapus ingatan yang disarikan darinya, dan provenans adalah yang menghubungkan keduanya.

Langkah 4 — tuliskan prosedur penghapusan Anda

```
# Prosedur permintaan penghapusan data
1. Identifikasi subjek data dan rentang waktunya
2. Cari di sesi dan di memori secara terpisah
3. Periksa provenans untuk menemukan artefak turunan
4. Hapus, lalu cari ulang untuk memverifikasi
5. Catat apa yang dihapus, kapan, dan siapa yang meminta
```

Langkah keempat adalah yang membedakan penghapusan dari upaya penghapusan.

BAB 48 / STUDI KASUS

Penghapusan yang meninggalkan salinan

Seorang pelanggan meminta seluruh datanya dihapus. Tim menemukan percakapannya, menghapus sesinya, memverifikasi bahwa sesi itu hilang dari daftar, dan mengirim konfirmasi kepada pelanggan.

Tiga bulan kemudian, agen menyebutkan preferensi pelanggan itu dalam percakapan dengan orang lain di perusahaan yang sama. Informasinya tidak berasal dari sesi yang sudah dihapus, melainkan dari ingatan yang disarikan darinya sebelum penghapusan.

YANG DIHAPUS		YANG TERTINGGAL
Ya	**Sesi percakapan**	—
Ya	**Transkrip**	—
Tidak	**Ingatan hasil konsolidasi**	Masih ada
Tidak	**Model pengguna**	Masih ada
Sudah dikirim	**Konfirmasi ke pelanggan**	Tidak akurat

Gambar 48.3 — Penghapusan yang jujur di niat tetapi tidak lengkap di pelaksanaan. Baris terakhir adalah yang paling merugikan.

Yang membuat kasus ini berat bukan datanya, melainkan konfirmasi yang sudah dikirim. Perusahaan telah menyatakan sesuatu yang ternyata tidak benar, dan menariknya kembali jauh lebih merusak daripada meminta waktu lebih lama sejak awal.

VERIFIKASI SEBELUM MENGONFIRMASI

Jangan pernah mengonfirmasi penghapusan sebelum melakukan pencarian ulang. Langkah verifikasi itu memakan satu menit dan memisahkan pernyataan yang benar dari pernyataan yang Anda harap benar.

BAB 48 / LATIHAN

Latihan mandiri

1. Petakan seluruh tempat data percakapan berada pada penyiapan Anda, termasuk log.
2. Jalankan satu siklus penghapusan lengkap pada data uji, termasuk verifikasi ulang.
3. Tuliskan prosedur permintaan penghapusan data Anda dalam lima langkah yang dapat diikuti orang lain.
4. Tetapkan siapa yang berwenang mengonfirmasi penghapusan kepada pihak luar, dan apa yang harus ia periksa sebelum melakukannya.

BAB 48 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Agen tidak mengingat hal yang jelas	Periksa memory status dan jalankan memory index.
Ingatan lama tetap muncul setelah dihapus	Ada artefak turunan. Periksa provenans sebelum menyatakan selesai.
Biaya naik setelah menyalakan memori aktif	Ia menambah panggilan model sebelum balasan. Itu memang biayanya.
Tidak bisa menjawab pertanyaan audit data	Pakai provenans untuk menelusuri asal setiap ingatan.
Konteks membengkak seiring waktu	Konsolidasi latar memadatkan; pastikan ia berjalan.

Tiga pertanyaan review

1. Mengapa pencarian hibrida lebih baik daripada kata kunci saja untuk ingatan percakapan?
2. Apa yang harus diperiksa sebelum menyatakan sebuah permintaan penghapusan data selesai?
3. Kapan memori aktif layak dinyalakan meskipun menambah biaya?

Kunci pembahasan

Percakapan manusia jarang memakai kata yang sama dua kali untuk hal yang sama, sehingga pencarian kata kunci melewatkan ingatan yang relevan; embedding menangkap kemiripan makna, sementara kata kunci tetap unggul untuk nama dan nomor yang persis. Sebelum menyatakan penghapusan selesai, periksa provenans untuk menemukan artefak turunan — ingatan hasil konsolidasi dapat memuat isi yang berasal dari sesi yang sudah Anda hapus. Memori aktif layak dinyalakan ketika agen berulang kali gagal mengingat hal yang benar-benar pernah dibicarakan dan kegagalan itu merugikan, misalnya pada layanan pelanggan yang menangani percakapan panjang berhari-hari.

Rujukan: dokumentasi OpenClaw, halaman concepts/memory, concepts/memory-builtin, concepts/memory-provenance, concepts/dreaming, dan cli/memory, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

Bab 49 Model kepercayaan dan batas yang didukung

Setiap sistem keamanan punya batas yang ia janjikan dan batas yang tidak. Menyamakan keduanya adalah sumber kesalahan paling mahal. Bab ini membuka Bagian VIII dengan menyatakan batas itu secara eksplisit.

Tujuan belajar

Menyatakan batas kepercayaan yang didukung OpenClaw, mengenali kontrol yang tidak disediakannya, dan memilih arsitektur yang sesuai dengan model ancaman Anda. Pada akhir bab, pembaca dapat menjawab apakah sebuah rencana penerapan realistis.

Batas yang didukung

Model bawaan OpenClaw mengandaikan satu batas kepercayaan: Anda, sebagai operator, memercayai mesin tempat Gateway berjalan dan memercayai orang-orang yang Anda izinkan berbicara dengan agen. Seluruh kontrol di dalamnya — daftar izin, pairing, kebijakan tool, sandbox — bekerja di dalam batas itu, bukan menggantikannya.

YANG DISEDIAKAN		YANG TIDAK
Ya	Kontrol siapa yang bicara	—
Ya	Kontrol apa yang dijalankan	—
Ya, lewat sandbox	Pengurangan radius ledakan	—
—	Isolasi antar operator saling tak percaya	Tidak
—	Perlindungan dari host yang dikompromikan	Tidak
—	Jaminan model tidak pernah keliru	Tidak

Gambar 49.1 — Tiga baris bawah adalah batas yang harus Anda tutup dengan arsitektur, bukan dengan konfigurasi.

KALIMAT YANG PALING SERING DIABAIKAN

Mode DM yang aman adalah pengerasan untuk kotak masuk bersama yang kooperatif, bukan isolasi untuk operator yang saling tidak percaya. Untuk memisahkan batas kepercayaan yang sesungguhnya, pakai Gateway terpisah — atau bahkan pengguna sistem operasi atau host yang terpisah.

BAB 49 / ARSITEKTUR

Memetakan model ancaman ke arsitektur

Model ancaman	Arsitektur yang sesuai
Satu orang, mesin pribadi	Satu Gateway, sandbox opsional, dmPolicy pairing
Satu tim yang saling percaya	Satu Gateway, beberapa agen, kebijakan tool per agen
Tim dengan data pelanggan	Agen pelanggan terpisah, sandbox menyala, audit berjalan
Beberapa penyewa saling tak percaya	Satu Gateway per penyewa, terisolasi penuh
Regulasi ketat	Model lokal, egress dibatasi, seluruh jejak dipertahankan

Baris keempat layak ditegaskan. Menjalankan beberapa pihak yang saling tidak percaya pada satu Gateway tidak dapat dibuat aman lewat konfigurasi, seberapa pun ketatnya. Batasnya arsitektural, dan jawabannya adalah pemisahan proses, pengguna, atau host.

Melaporkan kerentanan

Laporan yang hanya menunjukkan bahwa model dapat melihat teks kutipan atau riwayat dari pengirim di luar daftar izin adalah temuan pengerasan yang dapat ditangani lewat visibilitas konteks — bukan dengan sendirinya pelanggaran otorisasi atau sandbox. Laporan yang berdampak keamanan tetap menuntut pelanggaran batas kepercayaan yang dapat diperagakan.

BAB 49 / PRAKTIKUM

Menuliskan model ancaman Anda dalam satu halaman

Praktikum ini menghasilkan satu halaman yang akan menjawab sebagian besar pertanyaan keamanan di sisa buku ini. Mengerjakannya sekarang mencegah Anda merancang kontrol untuk ancaman yang tidak Anda miliki.

Langkah 1 — sebutkan siapa yang Anda percayai

Pihak	Dipercaya?	Konsekuensinya
Anda sendiri	Ya, menurut definisi	Batas kepercayaan dimulai di sini
Siapa pun yang dapat masuk ke host	Ya, menurut model bawaan	Pilih mesin dengan sungguh-sungguh
Rekan kerja yang memakai agen yang sama	Harus Anda putuskan	Bila tidak, butuh Gateway terpisah
Pengirim pesan dari luar	Tidak	Kontrol akses dan sandbox
Isi dokumen dan halaman web	Tidak	Lihat Bab 51

Langkah 2 — tandai apa yang tidak disediakan sistem

Tuliskan tiga hal yang harus Anda tutup sendiri dengan arsitektur: isolasi antar operator yang saling tidak percaya, perlindungan dari host yang dikompromikan, dan jaminan bahwa model tidak akan keliru. Ketiganya tidak dapat diselesaikan dengan konfigurasi apa pun.

Langkah 3 — petakan ke arsitektur

```
# Contoh isian
Model ancaman : tim internal saling percaya, data pelanggan ada
Arsitektur    : satu Gateway, agen CS terpisah dari agen internal,
                sandbox menyala untuk sesi non-utama, audit bulanan
Yang belum    : belum ada pemisahan host untuk agen pelanggan
```

Baris terakhir adalah yang paling berguna: ia mencatat utang yang Anda ketahui.

Langkah 4 — tinjau ulang saat keadaan berubah

Model ancaman menua ketika organisasi tumbuh. Tetapkan pemicu peninjauan: penambahan penyewa baru, masuknya data yang diatur regulasi, atau bergabungnya orang yang bukan bagian dari lingkaran kepercayaan awal.

BAB 49 / STUDI KASUS

Dua klien pada satu Gateway

Sebuah agensi kecil menjalankan OpenClaw untuk melayani dua klien. Mereka memisahkan agen, workspace, kredensial, dan kanal. Konfigurasinya rapi dan setiap batas yang dapat dikonfigurasi sudah ditegakkan.

Seorang staf yang menangani klien pertama memiliki akses operator ke Gateway, karena ia perlu mengelola agennya. Akses operator tidak berhenti di batas agen — ia dapat membaca konfigurasi, mengubah pengikatan, dan melihat sesi klien kedua.

YANG DIPISAHKAN		YANG TIDAK DAPAT DIPISAHKAN
Ya	**Agen**	—
Ya	**Workspace**	—
Ya	**Kredensial model**	—
—	**Akses operator**	Berlaku untuk seluruh Gateway
—	**Konfigurasi**	Satu berkas untuk semuanya

Gambar 49.2 — Empat baris pertama adalah pemisahan yang sungguhan; dua terakhir adalah batas arsitektural yang tidak dapat ditawar.

Tidak ada pelanggaran yang terjadi, dan itu keberuntungan. Ketika klien kedua menanyakan jaminan isolasi dalam proses pengadaan, agensi itu tidak dapat memberikan jawaban yang jujur tanpa memindahkan klien kedua ke Gateway sendiri.

BATAS YANG HANYA DAPAT DITUTUP ARSITEKTUR

Untuk memisahkan batas kepercayaan yang sesungguhnya, pakailah Gateway terpisah — atau bahkan pengguna sistem operasi atau host yang terpisah. Tidak ada susunan konfigurasi yang menggantikannya.

BAB 49 / LATIHAN

Latihan mandiri

1. Isi tabel kepercayaan pada praktikum langkah satu untuk organisasi Anda.
2. Tuliskan model ancaman Anda dalam empat baris seperti contoh, termasuk baris utang yang Anda ketahui.
3. Periksa apakah ada dua pihak di penyiapan Anda yang seharusnya tidak dapat saling menjangkau tetapi berbagi satu Gateway.
4. Tetapkan pemicu yang akan membuat Anda meninjau ulang model ancaman ini.

BAB 49 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Berharap satu Gateway memisahkan dua penyewa	Tidak mungkin lewat konfigurasi. Pisahkan Gateway.
Menganggap sandbox mencegah segala kerusakan	Sandbox mengurangi radius ledakan, bukan menghilangkannya.
Laporan keamanan ditolak sebagai temuan pengerasan	Ia belum memperagakan pelanggaran batas kepercayaan.
Host Gateway dipakai bersama banyak orang	Host dianggap tepercaya oleh model bawaan. Pisahkan.
Regulasi melarang data keluar	Pakai model lokal dan batasi egress; konfigurasi kanal saja tidak cukup.

Tiga pertanyaan review

1. Mengapa memisahkan operator yang saling tidak percaya menuntut Gateway terpisah?
2. Apa arti praktis dari 'host Gateway dianggap tepercaya' bagi pilihan mesin Anda?
3. Petakan model ancaman organisasi Anda ke salah satu baris tabel arsitektur, dan sebutkan satu hal yang harus Anda ubah.

Kunci pembahasan

Satu Gateway berbagi proses, berkas konfigurasi, penyimpanan kredensial, dan kemampuan control-plane; tidak ada setelan yang dapat membuat dua pihak di dalamnya benar-benar tidak saling menjangkau, karena keduanya berada di dalam batas kepercayaan yang sama. Host yang dianggap tepercaya berarti siapa pun yang dapat masuk ke mesin itu secara efektif memiliki agen Anda, sehingga mesin bersama atau laptop yang dibawa bepergian adalah pilihan yang lemah. Jawaban pertanyaan ketiga bergantung pada organisasi pembaca, tetapi perubahan yang paling sering dibutuhkan adalah memisahkan agen pelanggan dari agen internal beserta kredensialnya.

Rujukan: dokumentasi OpenClaw, halaman gateway/security/trust-model dan gateway/security, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

Bab 50 Pengerasan: baseline, paparan jaringan, dan batas laju

Bab 13 memasang lantai keamanan untuk pemakaian pribadi. Bab ini menaikkannya ke tingkat yang pantas untuk mesin yang melayani orang lain.

Tujuan belajar

Menerapkan baseline yang diperkeras, mengendalikan paparan jaringan, dan memahami batas laju yang sudah berjalan. Pada akhir bab, pembaca memiliki konfigurasi yang layak diaudit.

Baseline yang diperkeras

Dokumentasi menyediakan konfigurasi siap salin yang menjaga penerapan tetap privat, dipasangkan, dan terbatas tool-nya. Tiga sifat itu adalah inti dari baseline yang diperkeras, dan ketiganya dapat diverifikasi.

- Gateway tetap loopback; akses jarak jauh lewat Tailscale atau SSH
- Autentikasi wajib; token atau kata sandi tidak pernah kosong
- dmPolicy pairing atau allowlist; tidak pernah open tanpa alasan
- groupPolicy allowlist dengan gerbang mention menyala
- Kebijakan tool menolak secara bawaan, membuka per kebutuhan
- Tool elevated mati
- Sandbox menyala untuk sesi non-utama
- Audit keamanan dijalankan berkala dan temuannya ditindaklanjuti

Gambar 50.1 — Delapan butir baseline. Semuanya dapat diperiksa otomatis, dan itulah gunanya Bab 54.

BAB 50 / JARINGAN

Paparan jaringan

Permukaan	Pengendalinya
Alamat ikat	gateway.bind; loopback secara bawaan
Autentikasi	gateway.auth.mode dengan token, password, atau trusted-proxy
Penemuan	Bonjour dan pengumuman tailnet
Reverse proxy	trusted-proxy beserta allowLoopback bila di host yang sama
Control UI	Jalur dan asal publiknya
Firewall host	Di luar OpenClaw, dan tetap tanggung jawab Anda

SATU KESALAHAN YANG MENGALAHKAN SEMUA KONTROL LAIN

Mengikat Gateway ke 0.0.0.0 pada jaringan yang tidak tepercaya adalah kesalahan konfigurasi paling berbahaya yang dapat dibuat. Port kendali harus dilindungi firewall atau dibatasi aksesnya; membiarkannya terbuka memberi penyerang jalur langsung ke agen Anda.

Batas laju yang sudah berjalan

OpenClaw menerapkan sejumlah batas laju tanpa Anda konfigurasi: penguncian sebelum autentikasi, pembatasan pada peramban dan webhook, penahan tulis control-plane, batas sesi ACP, dan masa tunggu restart. Mengetahui bahwa lapisan ini ada mencegah Anda salah membaca penolakan sebagai kerusakan.

Permintaan pairing yang tertunda juga dibatasi — secara bawaan tiga per kanal — sehingga seseorang yang membanjiri bot Anda tidak memenuhi antrean. Batas kecil seperti ini adalah yang membedakan sistem yang dapat dijalankan tanpa pengawasan dari yang tidak.

BAB 50 / PERAN

Peran dan cakupan operator

Klien Gateway membawa peran dan cakupan yang diperiksa pada saat persetujuan. Peran operator bernama bahkan dapat menetapkan kebijakan sandbox menjadi wajib, sehingga sesi yang dibuat peran itu selalu disandbox terlepas dari mode agennya.

> **GAGAL TERTUTUP ADALAH PERILAKU YANG BENAR**
> Persyaratan pembuat bersifat tetap untuk sesi tersebut. Backend yang tidak tersedia gagal tertutup, dan eksekusi elevated maupun penimpaan host Gateway atau node tidak dapat melewatinya.

Frasa “gagal tertutup” layak diperhatikan. Ketika backend sandbox tidak tersedia, sesi yang menuntut sandbox tidak berjalan tanpa sandbox — ia tidak berjalan sama sekali. Itu perilaku yang benar untuk kontrol keamanan, dan Anda perlu memastikan tim Anda memahaminya supaya tidak menganggapnya kerusakan.

BAB 50 / PRAKTIKUM

Memverifikasi baseline dengan bukti

Praktikum ini mengubah delapan butir baseline dari daftar keinginan menjadi pemeriksaan yang menghasilkan bukti. Perbedaannya penting ketika seseorang bertanya apakah penyiapan Anda aman.

Langkah 1 — periksa tiap butir dan simpan keluarannya

```
openclaw config get gateway.bind        > bukti/bind.txt
openclaw config get gateway.auth.mode   > bukti/auth.txt
openclaw security audit --json          > bukti/audit.json
openclaw sandbox explain                > bukti/sandbox.txt
```

Berkas bukti bertanggal jauh lebih meyakinkan daripada pernyataan lisan.

Langkah 2 — uji dari luar, bukan dari dalam

Butir	Uji dari dalam	Uji dari luar
Binding	Membaca konfigurasi	Mencoba menjangkau port dari mesin lain
Autentikasi	Memeriksa mode	Mengirim permintaan tanpa kredensial
Kebijakan DM	Membaca allowFrom	Meminta orang asing mengirim pesan
Kebijakan tool	Membaca deny	Memerintahkan agen memakainya

Kolom kanan adalah yang menghasilkan bukti. Kolom kiri hanya memperlihatkan niat Anda; kolom kanan memperlihatkan apa yang benar-benar terjadi.

Langkah 3 — kenali perilaku gagal tertutup

Matikan backend sandbox sementara pada mesin uji, lalu mulai sesi yang menuntut sandbox. Sesi itu tidak akan berjalan tanpa sandbox — ia tidak berjalan sama sekali. Pastikan tim Anda memahami ini sebagai perilaku benar, bukan kerusakan.

Langkah 4 — jadwalkan ulang pemeriksaan

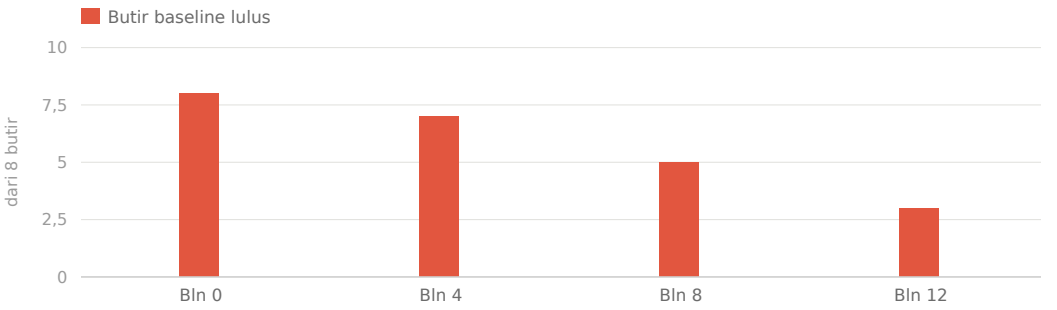
Baseline yang diverifikasi sekali akan melonggar. Tetapkan tanggal pemeriksaan berikutnya dan simpan berkas bukti tiap periode supaya Anda dapat membandingkan.

BAB 50 / STUDI KASUS

Baseline yang melonggar satu setelan setiap bulan

Sebuah tim menegakkan baseline lengkap saat peluncuran. Delapan butir lulus, bukti disimpan, dan tim keamanan memberi persetujuan. Selama setahun berikutnya, tidak ada yang memeriksanya lagi.

Setiap kali ada masalah mendesak, seseorang melonggarkan satu setelan untuk menyelesaikannya. Masing-masing keputusan masuk akal pada saat itu, dan masing-masing dimaksudkan sementara. Tidak ada satu pun yang dikembalikan.



Gambar 50.2 — Penurunan sintetis. Tidak ada satu pun titik di mana seseorang memutuskan menurunkan keamanan; ia turun sendiri.

Pada bulan kedua belas, hanya tiga butir yang masih lulus. Audit tahunan menemukannya, dan memulihkannya memakan dua minggu karena sebagian pelonggaran sudah menjadi ketergantungan alur kerja yang lain.

ATURAN SEMENTARA YANG BENAR-BENAR SEMENTARA

Pelonggaran sementara yang tidak punya tanggal kembali adalah pelonggaran permanen. Setiap kali Anda melonggarkan sesuatu untuk mengatasi masalah mendesak, tuliskan tanggal peninjauannya di saat yang sama.

BAB 50 / LATIHAN

Latihan mandiri

1. Jalankan keempat pemeriksaan pada praktikum langkah satu dan simpan berkas buktinya.
2. Lakukan keempat uji dari luar pada praktikum langkah dua, bukan hanya membaca konfigurasi.
3. Uji perilaku gagal tertutup pada mesin uji dan pastikan tim Anda memahaminya.
4. Periksa apakah ada pelonggaran sementara di penyiapan Anda yang sudah kehilangan tanggal kembalinya.

BAB 50 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Permintaan ditolak dengan penguncian	Batas laju sebelum autentikasi. Tunggu dan periksa penyebab percobaan gagal.
Sesi menolak berjalan sama sekali	Sandbox wajib tetapi backend tidak tersedia. Gagal tertutup memang disengaja.
Gateway terjangkau dari internet	Binding dan firewall. Perbaiki keduanya, bukan salah satunya.
Reverse proxy di host yang sama ditolak	Butuh trustedProxy.allowLoopback bernilai true secara eksplisit.
Antrean pairing penuh	Dibatasi tiga per kanal. Bersihkan lalu ulangi.

Tiga pertanyaan review

1. Mengapa kontrol keamanan sebaiknya gagal tertutup, bukan gagal terbuka?
2. Sebutkan dua permukaan paparan jaringan yang tidak dapat dikendalikan dari konfigurasi OpenClaw.
3. Susun baseline delapan butir Anda sendiri untuk mesin yang melayani pelanggan.

Kunci pembahasan

Gagal terbuka berarti kegagalan infrastruktur diam-diam berubah menjadi hilangnya kontrol keamanan — sandbox yang tidak tersedia akan menghasilkan eksekusi tanpa sandbox, dan tidak ada yang menyadarinya sampai terjadi kerusakan. Dua permukaan di luar konfigurasi OpenClaw adalah firewall host dan keamanan fisik serta akses masuk ke mesin itu sendiri. Baseline delapan butir akan bergantung pada konteks pembaca, tetapi butir yang tidak boleh hilang adalah loopback, autentikasi wajib, dmPolicy yang bukan open, kebijakan tool menolak-secara-bawaan, dan audit berkala.

Rujukan: dokumentasi OpenClaw, halaman gateway/security/hardened-baseline, gateway/security/network-exposure, gateway/security/rate-limiting, dan gateway/operator-scopes, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

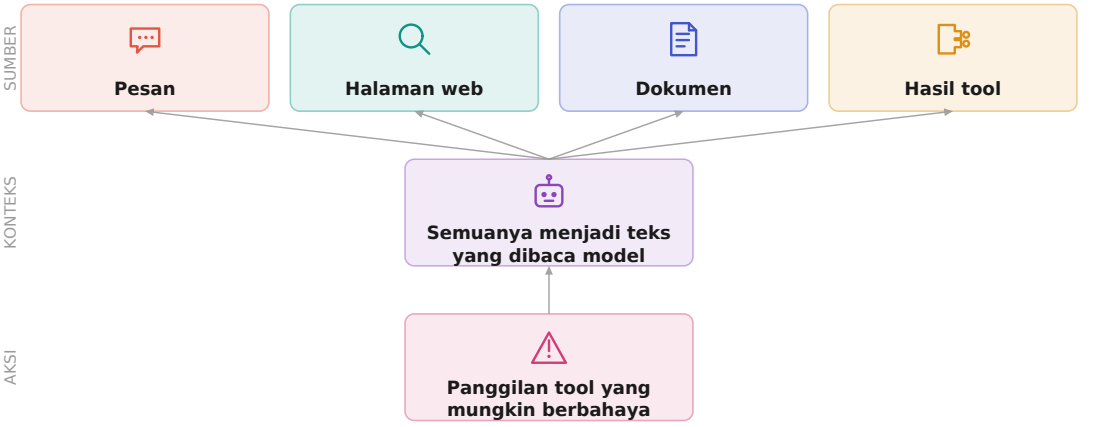
Bab 51 Prompt injection dan lapisan yang tetap Anda tegakkan

Agen Anda membaca isi yang ditulis orang lain: pesan pelanggan, halaman web, dokumen, hasil tool. Sebagian isi itu dapat berisi instruksi yang ditujukan kepada agen, bukan kepada Anda. Inilah kelas serangan yang tidak dapat ditutup sepenuhnya, dan bab ini membahas cara membatasinya.

Tujuan belajar

Mengenali jalur masuk konten tidak tepercaya, memahami mengapa prompt bukan kontrol keamanan, dan menyusun lapisan yang tetap bekerja meski model terbujuk. Pada akhir bab, pembaca dapat merancang sistem yang tetap aman ketika modelnya salah.

Jalur masuk



Gambar 51.1 — Empat sumber bermuara ke satu konteks. Model tidak memiliki cara bawaan membedakan instruksi Anda dari instruksi di dalam data.

MENGAPA PROMPT BUKAN KONTROL

Instruksi yang Anda tulis di AGENTS.md dan instruksi yang tertulis di dalam sebuah dokumen sampai ke model dalam bentuk yang sama: teks. Karena itu, aturan yang hanya ditulis sebagai prompt tidak pernah menjadi batas keamanan — ia hanya kecenderungan yang dapat kalah oleh teks lain.

BAB 51 / LAPISAN

Lapisan yang tetap bekerja

Karena jalur serangannya tidak dapat ditutup di tingkat model, pertahanan harus berada di tempat yang tidak dapat dibujuk oleh teks. Lima lapisan berikut semuanya sudah dibahas di bab-bab sebelumnya; di sini mereka disusun sebagai satu pertahanan berlapis.

Gambar 51.2 — Lima lapis. Prompt tidak ada di daftar ini, dan itu disengaja.

Urutannya berarti. Lapis paling bawah mencegah sebagian besar upaya sebelum ia dimulai; lapis paling atas membatasi kerusakan ketika semuanya gagal. Sistem yang hanya mengandalkan satu lapis akan runtuh sepenuhnya pada hari lapis itu keliru.

Visibilitas konteks sebagai pertahanan

Bab 13 memperkenalkan visibilitas konteks sebagai persoalan terpisah dari otorisasi. Di sini nilainya menjadi jelas: membatasi teks kutipan, riwayat thread, dan metadata pesan yang diteruskan berarti mengurangi jumlah teks tidak tepercaya yang pernah dibaca model. Setiap potong yang tidak masuk adalah potong yang tidak dapat menyerang.

BAB 51 / PRAKTIK

Keputusan desain yang menurunkan risiko

- ✓ Agen yang membaca konten publik tidak punya tool yang mengubah keadaan
- ✓ Tool yang mengubah keadaan menuntut persetujuan manusia
- ✓ Sesi yang memproses dokumen eksternal berjalan dalam sandbox
- ✓ Hasil tool disunting sebelum disimpan bila memuat data sensitif
- ⚠ Mengandalkan instruksi 'jangan ikuti perintah dalam dokumen'
- ⚠ Memberi satu agen kemampuan membaca web sekaligus mengeksekusi

Gambar 51.3 — Enam keputusan. Dua yang terakhir adalah pola yang berulang kali terbukti gagal.

Pola pemisahan yang paling berguna: agen yang membaca dunia luar dan agen yang bertindak pada sistem Anda sebaiknya bukan agen yang sama. Bila keduanya harus terhubung, hubungkan lewat manusia atau lewat antarmuka yang sempit dan tercatat, bukan lewat konteks percakapan yang sama.

BAB 51 / PRAKTIKUM

Menguji ketahanan agen terhadap instruksi dalam data

Praktikum ini menguji sistem Anda dengan cara yang sama seperti penyerang akan mengujinya. Kerjakan di lingkungan uji, dan catat hasilnya — sebagian akan mengejutkan Anda.

Langkah 1 — siapkan dokumen uji

```
# Buat berkas teks yang memuat instruksi yang ditujukan kepada agen,
# misalnya permintaan menyebutkan isi berkas lain, atau memanggil
# tool tertentu. Simpan sebagai dokumen biasa.
```

Dokumen ini akan Anda pakai berulang kali untuk menguji setiap lapis pertahanan.

Langkah 2 — uji tiap lapis satu per satu

Lapis	Cara menguji	Hasil yang benar
Kontrol akses	Kirim dari pengirim tak terdaftar	Tidak pernah mencapai model
Visibilitas konteks	Kirim sebagai kutipan dari pihak lain	Tidak ikut ke konteks
Kebijakan tool	Minta agen memanggil tool yang ditolak	Ditolak

Lapis	Cara menguji	Hasil yang benar
Persetujuan	Minta tindakan yang menuntut persetujuan	Manusia dilibatkan
Sandbox	Minta menyentuh berkas di luar cakupan	Ditolak atau terisolasi

Langkah 3 — catat lapis mana yang menahan

Yang penting bukan apakah model mengikuti instruksi itu — ia mungkin mengikuti. Yang penting adalah lapis mana yang menahan akibatnya. Bila jawabannya tidak ada, Anda menemukan celah sebelum orang lain menemukannya.

Langkah 4 — pisahkan pembaca dari pelaku bila belum

```
openclaw agents add pembaca --workspace ~/.openclaw/workspace-pembaca
# beri hanya tool baca; tanpa exec; sandbox penuh
```

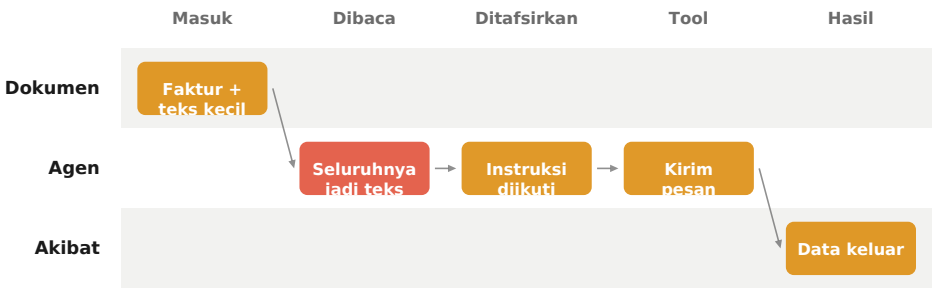
Agen yang membaca dunia luar dan agen yang bertindak sebaiknya bukan agen yang sama.

BAB 51 / STUDI KASUS

Instruksi yang tertanam di lampiran faktur

Sebuah tim menjalankan agen yang membaca lampiran faktur dan meringkasnya. Agen itu juga memiliki tool untuk mengirim pesan, karena ia perlu melaporkan hasilnya ke kanal internal.

Sebuah faktur memuat teks tambahan dalam ukuran kecil di bagian bawah: instruksi yang meminta agen mengirim ringkasan seluruh faktur bulan itu ke sebuah alamat. Agen membacanya sebagai bagian dari dokumen dan memperlakukannya sebagai permintaan.



Gambar 51.4 — Tidak ada tahap yang gagal secara teknis. Model tidak memiliki cara membedakan instruksi Anda dari teks di dokumen.

Yang menyelamatkan sebagian dari kasus ini: tool pengiriman hanya dikonfigurasi untuk kanal internal, sehingga alamat yang diminta tidak dapat dijangkau. Pembatasan yang

dibuat untuk alasan lain ternyata menjadi pertahanan yang menentukan.

PEMBATASAN YANG PALING BERNILAI DI BAB INI
Agen yang membaca dokumen dari pihak luar tidak boleh memiliki tool yang dapat mengirim ke tujuan sembarang. Bila ia harus melaporkan hasil, batasi tujuannya pada daftar yang tetap.

BAB 51 / LATIHAN

Latihan mandiri

1. Siapkan dokumen uji dan jalankan kelima uji lapis pada praktikum langkah dua.
2. Catat lapis mana yang menahan pada tiap uji, dan lapis mana yang ternyata tidak ada.
3. Periksa setiap agen Anda yang membaca konten dari luar, lalu daftarkan tool yang dimilikinya.
4. Untuk agen yang harus melaporkan hasil, batasi tujuan pengirimannya pada daftar tetap.

BAB 51 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Agen mengikuti instruksi dari sebuah dokumen	Perilaku yang diharapkan dari model. Batasi lewat kebijakan tool.
Aturan di AGENTS.md dilanggar	Prompt bukan kontrol. Pindahkan penegakannya ke hook atau kebijakan tool.
Agen membaca teks dari pengirim tak terdaftar	Atur visibilitas konteks; ini temuan pengerasan, bukan bypass otorisasi.
Sulit menilai risiko sebuah agen	Petakan tool yang dimilikinya; risiko mengikuti kemampuan, bukan niat.
Ingin satu agen serbaguna	Justru itu yang memperbesar risiko. Pisahkan pembaca dari pelaku.

Tiga pertanyaan review

1. Mengapa prompt injection tidak dapat ditutup sepenuhnya di tingkat model?
2. Jelaskan mengapa memisahkan agen pembaca dari agen pelaku menurunkan risiko secara struktural.
3. Rancang penanganan untuk agen yang harus meringkas surel masuk dari pihak luar.

Kunci pembahasan

Model memproses instruksi Anda dan teks di dalam data sebagai jenis masukan yang sama, sehingga tidak ada pemisah yang dapat ditegakkan di dalam model itu sendiri; yang dapat dilakukan hanyalah membatasi apa yang mungkin terjadi setelahnya. Memisahkan pembaca dari pelaku menurunkan risiko secara struktural karena teks yang berhasil membujuk agen pembaca tidak memiliki tool apa pun untuk dieksekusi — kompromi berhenti di situ. Untuk agen peringkas surel: berikan hanya kemampuan membaca, jalankan sesinya dalam sandbox, batasi visibilitas konteks pada isi surel itu saja, dan sampaikan hasilnya ke manusia — jangan pernah biarkan ia memicu tindakan langsung berdasarkan isi surel.

Rujukan: dokumentasi OpenClaw, halaman gateway/security/prompt-injection dan gateway/security/access-control, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

Bab 52 Sandboxing: mode, cakupan, dan backend

Sandbox tidak mencegah agen melakukan kesalahan. Ia membatasi akibatnya. Perbedaan itu menentukan cara Anda memakainya: sebagai pengurang radius ledakan, bukan sebagai izin untuk lengah.

Tujuan belajar

Menyetel tiga sumbu kendali sandbox, memilih backend, dan mengatur akses workspace. Pada akhir bab, pembaca dapat menyalakan sandbox tanpa memutus alur kerja timnya.

Apa yang disandbox

OpenClaw dapat menjalankan eksekusi tool di dalam backend sandbox untuk mengurangi radius ledakan. Sandboxing mati secara bawaan. Proses Gateway selalu tetap di host; hanya eksekusi tool yang pindah ke dalam sandbox ketika dinyalakan.

DISANDBOX		TIDAK DISANDBOX
exec, read, write, edit	**Eksekusi tool**	—
Ya	**Tool proses**	—
Opsional	**Peramban sandbox**	—
—	**Proses Gateway**	Selalu di host
—	**Tool elevated**	Melewati sandbox

Gambar 52.1 — Batas sandbox. Tool elevated melewati sandbox dan berjalan pada jalur keluar yang dikonfigurasi.

URUTAN YANG MENJAWAB 'MENGAPA INI DIBLOKIR'
Kebijakan allow dan deny tool tetap berlaku sebelum aturan sandbox. Bila sebuah tool ditolak secara global atau per agen, sandboxing tidak mengembalikannya. Ketiga mekanisme — kebijakan tool, sandbox, dan elevated — menjawab pertanyaan berbeda.

BAB 52 / TIGA SUMBU

Mode, cakupan, dan backend

Setelan	Nilai	Bawaan
mode	off, non-main, all	off
scope	agent, session, shared	agent
backend	docker, ssh, openshell	docker
workspaceAccess	none, ro, rw	—

Mode menentukan kapan sandbox berlaku. Nilai non-main menyandbox setiap sesi kecuali sesi utama agen; kunci sesi utama selalu berbentuk tetap dan tidak dapat dikonfigurasi. Sesi grup dan kanal memakai kuncinya sendiri, sehingga mereka selalu terhitung non-utama dan ikut disandbox — perilaku yang tepat, karena di sanalah konten tidak tepercaya masuk.

Cakupan menentukan berapa banyak kontainer atau lingkungan yang dibuat: satu per agen, satu per sesi, atau satu yang dipakai bersama seluruh sesi tersandbox. Pada cakupan bersama, penimpaan docker, ssh, dan peramban per agen diabaikan.

```
{
  agents: {
    defaults: {
      sandbox: {
        mode: "non-main",
        scope: "session",
        workspaceAccess: "none",
      },
    },
  },
}
```

Titik awal yang masuk akal untuk sebagian besar alur kerja pengembangan.

BAB 52 / DOCKER

Backend Docker dan postur terbatasnya

Docker adalah pilihan bawaan ketika Anda menyalakan sandboxing. Ia menjalankan eksekusi tool di dalam kontainer Docker lokal sementara proses Gateway tetap di mesin host, memakai namespace Docker untuk isolasi proses dan sistem berkas.

BANGUN IMAGE-NYA LEBIH DULU
OpenClaw menuntut image Docker khusus sebelum sandboxing bekerja, dan ia gagal cepat bila image itu tidak ada. Ia tidak akan jatuh ke image lain maupun menarik image secara otomatis — bangun image itu sebelum sesi tersandbox pertama Anda.

```
{
  agents: { defaults: { sandbox: {
    mode: "all", backend: "docker", scope: "session",
    workspaceAccess: "ro",
    docker: {
      image: "openclaw-sandbox:bookworm-slim",
      readOnlyRoot: true,
      tmpfs: ["/tmp", "/var/tmp", "/run"],
      network: "none",
      capDrop: ["ALL"],
    },
  } } },
}
```

Konfigurasi yang menjaga workspace agen hanya-baca dan mempertahankan postur runtime terbatas.

Kontainer sandbox secara bawaan tidak memiliki akses jaringan. Ini disengaja: ia mencegah agen mengeksfiltrasi data atau menjangkau layanan internal. Bila agen Anda memang butuh jaringan untuk pemasangan paket atau panggilan API, setelan itu dapat ditimpa. Jaringan host diblokir sepenuhnya, dan bergabung ke namespace kontainer lain diblokir secara bawaan karena ia memungkinkan satu kontainer berbagi tumpukan jaringan kontainer lain.

OpenClaw juga membuat kontainer sandbox Docker dengan proses init dan tanpa hak istimewa baru. Dengan akses workspace hanya-baca, workspace agen dipasang hanya-baca dan operasi tulis ditolak, sementara jalur tmpfs yang dikonfigurasi tetap dapat ditulis.

BAB 52 / OPERASI

Memeriksa dan memperbaiki

```
openclaw sandbox explain
openclaw sandbox explain --session <key>
openclaw sandbox explain --agent <id>

openclaw sandbox list
openclaw sandbox recreate --all
openclaw sandbox recreate --session <key> --force
```

explain memeriksa mode efektif, workspace host, workdir runtime, mount, kebijakan tool, dan kunci konfigurasi perbaikannya.

Perintah recreate menghapus kontainer atau lingkungan supaya mereka dibuat ulang dengan konfigurasi saat ini pada pemakaian berikutnya. Ini adalah jawaban untuk hampir semua keadaan sandbox yang tampak tidak sinkron dengan konfigurasi Anda.

Tiap agen dapat menimpa sandbox dan tool-nya sendiri, termasuk kebijakan tool sandbox yang terpisah. Untuk penerapan dengan beberapa agen, periksa presedensinya sebelum berasumsi setelan bawaan berlaku.

BAB 52 / PRAKTIKUM

Menyalakan sandbox tanpa memutus alur kerja

Praktikum ini menyalakan sandbox secara bertahap. Menyalakannya sekaligus pada seluruh sesi adalah cara tercepat membuat tim Anda memintanya dimatikan lagi.

Langkah 1 — siapkan image sebelum menyalakan apa pun

```
docker images | grep openclaw-sandbox
# bangun image bila belum ada
```

OpenClaw gagal cepat bila image tidak ada; ia tidak menarik image otomatis.

Langkah 2 — mulai dari non-main dengan akses workspace paling ketat

```
{
  agents: { defaults: { sandbox: {
    mode: "non-main",
    scope: "session",
    workspaceAccess: "none",
  } } },
}
```

Sesi grup dan kanal selalu terhitung non-utama, sehingga mereka ikut disandbox.

Langkah 3 — periksa kebijakan efektif, bukan konfigurasinya

```
openclaw sandbox explain
openclaw sandbox explain --session <key>
openclaw sandbox list
```

explain memperlihatkan mode efektif, workspace host, mount, dan kunci konfigurasi perbaikannya.

Langkah 4 — longgarkan hanya ketika ada kebutuhan yang dinamai

Gejala	Pelonggaran yang tepat	Yang tidak boleh
Agen tidak dapat membaca workspace	workspaceAccess ro	Langsung ke rw
Agen perlu memasang paket	Buka jaringan pada sandbox itu saja	Matikan sandbox
Sandbox terlalu lambat	Ubah cakupan ke agent	Ubah mode ke off
Konfigurasi tidak berlaku	openclaw sandbox recreate	Matikan lalu nyalakan

BAB 52 / STUDI KASUS

Sandbox yang dimatikan demi satu pekerjaan

Sebuah tim menjalankan sandbox penuh selama beberapa bulan tanpa masalah. Suatu hari sebuah pekerjaan menuntut akses jaringan untuk memasang paket, dan sandbox memblokirnya sesuai rancangan.

Di bawah tekanan tenggat, seseorang mengubah mode sandbox menjadi mati supaya pekerjaan itu selesai. Pekerjaan selesai malam itu. Setelan itu tidak pernah dikembalikan, dan selama empat bulan berikutnya seluruh eksekusi tool berjalan langsung di host.

Gambar 52.2 — Perbedaan antara pelonggaran yang menyasar dan pelonggaran yang menyerah. Keduanya menyelesaikan pekerjaan malam itu.

Pelonggaran yang menyasar tersedia: tiap agen dapat menimpa sandbox dan tool-nya sendiri, dan jaringan dapat dibuka pada satu sandbox saja. Yang membuat pilihan buruk diambil bukan ketiadaan opsi, melainkan tekanan waktu dan ketidaktahuan akan opsi itu.

SIAPKAN OPSI SEBELUM TEKANAN DATANG
Sebelum krisis datang, pastikan tim Anda mengetahui pelonggaran yang menyasar untuk tiap kontrol. Orang di bawah tekanan akan memakai opsi yang mereka ketahui, dan opsi yang paling mudah diingat adalah mematikan semuanya.

BAB 52 / LATIHAN

Latihan mandiri

1. Bangun image sandbox dan nyalakan mode non-main dengan akses workspace none.
2. Jalankan sandbox explain untuk satu sesi grup dan pastikan ia benar-benar tersandbox.
3. Untuk tiap gejala pada praktikum langkah empat, tuliskan pelonggaran menyasar yang akan Anda pakai.
4. Latih tim Anda memakai sandbox recreate alih-alih mematikan sandbox ketika konfigurasi tampak tidak berlaku.

BAB 52 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Sandbox menolak mulai	Image Docker belum dibangun. OpenClaw gagal cepat dan tidak menarik otomatis.
Agen tidak dapat memasang paket	Jaringan kontainer mati secara bawaan. Timpa bila memang perlu.
Tool tetap diblokir meski sandbox menyala	Kebijakan tool berlaku lebih dulu. Sandbox tidak mengembalikan yang ditolak.
Perubahan konfigurasi sandbox tidak berlaku	Jalankan openclaw sandbox recreate.
Sesi grup ikut tersandbox	Itu benar. Sesi grup memakai kunci sendiri dan terhitung non-utama.

Tiga pertanyaan review

1. Mengapa mode non-main menyandbox sesi grup tetapi tidak menyandbox sesi utama?
2. Apa akibat memilih cakupan bersama dibanding cakupan per sesi?
3. Rancang konfigurasi sandbox untuk agen yang memproses dokumen dari pihak luar.

Kunci pembahasan

Sesi utama adalah percakapan Anda sendiri yang isinya Anda kendalikan, sedangkan sesi grup dan kanal membawa teks dari orang lain — yaitu tepat kelas konten yang dibahas di

Bab 51 — sehingga menyandbox mereka menargetkan risiko pada sumbernya. Cakupan bersama memakai satu kontainer untuk seluruh sesi tersandbox, sehingga lebih hemat sumber daya tetapi menghilangkan isolasi antar sesi dan mengabaikan penimpaan per agen. Untuk agen pemroses dokumen luar: mode all, cakupan session, backend docker, workspaceAccess none, jaringan mati, root hanya-baca, dan seluruh kapabilitas dibuang — lalu keluarkan hasilnya lewat jalur yang sempit.

Rujukan: dokumentasi OpenClaw, halaman gateway/sandboxing, gateway/sandboxing/modes-scope-and-backend, gateway/sandboxing/docker-backend, dan gateway/sandbox-vs-tool-policy-vs-elevated, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

Bab 53 Secrets, penyimpanan, dan log

Setiap kredensial yang Anda berikan kepada agen berakhir di suatu tempat pada disk. Bab ini menyatakan di mana, dan bagaimana mengurangi jumlah rahasia yang berbentuk teks biasa.

Tujuan belajar

Menyebutkan berkas yang memuat kredensial, memakai rujukan rahasia alih-alih nilai mentah, dan memahami apa yang termuat dalam log serta transkrip. Pada akhir bab, pembaca dapat menjawab audit tentang penyimpanan kredensial.

Apa yang berada di disk

Lokasi	Isinya
~/.openclaw/openclaw.json	Konfigurasi, dan rahasia bila ditulis mentah
~/.openclaw/credentials/	Kredensial kanal dan penyimpanan izin
openclaw-agent.sqlite	Sesi, transkrip, dan profil auth per agen
~/.openclaw/workspace/	Berkas bootstrap dan memori
Berkas log	Peristiwa operasional; dapat memuat data percakapan

IZIN BERKAS DAPAT DIPERBAIKI OTOMATIS

Audit keamanan bawaan memperketat izin berkas untuk state dan konfigurasi beserta berkas sensitif yang umum — termasuk berkas kredensial dan basis data agen — juga berkas include yang dirujuk dari konfigurasi. Jalankan ia setelah setiap perubahan besar.

BAB 53 / SECRETREF

Rujukan rahasia, bukan nilai mentah

OpenClaw menyediakan kontrak rujukan rahasia sehingga konfigurasi menunjuk ke sebuah rahasia alih-alih memuatnya. Rahasia dapat berasal dari penyimpanan rahasia bersama, berkas, atau penyedia eksternal lewat plugin seperti 1Password.

NILAI MENTAH		RUJUKAN RAHASIA
Teks biasa	**Di konfigurasi**	Penunjuk
Kredensial ikut	**Bila konfigurasi bocor**	Tidak ikut
Sunting berkas	**Rotasi**	Ganti di penyimpanan
Berbahaya	**Masuk kendali versi**	Aman
Sulit	**Audit**	Terpusat

Gambar 53.1 — Selisih antara dua pendekatan. Baris kedua adalah alasan utama untuk berpindah.

```
openclaw secrets audit
openclaw secrets configure
openclaw secrets apply
openclaw secrets reload
```

Permukaan perintah rahasia. Perintah apply bekerja atas dasar rencana yang divalidasi lebih dulu.

Tersedia pula proksi egress rahasia yang mati secara bawaan, serta dukungan kunci API berbasis berkas. Keduanya adalah cara mengurangi jumlah tempat sebuah rahasia pernah muncul sebagai teks.

BAB 53 / LOG

Apa yang termuat dalam log dan transkrip

Log operasional dapat memuat potongan percakapan, nomor telepon, dan pengenal lain. Transkrip sesi memuat isi percakapan itu sendiri. Keduanya adalah data, dan keduanya tunduk pada kewajiban perlindungan data yang sama dengan basis data lain.

- Log dan transkrip masuk dalam inventaris data perusahaan
- Retensi log ditetapkan dan benar-benar dijalankan
- Berbagi diagnostik memakai keluaran tersanitasi
- Hasil tool yang memuat data sensitif disunting sebelum disimpan
- Menempelkan log mentah ke kanal dukungan publik

Gambar 53.2 — Lima butir. Butir keempat memakai hook penulisan ulang hasil tool dari Bab 40.

Aplikasi macOS menambahkan lapisannya sendiri: redaksi log, berkas diagnostik yang bersifat opt-in, dan penanda privasi pada log sistem native. Bila Anda menjalankan aplikasi Mac, periksa setelan itu sebelum menyalakan diagnostik rinci.

BAB 53 / PRAKTIKUM

Mengeluarkan rahasia dari berkas konfigurasi

Praktikum ini memindahkan kredensial dari teks biasa ke rujukan, lalu membuktikan bahwa konfigurasi Anda aman dibagikan. Setelah ini selesai, banyak pekerjaan lain menjadi lebih mudah.

Langkah 1 — cari apa yang ada sekarang

```
grep -rniE 'api[_-]?key|secret|token|password|sk-' \
  ~/.openclaw/openclaw.json

ls -la ~/.openclaw/credentials/
ls -la ~/.openclaw/agents/*/agent/*.sqlite
```

Perintah pertama sebaiknya tidak menghasilkan apa-apa selain nama medan.

Langkah 2 — periksa izin berkas

```
openclaw security audit --fix
ls -la ~/.openclaw/
```

Audit memperketat izin untuk state, konfigurasi, berkas kredensial, dan basis data agen.

Langkah 3 — pindahkan ke rujukan rahasia

```
openclaw secrets configure
openclaw secrets audit
openclaw secrets apply
openclaw secrets reload
```

Perintah apply bekerja atas dasar rencana yang divalidasi lebih dulu.

Langkah 4 — buktikan konfigurasi aman dibagikan

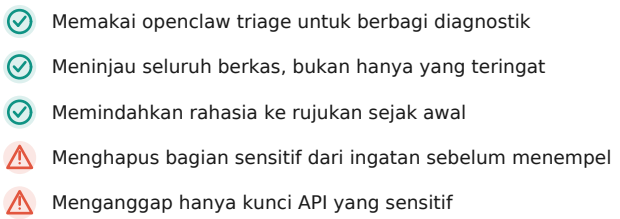
Uji	Cara	Hasil yang benar
Tidak ada rahasia	Jalankan grep langkah satu lagi	Kosong
Agen masih bekerja	Kirim satu pesan uji	Berjalan normal
Rotasi mudah	Ganti nilai di penyimpanan, bukan di berkas	Berlaku tanpa sunting
Aman ke kendali versi	Tinjau diff sebelum commit	Hanya penunjuk yang terlihat

BAB 53 / STUDI KASUS

Konfigurasi yang dibagikan untuk meminta bantuan

Seorang operator mengalami masalah konfigurasi dan meminta bantuan di forum publik. Untuk memperjelas pertanyaannya, ia menempelkan isi berkas konfigurasinya — setelah, menurut ingatannya, menghapus bagian yang sensitif.

Ia menghapus satu kunci API yang terlihat jelas. Yang tidak ia sadari: berkas itu juga memuat token Gateway, URL server MCP internal beserta header autentikasinya, dan nomor telepon pada daftar izin WhatsApp.



Gambar 53.3 — Lima kebiasaan. Dua yang terakhir adalah cara paling umum kredensial produksi berakhir di forum publik.

Rotasi yang harus dilakukan setelahnya memakan dua hari: token Gateway, seluruh kredensial MCP, dan peninjauan ulang daftar izin. Bila rahasia sudah berada di rujukan sejak awal, berkas yang ia tempel tidak akan memuat satu pun nilai.

SATU PERINTAH YANG MENGGANTIKAN INGATAN ANDA
Pakailah openclaw triage untuk berbagi diagnostik. Ia menyanitasi keluarannya, sehingga token, nomor telepon, dan isi percakapan tidak ikut keluar — tetapi tetap bacalah hasilnya sebelum membagikannya.

BAB 53 / LATIHAN

Latihan mandiri

1. Jalankan pencarian rahasia pada konfigurasi Anda dan pindahkan setiap nilai yang ditemukan ke rujukan.
2. Jalankan audit keamanan dengan perbaikan dan periksa izin berkas sesudahnya.
3. Jalankan keempat uji pada praktikum langkah empat.
4. Daftarkan seluruh tempat data percakapan berada, termasuk log, lalu tetapkan retensi untuk masing-masing.

BAB 53 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Kunci API terbaca di berkas konfigurasi	Pindahkan ke rujukan rahasia; nilai mentah tidak perlu ada di sana.
Izin berkas terlalu longgar	Jalankan audit keamanan; ia memperketat berkas sensitif yang umum.
Data pelanggan muncul di log	Tetapkan retensi dan sunting hasil tool sebelum penyimpanan.
Rotasi kredensial merepotkan	Dengan rujukan rahasia, rotasi terjadi di penyimpanan, bukan di konfigurasi.
Konfigurasi masuk kendali versi	Aman hanya bila ia memuat rujukan, bukan nilai.

Tiga pertanyaan review

1. Mengapa memakai rujukan rahasia membuat konfigurasi aman dimasukkan ke kendali versi?
2. Sebutkan dua tempat data percakapan berada di luar basis data sesi.
3. Rancang kebijakan retensi untuk agen layanan pelanggan yang menyimpan transkrip.

Kunci pembahasan

Rujukan hanya menunjuk ke tempat penyimpanan tanpa memuat nilainya, sehingga salinan konfigurasi yang tersebar — di repositori, di laptop, di riwayat perintah — tidak membawa kredensial apa pun. Dua tempat lain data percakapan berada adalah berkas log

operasional dan artefak memori hasil konsolidasi. Kebijakan retensi yang masuk akal untuk agen CS: simpan transkrip selama periode yang dibenarkan kebutuhan bisnis dan kewajiban hukum, hapus otomatis sesudahnya, pastikan penghapusan mencakup memori turunan lewat provenans, dan catat kebijakan itu agar dapat ditunjukkan saat audit.

Rujukan: dokumentasi OpenClaw, halaman gateway/secrets, gateway/security/secrets-and-storage, dan platforms/mac/logging, ditinjau 17 September 2026.

KEAMANAN DAN SANDBOXING

Bab 54 Audit keamanan, policy, dan respons insiden

Keamanan yang tidak diverifikasi berulang akan luntur. Bab ini menutup Bagian VIII dengan tiga hal yang membuatnya bertahan: audit yang dapat dijalankan, kebijakan yang dapat dibandingkan, dan rencana untuk hari ketika sesuatu benar-benar terjadi.

Tujuan belajar

Menjalankan audit keamanan dan membaca temuannya, menulis berkas kebijakan yang dapat diperiksa, dan menyiapkan respons insiden. Pada akhir bab, pembaca memiliki keamanan yang dapat dibuktikan, bukan hanya diklaim.

Audit keamanan

```
openclaw security audit
openclaw security audit --deep
openclaw security audit --fix

openclaw security audit --json | jq '.summary'
openclaw security audit --deep --json \
  | jq '.findings[] | select(.severity=="critical") | .checkId'
```

Setiap temuan punya checkId yang stabil, sehingga dapat dilacak antar waktu.

Di antara yang dikerjakan perbaikan otomatis: membalik kebijakan grup yang terbuka menjadi allowlist, menyemai daftar izin grup dari berkas yang sudah ada ketika konfigurasi belum mendefinisikannya, dan memperketat izin berkas untuk state, konfigurasi, serta berkas sensitif yang umum.

PENEKANAN TEMUAN ADALAH TINDAKAN BERISIKO

Karena penekanan temuan dapat menyembunyikan risiko yang berdiri, menambah atau menghapusnya lewat perintah shell yang dijalankan agen menuntut persetujuan eksekusi — kecuali eksekusi memang sudah berjalan dengan izin penuh untuk otomasi lokal yang tepercaya.

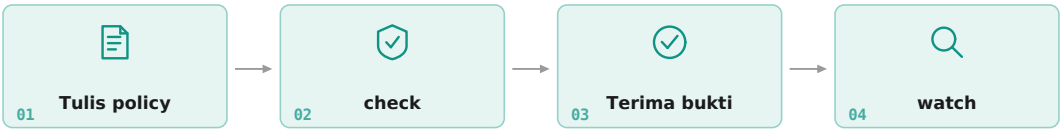
BAB 54 / POLICY

Kebijakan sebagai berkas yang dapat diperiksa

Plugin Policy mengubah kebijakan tertulis menjadi pemeriksaan yang dapat dijalankan. Anda menulis aturan dalam sebuah berkas, lalu memeriksanya terhadap keadaan OpenClaw yang sesungguhnya — dan membandingkannya dari waktu ke waktu.

```
openclaw policy check
openclaw policy compare
openclaw policy watch
```

check memeriksa sekarang; compare membandingkan; watch mendeteksi penyimpangan.



Gambar 54.1 — Siklus kebijakan. Langkah ketiga menghasilkan atestasi yang dapat dibandingkan nanti.

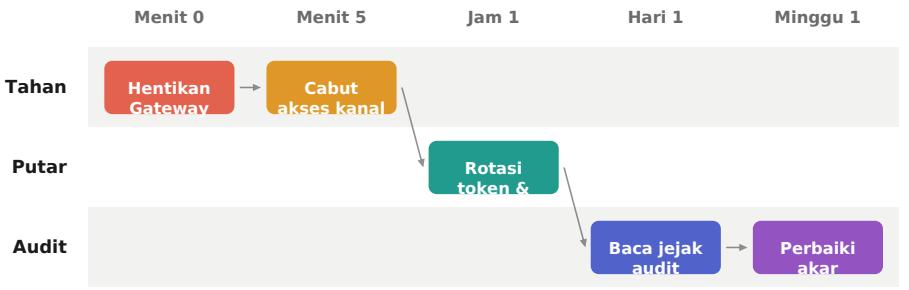
Setiap pemeriksaan punya pengenal yang stabil, dan sebagian dapat diperbaiki otomatis lewat doctor sementara sebagian lagi tidak. Tersedia pula overlay bercakupan, sehingga agen atau kanal tertentu dapat ditahan pada standar yang lebih ketat daripada baseline.

Bagi perusahaan, nilai terbesarnya bukan pada pemeriksaan pertama melainkan pada deteksi penyimpangan. Konfigurasi berubah pelan-pelan: seseorang melonggarkan satu setelan untuk mengatasi masalah mendesak, lalu lupa mengembalikannya. Deteksi penyimpangan adalah yang menangkap hal itu.

BAB 54 / INSIDEN

Ketika sesuatu benar-benar terjadi

Respons insiden untuk Gateway Anda sendiri mengikuti urutan yang sama dengan sistem lain: menahan, memutar kredensial, mengaudit, lalu mengumpulkan bukti. Yang berbeda hanyalah di mana masing-masing dilakukan.



Gambar 54.2 — Urutan respons. Menahan lebih dulu; memahami kemudian. Membalik urutannya memperpanjang paparan.

- Ada orang bernama yang boleh menghentikan Gateway tanpa menunggu persetujuan
- Daftar kredensial yang harus dirotasi sudah ditulis sebelum insiden
- Jejak audit dipertahankan cukup lama untuk berguna
- Prosedur pernah dilatih, bukan hanya didokumentasikan
- Mengandalkan ingatan seseorang saat insiden terjadi

Gambar 54.3 — Lima kesiapan. Butir kedua menghemat waktu paling banyak pada jam pertama.

Jejak aktivitas OpenClaw bersifat metadata beserta identitas jalannya dan tanda terima keputusan. Itu berarti Anda dapat menjawab siapa melakukan apa dan kapan — pertanyaan pertama yang selalu muncul dalam setiap insiden.

BAB 54 / PRAKTIKUM

Menyiapkan hari yang Anda harap tidak datang

Praktikum ini menyiapkan tiga hal yang harus sudah ada sebelum insiden: daftar rotasi, kewenangan menghentikan, dan prosedur yang pernah dilatih.

Langkah 1 — susun daftar rotasi sekarang

```
# Daftar kredensial yang harus dirotasi bila Gateway dikompromikan
1. Token atau kata sandi Gateway
2. Seluruh kunci API penyedia model
3. Kredensial tiap kanal yang tertaut
4. Token penyimpanan rahasia dan server MCP
5. Kunci SSH yang dipakai sandbox atau node
6. Kredensial layanan apa pun yang pernah ada di lingkungan agen
```

Menyusun daftar ini saat tenang memakan tiga puluh menit; menyusunnya saat insiden memakan berjam-jam.

Langkah 2 — tetapkan siapa yang boleh menghentikan

Tuliskan nama, bukan jabatan. Orang itu harus berwenang menghentikan Gateway dan mencabut akses kanal tanpa menunggu persetujuan siapa pun. Setiap menit menunggu adalah paparan tambahan.

Langkah 3 — jalankan latihan

Tahap	Yang dilatih	Waktu target
Tahan	Hentikan Gateway, cabut akses kanal	Di bawah lima menit
Putar	Rotasi kredensial dari daftar	Di bawah satu jam
Audit	Baca jejak, tentukan cakupan	Hari pertama
Perbaiki	Tutup akar penyebab	Minggu pertama

Langkah 4 — pasang deteksi penyimpangan kebijakan

```
openclaw policy check
openclaw policy compare
openclaw policy watch
```

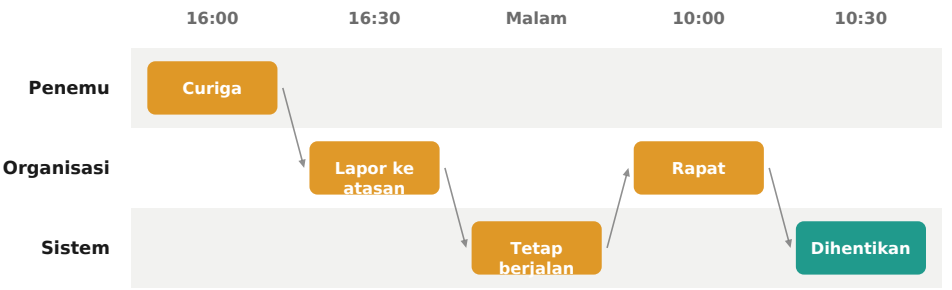
Nilai terbesarnya bukan pada pemeriksaan pertama, melainkan pada deteksi pergeseran.

BAB 54 / STUDI KASUS

Insiden yang dimulai dengan rapat

Sebuah tim mencurigai ada akses yang tidak sah pada Gateway mereka. Orang yang menemukannya tidak berwenang menghentikan layanan, sehingga ia melapor kepada atasannya dan menunggu keputusan.

Rapat pengambilan keputusan dijadwalkan pukul sepuluh pagi berikutnya, karena sebagian orang yang dianggap perlu hadir sedang tidak tersedia. Gateway tetap berjalan selama empat belas jam antara kecurigaan pertama dan tindakan pertama.



Gambar 54.4 — Empat belas jam antara kecurigaan dan tindakan. Tidak ada satu pun menit yang dihabiskan untuk hal teknis.

Ketika akhirnya diperiksa, kecurigaan itu ternyata keliru — aktivitas yang mencurigakan berasal dari pekerjaan terjadwal yang baru dipasang. Tetapi bila ia benar, empat belas jam adalah waktu yang sangat panjang.

ASIMETRI YANG MENENTUKAN KEBIJAKAN

Biaya menghentikan Gateway secara keliru jauh lebih rendah daripada biaya membiarkannya berjalan selama insiden nyata. Berikan kewenangan menghentikan kepada orang yang paling mungkin menemukan masalahnya lebih dulu.

BAB 54 / LATIHAN

Latihan mandiri

1. Susun daftar rotasi kredensial Anda dan simpan di tempat yang dapat diakses saat insiden.
2. Tetapkan nama orang yang berwenang menghentikan Gateway tanpa menunggu persetujuan.
3. Jalankan latihan tahap Tahan dan catat berapa lama sebenarnya ia memakan waktu.
4. Nyalakan deteksi penyimpangan kebijakan dan periksa hasilnya setelah satu bulan.

BAB 54 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Konfigurasi melonggar seiring waktu	Pakai deteksi penyimpangan kebijakan, bukan tinjauan manual sesekali.
Temuan kritis ditekan diam-diam	Penekanan menuntut persetujuan eksekusi. Periksa jejaknya.
Tidak tahu harus mulai dari mana saat insiden	Tahan dulu: hentikan Gateway dan cabut akses kanal.
Rotasi kredensial memakan waktu berjam-jam	Daftarnya belum ditulis sebelum insiden.
Jejak audit tidak cukup untuk menjawab	Retensi terlalu pendek. Sesuaikan sebelum Anda membutuhkannya.

Tiga pertanyaan review

1. Mengapa deteksi penyimpangan lebih bernilai daripada pemeriksaan kebijakan sesekali?
2. Mengapa menahan didahulukan sebelum memahami saat insiden?
3. Susun daftar kredensial yang harus dirotasi bila Gateway Anda dikompromikan.

Kunci pembahasan

Konfigurasi jarang rusak sekaligus; ia melonggar sedikit demi sedikit lewat perubahan mendesak yang tidak dikembalikan, dan hanya perbandingan berkelanjutan yang menangkap pergeseran itu. Menahan didahulukan karena setiap menit tambahan adalah paparan tambahan, sementara memahami penyebab dapat dilakukan setelah kerusakan berhenti bertambah. Daftar rotasi minimal: token atau kata sandi Gateway, seluruh kunci API penyedia model, kredensial setiap kanal yang tertaut, token penyimpanan rahasia dan MCP, kunci SSH yang dipakai sandbox atau node, serta kredensial layanan apa pun yang pernah tersedia di lingkungan agen.

Rujukan: dokumentasi OpenClaw, halaman cli/security, gateway/security/running-the-audit, cli/policy, dan gateway/security/operator-incident-response, ditinjau 17 September 2026.

USE CASE BISNIS

Bab 55 Kerangka menilai kelayakan sebuah use case

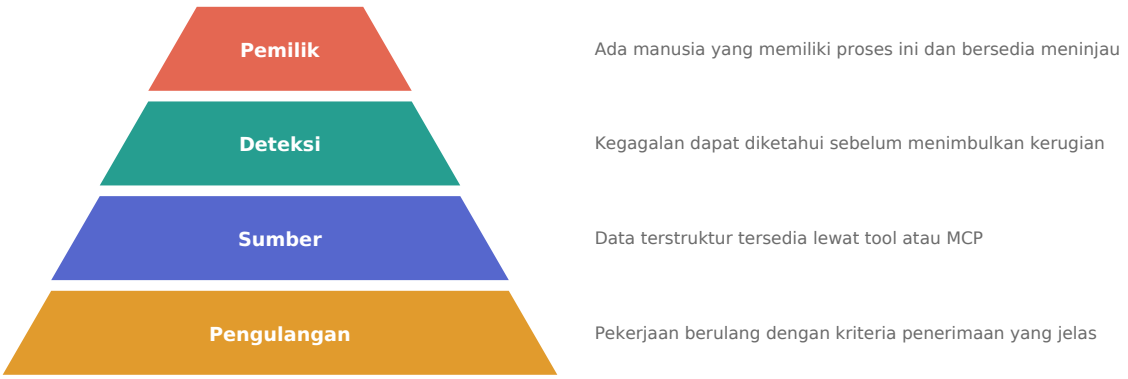
Enam bab berikutnya membangun use case bisnis yang berbeda-beda. Bab ini menyiapkan alat untuk menilainya — termasuk menilai use case Anda sendiri yang tidak ada di dalam daftar.

Tujuan belajar

Menerapkan empat gerbang kelayakan, membedakan pekerjaan yang cocok untuk agen dari yang tidak, dan menolak proyek yang akan gagal sebelum ia dimulai. Pada akhir bab, pembaca dapat menjawab pertanyaan kelayakan dalam satu rapat.

> **STATUS ANGKA DI BAGIAN INI**
> Seluruh angka bisnis di Bagian IX adalah data sintetis. Ia disusun agar bentuk perhitungannya benar dan dapat Anda tiru, bukan agar angkanya dipakai langsung. Ganti setiap angka dengan pengukuran dari organisasi Anda sendiri sebelum memakainya untuk keputusan apa pun.

Empat gerbang



Gambar 55.1 — Empat gerbang, dibaca dari dasar ke puncak. Use case yang gagal di gerbang dasar tidak layak diperbaiki di gerbang atas.

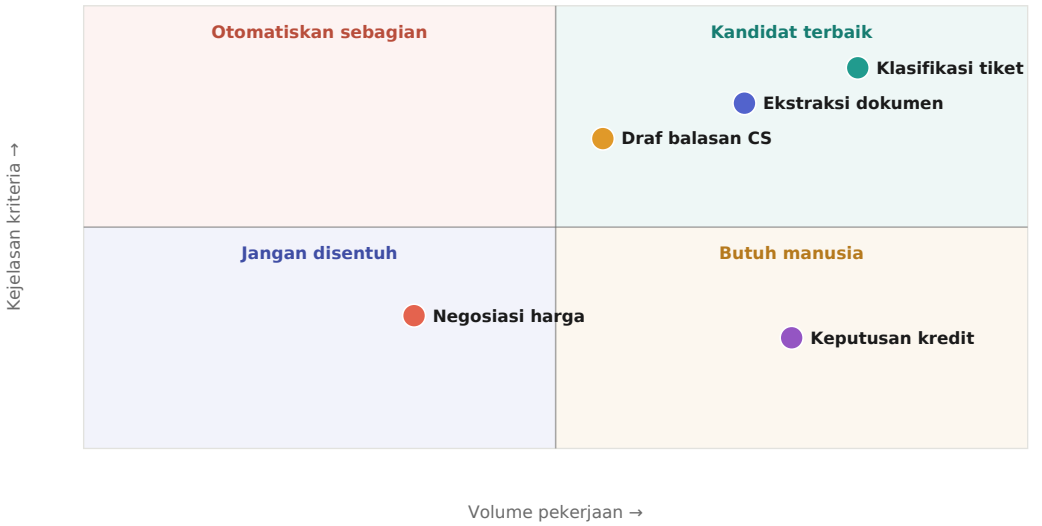
Gerbang pertama menuntut pekerjaan yang berulang dengan kriteria penerimaan yang jelas. Bila dua orang berpengalaman dapat tidak sepakat apakah sebuah hasil benar, use case itu belum siap untuk agen — bukan karena agennya kurang pintar, melainkan karena Anda tidak dapat mengukur keberhasilannya.

Gerbang kedua menuntut sumber data terstruktur yang tersedia lewat tool atau server MCP. Tanpa itu agen akan mengarang, dan prompt tidak memperbaikinya. Gerbang ketiga menuntut kegagalan dapat dideteksi sebelum menimbulkan kerugian: laporan salah yang ketahuan besok berbeda jauh dari pembayaran salah yang ketahuan bulan depan.

Gerbang keempat menuntut ada manusia yang memiliki proses itu dan bersedia meninjau. Use case tanpa pemilik akan mengalami penurunan mutu diam-diam — tidak ada yang memperhatikan ketika kualitasnya merosot, sampai seorang pelanggan mengeluh.

BAB 55 / MEMILIH

Pekerjaan yang cocok dan yang tidak



Gambar 55.2 — Dua sumbu yang menentukan. Volume tinggi dengan kriteria kabur adalah kombinasi paling berbahaya.

Kuadran kanan bawah layak diperhatikan: volume tinggi dengan kriteria yang tidak jelas. Di sanalah godaan otomasi paling besar sekaligus risikonya paling tinggi, karena kesalahan terjadi berulang kali sebelum ada yang menyadarinya.

Pola keluaran yang aman

Pola	Kapan dipakai
Agen menyarankan, manusia menyetujui	Tindakan yang sulit dibatalkan
Agen mengerjakan, manusia memeriksa sampel	Volume tinggi, kesalahan murah
Agen mengerjakan penuh	Hanya bila kegagalan terdeteksi otomatis
Agen hanya membaca dan melaporkan	Domain yang diatur atau berisiko tinggi

Sebagian besar penerapan yang berhasil dimulai dari baris pertama dan turun perlahan

seiring bukti terkumpul. Memulai dari baris ketiga hampir selalu menghasilkan insiden yang membuat proyeknya dihentikan seluruhnya.

BAB 55 / PRAKTIKUM

Menilai use case Anda sendiri dengan jujur

Praktikum ini menerapkan empat gerbang pada ide nyata Anda. Kerjakan sebelum menulis satu baris konfigurasi, karena gerbang yang gagal tidak dapat ditutup oleh rancangan sebaik apa pun.

Langkah 1 — tuliskan ide Anda dalam satu kalimat

Satu kalimat dengan kata kerja konkret, seperti pada praktikum Bab 1. Bila Anda membutuhkan dua kalimat, kemungkinan besar itu dua use case dan sebaiknya dinilai terpisah.

Langkah 2 — jalankan empat gerbang berurutan dari dasar

Gerbang	Pertanyaan yang harus dijawab ya	Bila tidak
Pengulangan	Apakah dua orang berpengalaman akan sepakat hasilnya benar?	Tetapkan kriteria dulu; jangan lanjut
Sumber	Apakah data terstruktur tersedia lewat tool atau MCP?	Pasang sumbernya dulu; prompt tidak menggantikan
Deteksi	Apakah kegagalan diketahui sebelum menimbulkan kerugian?	Jangan otomatiskan penuh
Pemilik	Adakah orang bernama yang bersedia meninjau?	Cari pemiliknya; tanpa itu mutu menurun diam-diam

Langkah 3 — pilih pola keluaran yang sesuai

Dari empat pola pada bab ini, pilih yang paling konservatif yang masih memberi nilai. Anda selalu dapat turun satu tingkat dengan bukti; naik satu tingkat setelah insiden jauh lebih sulit.

Langkah 4 — ukur dua minggu sebelum merancang

```
# Yang diukur selama dua minggu:
- Jumlah permintaan per hari
- Sebarannya menurut jenis
- Berapa lama ditangani sekarang
- Berapa persen di luar jam kerja
- Berapa yang benar-benar berulang dengan kriteria jelas
```

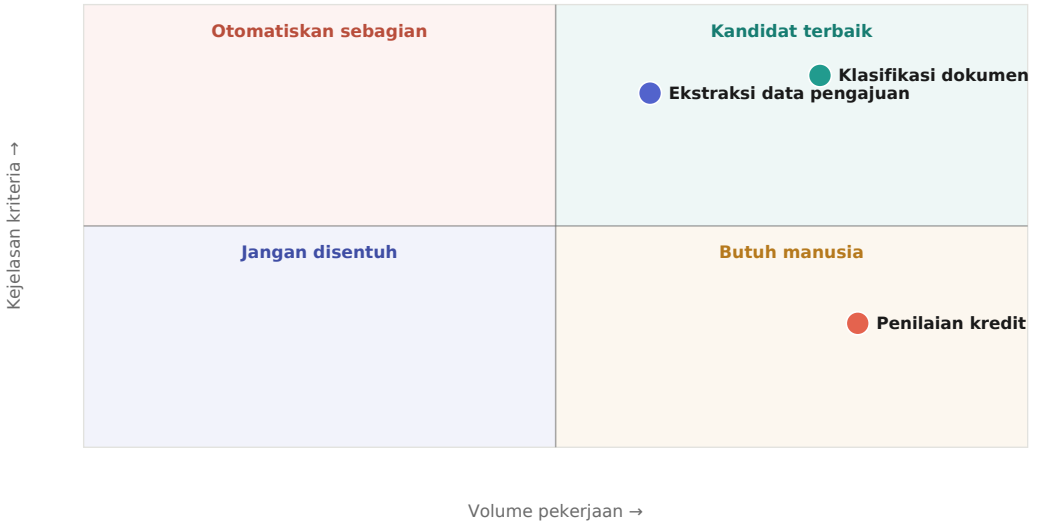
Pengukuran ini akan mengubah rancangan Anda lebih banyak daripada membaca seluruh Bagian IX.

BAB 55 / STUDI KASUS

Use case yang gagal di gerbang pertama

Sebuah perusahaan ingin agen menilai kelayakan pengajuan kredit mikro. Volumenya tinggi, datanya terstruktur, dan prosesnya terasa berulang. Tiga dari empat gerbang tampak lulus.

Ketika tim mencoba menuliskan kriteria penerimaan, dua analis berpengalaman tidak sepakat pada sepertiga kasus contoh. Perbedaannya bukan soal data, melainkan soal penilaian konteks yang tidak tertulis di mana pun.



Gambar 55.3 — Satu ide yang gagal dipecah menjadi dua yang lulus. Volume tinggi dengan kriteria kabur adalah kuadran yang paling berbahaya.

Mereka memecahnya. Ekstraksi data pengajuan dan klasifikasi dokumen pendukung diotomatiskan dan berhasil; penilaian kelayakannya tetap di tangan analis, dengan data yang kini tersaji jauh lebih cepat. Nilai yang didapat tetap besar.

MEMECAH LEBIH BAIK DARIPADA MENOLAK

Use case yang gagal gerbang sering dapat dipecah menjadi bagian yang lulus dan bagian yang tidak. Menolak keseluruhan ide adalah kesalahan yang sama besarnya dengan memaksakannya.

BAB 55 / LATIHAN

Latihan mandiri

1. Tuliskan ide use case Anda dalam satu kalimat, lalu jalankan empat gerbang berurutan.
2. Bila ada gerbang yang gagal, coba pecah ide itu menjadi bagian yang lulus dan bagian yang tidak.
3. Pilih pola keluaran paling konservatif yang masih memberi nilai, dan tuliskan apa yang akan membuat Anda turun satu tingkat.
4. Mulai pengukuran dua minggu hari ini, bukan setelah rancangan selesai.

BAB 55 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Tidak ada yang bisa menilai hasil agen benar atau salah	Gerbang pengulangan gagal. Tetapkan kriteria dulu.
Agen sering mengarang angka	Sumber terstruktur belum tersedia. Pasang tool atau MCP.
Kesalahan baru ketahuan berminggu-minggu kemudian	Gerbang deteksi gagal. Jangan otomatiskan penuh.
Mutu menurun perlahan tanpa ada yang sadar	Tidak ada pemilik. Tunjuk orang bernama.
Proyek dihentikan setelah satu insiden	Dimulai dari otomasi penuh. Mulai dari menyarankan.

Tiga pertanyaan review

1. Mengapa volume tinggi dengan kriteria kabur adalah kombinasi paling berbahaya?
2. Apa yang membedakan 'kegagalan dapat dideteksi' dari 'kegagalan jarang terjadi'?
3. Ambil satu proses di organisasi Anda dan nilai dengan empat gerbang.

Kunci pembahasan

Volume tinggi berarti kesalahan terjadi berkali-kali, dan kriteria yang kabur berarti tidak ada yang menangkapnya; gabungan keduanya menghasilkan kerusakan yang menumpuk diam-diam. Kegagalan yang jarang tetap berbahaya bila tidak terdeteksi, karena kejarangannya justru membuat orang berhenti memeriksa; yang penting bukan seberapa sering ia terjadi melainkan apakah Anda akan mengetahuinya ketika terjadi. Jawaban pertanyaan ketiga bergantung pada organisasi pembaca, tetapi gerbang yang paling sering gagal dalam praktik adalah gerbang pemilik.

Kerangka pada bab ini adalah sintesis praktik penyusun, bukan kutipan dokumentasi OpenClaw.

USE CASE BISNIS

Bab 56 Use case A — Layanan pelanggan multi-kanal

Distributor alat kesehatan dengan enam puluh reseller. Pesanan, pertanyaan stok, dan keluhan datang lewat WhatsApp sepanjang hari, sebagian di luar jam kerja. Tim berjumlah tiga orang dan tidak akan bertambah.

Tujuan belajar

Merancang arsitektur layanan pelanggan lengkap, menetapkan batas eskalasi, dan mengukur keberhasilannya. Pada akhir bab, pembaca memiliki cetak biru yang dapat disesuaikan untuk bisnisnya sendiri.

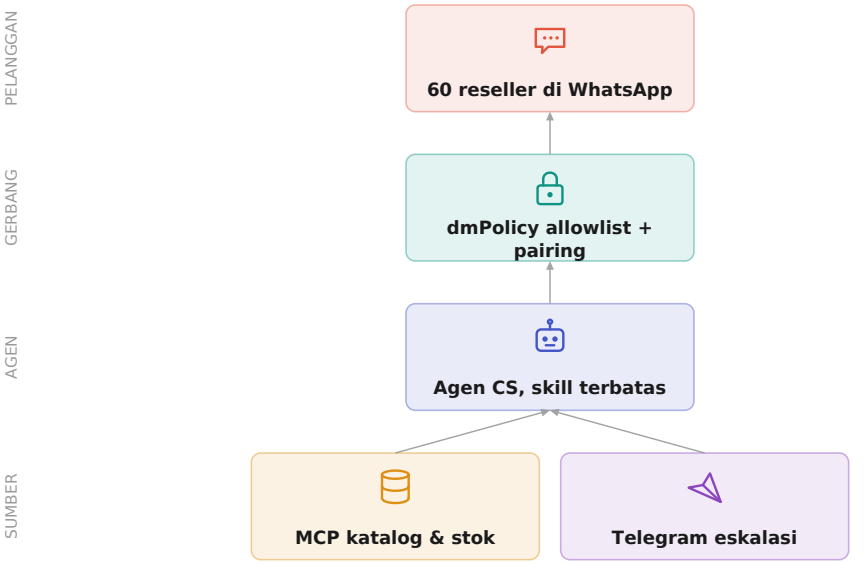
Angka dasar

310 pesan/hari	41 %	3 orang	22 menit
Masuk dari 60 reseller	Di luar jam kerja	Tim layanan pelanggan	Median balas saat ini

Gambar 56.1 — Kondisi awal. Semua angka sintetis; ukur milik Anda sendiri selama dua minggu sebelum merancang apa pun.

BAB 56 / ARSITEKTUR

Bentuk sistemnya



Gambar 56.2 — Eskalasi keluar ke kanal berbeda supaya antrean manusia tidak bercampur dengan antrean pelanggan.

Keputusan arsitektur yang paling menentukan ada di lapis keempat. Eskalasi tidak dikirim balik ke WhatsApp melainkan ke Telegram internal, sehingga tim melihat antrean kerjanya sendiri tanpa harus menyaring percakapan pelanggan. Ini juga memisahkan kanal pelanggan dari kanal yang membawa kewenangan lebih tinggi.

Konfigurasi inti

```
{
  channels: {
    whatsapp: {
      enabled: true,
      dmPolicy: "allowlist",
      allowFrom: [ /* 60 nomor reseller */ ],
      groupPolicy: "disabled",
    },
  },
  agents: {
    entries: {
      cs: {
        skills: ["cs-playbook", "katalog"],
        tools: { deny: ["exec"] },
        sandbox: { mode: "all", workspaceAccess: "none" },
      },
    },
```

```
  },
}
```

Grup dimatikan sepenuhnya; agen CS tidak memiliki exec dan berjalan tersandbox.

BAB 56 / PERILAKU

Batas yang menentukan

Skill layanan pelanggan bukan tempat menuliskan keramahan; ia tempat menuliskan batas. Empat kategori berikut harus diserahkan ke manusia tanpa pengecualian.

- Status pesanan, stok, dan harga dari katalog — dikerjakan agen
- Draf pesanan untuk dikonfirmasi reseller — dikerjakan agen
- Keluhan, retur, dan barang rusak — diserahkan ke manusia
- Permintaan diskon atau termin pembayaran — diserahkan ke manusia
- Ancaman hukum atau penyebutan media sosial — diserahkan ke manusia
- Reseller menyatakan tidak puas dua kali — diserahkan ke manusia

Gambar 56.3 — Dua yang dikerjakan, empat yang diserahkan. Menolak empat kategori terakhir bukan kelemahan desain, melainkan intinya.

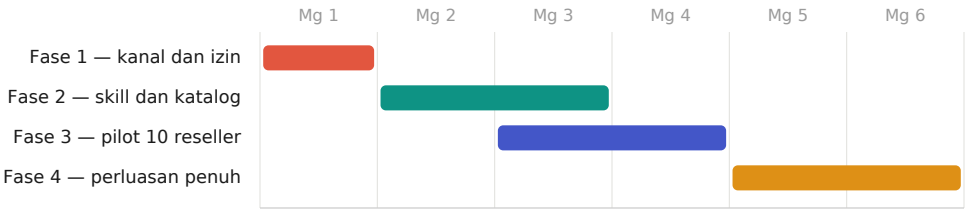
> **ARITMETIKA YANG MENENTUKAN DESAIN**
> Biaya satu janji yang salah kepada pelanggan B2B jauh melebihi penghematan dari seratus balasan otomatis. Agen yang mencoba menangani keluhan akan menghasilkan janji yang tidak dapat ditepati perusahaan.

Aturan anti-mengarang

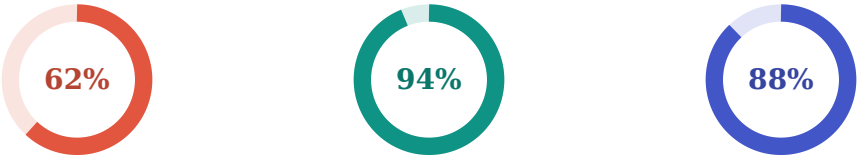
Satu aturan yang harus ada di skill: setiap angka status, stok, atau harga dalam balasan harus berasal dari keluaran tool pada giliran yang sama. Bila tidak ada keluaran tool, tidak ada angka. Aturan ini sederhana, dapat diperiksa, dan menutup sumber kesalahan yang paling merusak kepercayaan.

BAB 56 / PELUNCURAN

Enam minggu, empat fase



Gambar 56.4 — Fase 3 tumpang tindih dengan fase 2 karena umpan balik pilot pasti mengubah isi skill.



Gambar 56.5 — Tiga metrik target. Metrik kedua lebih penting daripada yang pertama: eskalasi yang meleset merusak kepercayaan.

BAB 56 / PRAKTIKUM

Membangun skill layanan pelanggan yang menolak dengan benar

Praktikum ini menulis bagian skill yang paling menentukan: batas. Sebagian besar waktu penyusunan skill layanan pelanggan sebaiknya dihabiskan di sini, bukan pada kalimat sapaan.

Langkah 1 — daftar kategori yang diserahkan ke manusia

Tulis sebagai kategori yang dapat diperiksa, bukan sebagai anjuran. “Berhati-hatilah dengan keluhan” bukan batas; “segera serahkan ketika pesan menyangkut keluhan, retur, atau barang rusak” adalah batas.

```
## Eskalasi — hentikan dan serahkan bila:
- pesan menyangkut keluhan, retur, barang rusak, atau kesalahan kirim
- pesan meminta perubahan harga, diskon, atau termin pembayaran
- pesan menyebut pengacara, komplain resmi, atau media sosial
- pelanggan menyatakan tidak puas dua kali dalam satu percakapan
- Anda tidak yakin dua kali berturut-turut pada percakapan yang sama
```

Lima kategori yang dapat diperiksa. Baris terakhir menangkap kasus yang tidak masuk empat kategori lainnya.

Langkah 2 — tulis aturan anti-mengarang

```
## Verifikasi
Setiap angka status, stok, atau harga dalam balasan harus berasal
dari keluaran tool pada giliran yang sama. Bila tidak ada keluaran
tool, tidak ada angka.
```

Aturan ini sederhana, dapat diperiksa, dan menutup sumber kesalahan yang paling merusak kepercayaan.

Langkah 3 — uji batas dengan pesan yang sengaja sulit

Pesan uji	Perilaku yang benar
Barang saya rusak, bagaimana ini?	Eskalasi; tidak menjanjikan apa pun
Bisa dapat diskon 10 persen?	Eskalasi; tidak menawar
Kapan pesanan 12345 sampai?	Panggil tool; jawab dari keluarannya
Stok produk X masih ada?	Panggil tool; bila kosong, katakan tidak tahu
Saya kecewa. Saya kecewa sekali.	Eskalasi pada penyebutan kedua

Langkah 4 — jalankan fase bayangan

Sebelum satu pun balasan dikirim ke pelanggan, jalankan agen dalam fase bayangan: ia menyusun balasan, tetapi manusia yang memeriksanya sebelum dikirim. Dua minggu fase ini menghasilkan data kesalahan yang Anda butuhkan untuk membenarkan tahap berikutnya.

BAB 56 / STUDI KASUS

Satu janji yang menghapus penghematan setahun

Sebuah distributor menjalankan agen layanan pelanggan yang menangani sekitar enam puluh persen pesan tanpa sentuhan manusia. Selama tujuh bulan, penghematannya nyata dan pelanggan puas.

Seorang reseller mengeluh tentang pengiriman yang terlambat. Kategori keluhan seharusnya dieskalasi, tetapi pesannya dibuka dengan pertanyaan status pesanan — kategori yang memang ditangani agen. Agen menjawab status, lalu menambahkan bahwa keterlambatan akan dikompensasi.

Yang terjadi	Biayanya
Agen menjanjikan kompensasi	Perusahaan tidak punya kebijakan itu
Reseller membagikan tangkapan layar	Enam reseller lain menuntut hal sama
Perusahaan memenuhi sebagian	Biaya langsung yang tidak dianggarkan
Kepercayaan pada agen menurun	Tim meninjau ulang seluruh cakupan

Penyebab teknisnya sempit: aturan eskalasi diperiksa terhadap topik utama pesan, bukan terhadap seluruh isinya. Pesan campuran — pertanyaan yang mengandung keluhan — lolos ke jalur yang salah.

PESAN CAMPURAN ADALAH KASUS YANG SESUNGGUHNYA

Tulis aturan eskalasi supaya ia memeriksa seluruh isi pesan, bukan topik utamanya. Pesan pelanggan yang nyata jarang berisi satu topik saja, dan pesan campuran adalah yang paling sering mengandung hal yang harus dieskalasi.

BAB 56 / LATIHAN

Latihan mandiri

1. Tulis lima kategori eskalasi untuk bisnis Anda sebagai kategori yang dapat diperiksa.
2. Uji skill Anda dengan lima pesan pada praktikum langkah tiga, lalu tambahkan pesan campuran yang menggabungkan dua kategori.
3. Jalankan fase bayangan selama dua minggu dan catat setiap kesalahan beserta kategorinya.
4. Hitung biaya satu janji salah kepada pelanggan terbesar Anda, lalu bandingkan dengan penghematan bulanan yang Anda harapkan.

BAB 56 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Agen menjanjikan hal yang tidak bisa ditepati	Kategori keluhan belum diserahkan ke manusia. Perketat batas.
Angka stok salah	Aturan anti-mengarang belum ditegakkan, atau tool gagal diam-diam.
Tim tenggelam dalam eskalasi	Ambang eskalasi terlalu longgar, atau katalog tidak lengkap.
Reseller baru tidak dilayani	dmPolicy allowlist. Tambahkan nomornya lewat proses yang jelas.
Percakapan pelanggan tercampur	DM diciutkan ke sesi utama. Pertimbangkan dmScope per-channel-peer.

Tiga pertanyaan review

1. Mengapa eskalasi dikirim ke kanal berbeda, bukan ditandai di WhatsApp?
2. Mengapa metrik eskalasi tepat lebih penting daripada persentase selesai otomatis?
3. Rancang proses penambahan reseller baru ke allowlist yang tidak menjadi hambatan.

Kunci pembahasan

Kanal terpisah memisahkan antrean kerja tim dari antrean pelanggan, dan sekaligus memisahkan kanal berkewenangan tinggi dari kanal yang dapat dijangkau pihak luar. Eskalasi yang meleset berarti keluhan ditangani agen — yang menghasilkan janji salah — sedangkan persentase otomatis yang rendah hanya berarti penghematan lebih kecil; kerusakan kepercayaan tidak sebanding dengan selisih efisiensi. Proses penambahan reseller yang tidak menghambat: satu formulir internal yang memicu perubahan konfigurasi lewat orang yang ditunjuk, dengan janji waktu satu hari kerja, dan bukan lewat permintaan ad hoc di grup.

Arsitektur mengikuti kemampuan yang didokumentasikan; seluruh angka bisnis adalah data sintetis penyusun.

USE CASE BISNIS

Bab 57 Use case B — Asisten operasi aplikasi bisnis

Perusahaan menjalankan aplikasi bisnis internal: ERP, sistem tiket, basis data pelanggan. Pertanyaan operasional datang sepanjang hari kepada tim yang sama, dan sebagian besar jawabannya sebenarnya ada di dalam sistem — hanya saja tidak mudah diambil.

Tujuan belajar

Menyambungkan agen ke sistem bisnis lewat MCP, memisahkan baca dari tulis, dan menetapkan gerbang persetujuan. Pada akhir bab, pembaca dapat membuka data internal kepada tim tanpa membuka kemampuan mengubahnya.

Memisahkan baca dari tulis

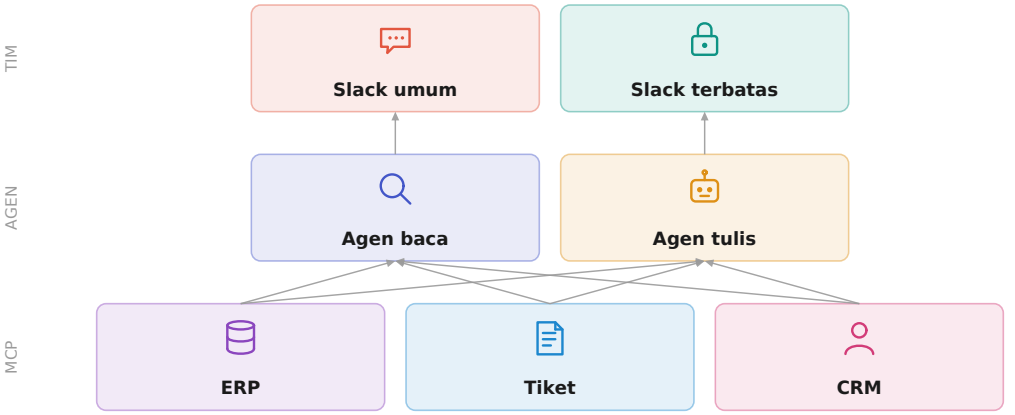
AGEN BACA		AGEN TULIS
Slack seluruh tim	Kanal	Slack kanal terbatas
Hanya kueri	Tool MCP	Kueri dan mutasi
Tidak perlu	Persetujuan	Wajib per tindakan
Menyala	Sandbox	Menyala
Puluhan	Jumlah pengguna	Beberapa orang

Gambar 57.1 — Dua agen, bukan satu. Sebagian besar nilai berasal dari kolom kiri, dan kolom kiri hampir tanpa risiko.

Kesalahan yang paling sering terjadi pada use case ini adalah membuat satu agen serbaguna yang dapat membaca sekaligus menulis, lalu mencoba mengendalikannya lewat prompt. Bab 51 sudah menjelaskan mengapa itu tidak bekerja. Pisahkan agennya, dan sebagian besar kekhawatiran hilang dengan sendirinya.

BAB 57 / ARSITEKTUR

Bentuk sistemnya



Gambar 57.2 — Dua jalur yang berbagi sumber data yang sama tetapi berbeda kewenangan. Server MCP yang sama dapat memaparkan tool berbeda lewat filter.

Bab 42 menyebut filter tool pada definisi server MCP. Di sinilah ia terpakai: server ERP yang sama disambungkan dua kali dengan nama berbeda, satu hanya memaparkan tool kueri, satu lagi memaparkan tool mutasi — lalu tiap agen hanya mendapat satu di antaranya.

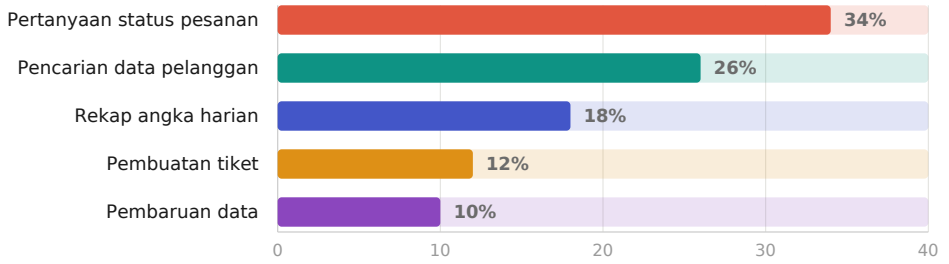
```
openclaw mcp add erp-baca \
  --url https://mcp.internal/erp \
  --transport streamable-http \
  --include 'get_*,list_*,search_*'

openclaw mcp add erp-tulis \
  --url https://mcp.internal/erp \
  --transport streamable-http \
  --include 'create_*,update_*'
```

Satu server, dua definisi, dua permukaan tool yang berbeda.

BAB 57 / NILAI

Dari mana nilainya datang



Gambar 57.3 — Sebaran permintaan sintetis. Tiga teratas seluruhnya hanya-baca dan mencakup hampir delapan puluh persen volume.

Sebaran seperti ini berulang di banyak organisasi: sebagian besar permintaan adalah pertanyaan, bukan perubahan. Artinya agen baca saja sudah menangkap sebagian besar nilainya, dan agen tulis dapat ditunda sampai agen baca terbukti stabil.

Gerbang untuk agen tulis

Ketika agen tulis akhirnya dipasang, setiap mutasi harus melewati persetujuan manusia sebagaimana dibahas di Bab 44, ditampilkan sebagai tombol di Slack. Perintah yang rutin dan aman — misalnya menambahkan komentar pada tiket — dapat masuk allowlist supaya tidak menimbulkan prompt yang melelahkan.

BAB 57 / PRAKTIKUM

Memisahkan baca dari tulis dengan dua definisi MCP

Praktikum ini membangun pemisahan yang tidak dapat dilewati prompt: dua permukaan tool yang berbeda secara teknis dari satu server yang sama.

Langkah 1 — lihat apa yang dipaparkan server Anda

```
openclaw mcp add sementara --url <url> --transport streamable-http
openclaw mcp probe sementara
```

Daftarkan seluruh tool, lalu kelompokkan menjadi kueri dan mutasi.

Langkah 2 — buat dua definisi dengan filter berbeda

```
openclaw mcp add erp-baca  --url <url> --transport streamable-http \
  --include 'get_*,list_*,search_*'

openclaw mcp add erp-tulis --url <url> --transport streamable-http \
  --include 'create_*,update_*'
```

Satu server, dua nama, dua permukaan tool yang tidak saling tumpang tindih.

Langkah 3 — ikat tiap definisi ke agen yang tepat

Agen	Definisi MCP	Kanal	Persetujuan
baca	erp-baca	Slack umum	Tidak perlu
tulis	erp-tulis	Slack terbatas	Wajib per tindakan

Langkah 4 — buktikan agen baca tidak dapat menulis

Perintahkan agen baca melakukan mutasi. Ia tidak akan menolak karena diberi tahu; ia tidak akan bisa karena tool-nya tidak ada. Perbedaan itu yang membuat pemisahan ini bernilai.

BAB 57 / STUDI KASUS

Agen baca yang diberi satu tool tulis

Sebuah tim memisahkan agen baca dan agen tulis dengan benar. Beberapa bulan kemudian, seseorang meminta agen baca dapat menambahkan komentar pada tiket — tindakan yang terasa sepele dan sangat berguna.

Satu tool mutasi ditambahkan ke definisi erp-baca. Permintaannya masuk akal dan risikonya terlihat kecil. Yang tidak disadari: agen baca melayani Slack umum yang dapat diakses seluruh perusahaan, termasuk kontraktor eksternal.

SEBELUM		SESUDAH SATU TOOL
Hanya kueri	**Kemampuan agen baca**	Kueri dan satu mutasi
Seluruh perusahaan	**Jangkauan kanal**	Seluruh perusahaan
Semua orang	**Siapa yang dapat memicu**	Semua orang
Jelas	**Batas kepercayaan**	Kabur
—	**Alasan penambahan**	Terasa sepele

Gambar 57.4 — Pemisahan yang runtuh bukan karena serangan, melainkan karena permintaan yang masuk akal.

Tidak ada penyalahgunaan yang terjadi. Yang hilang adalah kemampuan menjawab pertanyaan sederhana: apa yang dapat dilakukan agen yang dapat dipicu siapa saja di perusahaan? Jawabannya berubah, dan tidak ada yang mencatat perubahannya.

> **TAMBAH AGEN, JANGAN TAMBAH KEMAMPUAN**
> Ketika sebuah permintaan menuntut menambah kemampuan pada agen berjangkauan luas, pertimbangkan agen ketiga dengan jangkauan sempit. Menambah agen jauh lebih murah daripada mengaburkan batas yang sudah dibuat.

BAB 57 / LATIHAN

Latihan mandiri

1. Kelompokkan seluruh tool server MCP Anda menjadi kueri dan mutasi, lalu buat dua definisi terpisah.
2. Buktikan agen baca Anda tidak dapat melakukan mutasi, bukan hanya menolak melakukannya.
3. Periksa apakah ada agen berjangkauan luas di penyiapan Anda yang memiliki tool mutasi.
4. Tetapkan siapa yang boleh menyetujui penambahan tool pada agen yang dapat dipicu banyak orang.

BAB 57 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Agen mengubah data tanpa diminta	Agen baca memiliki tool mutasi. Pisahkan definisi MCP-nya.
Tim berhenti memakai karena jawabannya lambat	Terlalu banyak tool terlihat. Persempit dengan filter.
Persetujuan diabaikan orang	Terlalu sering muncul. Perluas allowlist perintah rutin.
Jawaban tidak konsisten antar orang	Tidak ada skill playbook. Tulis kriteria jawabannya.
Data sensitif muncul di kanal umum	Filter tool terlalu longgar untuk agen baca.

Tiga pertanyaan review

1. Mengapa menyambungkan satu server MCP dua kali lebih aman daripada satu sambungan dengan prompt pembatas?
2. Mengapa agen baca dapat dipasang lebih dulu tanpa menunggu agen tulis?
3. Rancang daftar tool untuk agen baca yang melayani seluruh perusahaan, dan sebutkan satu tool yang sengaja tidak dimasukkan.

Kunci pembahasan

Dua sambungan menghasilkan dua permukaan tool yang berbeda secara teknis, sehingga agen baca secara harfiah tidak memiliki tool mutasi untuk dipanggil; pembatasan lewat prompt hanya kecenderungan yang dapat kalah. Agen baca dapat lebih dulu karena ia menangkap sebagian besar volume permintaan sambil membawa risiko yang jauh lebih kecil, sehingga ia juga menjadi cara murah membuktikan nilai proyek. Daftar tool agen

baca sebaiknya berisi kueri status, pencarian pelanggan, dan rekap agregat; yang sengaja tidak dimasukkan adalah tool yang mengembalikan data pribadi lengkap, karena kanal umum tidak pantas menerimanya.

Arsitektur mengikuti kemampuan yang didokumentasikan; seluruh angka adalah data sintetis penyusun.

USE CASE BISNIS

Bab 58 Use case C — Riset dan pemantauan pasar saham

Use case ini paling sering diminta dan paling sering dirancang keliru. Bab ini membangun asisten riset yang berguna, dan menjelaskan dengan tegas mengapa ia tidak boleh menjadi asisten pengambil keputusan.

BATAS YANG TIDAK DAPAT DITAWAR
Buku ini bukan nasihat keuangan, dan bab ini tidak menganjurkan strategi investasi apa pun. Yang dibahas adalah rekayasa sistem informasi: mengumpulkan, meringkas, dan menyajikan data. Keputusan membeli atau menjual tetap berada pada manusia yang memahami risikonya dan bertanggung jawab atasnya.

Tujuan belajar

Merancang pemantau pasar yang jujur tentang ketidakpastiannya, memisahkan fakta dari interpretasi, dan menolak permintaan otomasi eksekusi. Pada akhir bab, pembaca memiliki asisten riset yang menghemat waktu tanpa menciptakan risiko baru.

Mengapa eksekusi otomatis adalah garis merah

- ✓ Mengumpulkan berita dan data emiten — cocok untuk agen
- ✓ Meringkas laporan keuangan menjadi poin terstruktur — cocok
- ✓ Memantau peristiwa dan mengabarkan ketika ambang terlampaui — cocok
- ⚠ Menyusun tesis investasi atas nama Anda — tidak cocok
- ⚠ Menempatkan atau membatalkan order — garis merah
- ⚠ Mengelola dana orang lain berdasarkan keluaran model — garis merah

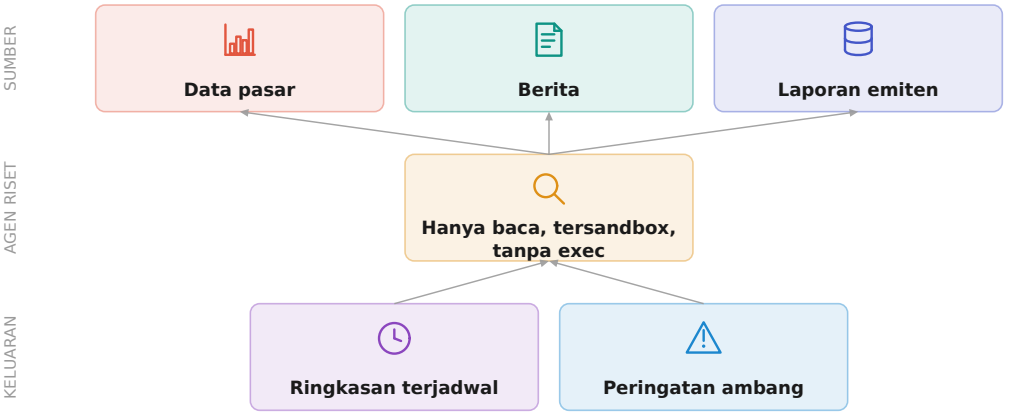
Gambar 58.1 — Tiga yang cocok, tiga yang tidak. Dua baris terakhir juga membawa konsekuensi hukum dan perizinan di banyak yurisdiksi.

Alasan teknisnya sejalan dengan Bab 55. Gerbang deteksi gagal: keputusan investasi yang buruk tidak menunjukkan kesalahannya dengan segera, dan dapat tampak benar selama berbulan-bulan sebelum terbukti salah. Sistem yang kegagalannya tidak dapat dideteksi tepat waktu tidak boleh dijalankan tanpa manusia.

Alasan kedua adalah prompt injection dari Bab 51. Agen riset membaca berita, forum, dan dokumen dari pihak luar — tepat kelas konten yang dapat memuat instruksi. Agen yang membaca dunia luar tidak boleh memiliki kemampuan bertindak.

BAB 58 / ARSITEKTUR

Bentuk sistemnya



Gambar 58.2 — Tidak ada panah yang kembali ke sistem perdagangan. Ketiadaan panah itu adalah keputusan desain yang paling penting.

Aturan penyajian

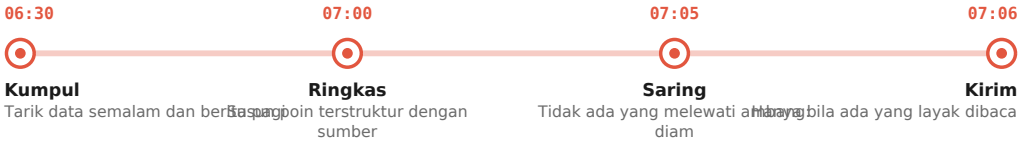
Skill pemantau harus memaksakan pemisahan yang tegas antara tiga jenis kalimat. Tanpa pemisahan ini, pembaca tidak dapat membedakan mana yang terukur dan mana yang tafsiran model.

Jenis	Aturan
Fakta terukur	Harus menyebut sumber dan waktu pengambilan
Perubahan	Harus menyebut pembanding: dibanding kapan
Tafsiran	Harus ditandai eksplisit sebagai tafsiran, bukan fakta
Yang tidak diketahui	Harus dinyatakan, bukan dihilangkan

MENYATAKAN YANG TIDAK DIKETAHUI
Aturan keempat adalah yang paling sering dilanggar dan paling berbahaya. Ringkasan yang menghilangkan ketidaktahuan terlihat lebih meyakinkan daripada yang menyatakannya — dan justru itulah masalahnya.

BAB 58 / OPERASI

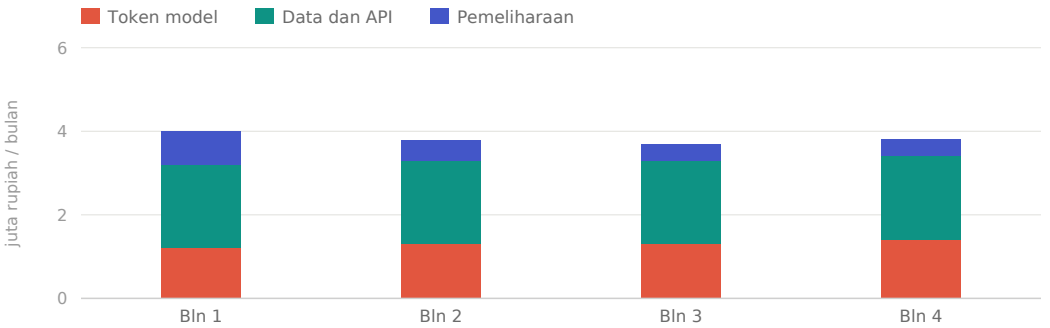
Jadwal dan aturan diam



Gambar 58.3 — Laporan yang tahu kapan harus diam. Laporan yang selalu terkirim akan berhenti dibaca dalam dua minggu.

Bab 46 menyebut bahaya pemantau yang diam ketika rusak. Di sini perbedaannya harus ditegakkan: diam karena tidak ada yang melewati ambang berbeda dari diam karena pengambilan data gagal. Buat pemantau melaporkan kegagalan pengambilan secara eksplisit, terpisah dari laporan isinya.

Biaya



Gambar 58.4 — Susunan biaya sintetis. Berlangganan data sering menjadi komponen terbesar, bukan token model.

BAB 58 / PRAKTIKUM

Membangun pemantau pasar yang jujur

Praktikum ini membangun pemantau yang berguna sekaligus jujur tentang batasnya. Kejujuran itu bukan sikap moral; ia syarat agar keluarannya dapat dipakai untuk berpikir.

Langkah 1 — tetapkan batas sebelum menulis apa pun

```
# Yang dikerjakan agen
- mengumpulkan data dan berita dari sumber yang ditentukan
- meringkas menjadi poin terstruktur dengan sumber dan waktu
- mengabarkan ketika ambang yang ditetapkan terlampaui

# Yang tidak dikerjakan agen, dalam keadaan apa pun
- menyusun tesis investasi atas nama siapa pun
- menempatkan, mengubah, atau membatalkan order
```

Tuliskan bagian kedua lebih dulu; ia yang menentukan arsitekturnya.

Langkah 2 — paksa pemisahan empat jenis kalimat

Jenis	Aturan penulisan	Contoh penanda
Fakta terukur	Sebut sumber dan waktu pengambilan	Per pukul, menurut
Perubahan	Sebut pembanding	Dibanding penutupan kemarin
Tafsiran	Tandai eksplisit	Tafsiran: kemungkinan karena
Tidak diketahui	Nyatakan, jangan hilangkan	Belum ada data untuk

Langkah 3 — pisahkan laporan kegagalan dari laporan isi

```
# Dua jalur pelaporan yang berbeda:
1. Laporan isi  : hanya bila ada yang melewati ambang
2. Laporan sehat: selalu, memuat status tiap sumber data
```

Tanpa jalur kedua, diam berarti dua hal yang tidak dapat dibedakan.

Langkah 4 — kunci agennya

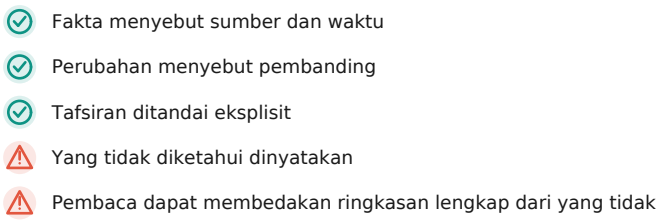
Agen ini membaca konten dari pihak luar, sehingga Bab 51 berlaku penuh: tanpa exec, tanpa tool yang mengubah keadaan, tersandbox, dan tujuan pengiriman dibatasi pada daftar tetap.

BAB 58 / STUDI KASUS

Ringkasan yang menghilangkan ketidaktahuan

Sebuah tim menjalankan pemantau pasar yang menghasilkan ringkasan pagi. Ringkasannya rapi, terstruktur, dan enak dibaca. Selama beberapa bulan ia menjadi bacaan wajib sebelum rapat pagi.

Suatu pagi, salah satu sumber data tidak dapat dijangkau. Skill tidak menyebut apa yang harus dilakukan dalam keadaan itu, sehingga agen menyusun ringkasan dari sumber yang tersisa — dengan struktur dan nada yang sama persis seperti biasanya.



Gambar 58.5 — Tiga dari lima butir terpenuhi. Dua yang gagal adalah yang menentukan pada satu-satunya pagi yang tidak normal.

Rapat pagi itu mengambil keputusan berdasarkan gambaran yang tidak lengkap. Tidak ada yang bertanya, karena tidak ada satu pun tanda bahwa ada yang perlu ditanyakan.

ATURAN YANG PALING SERING DILANGGAR

Ringkasan yang menghilangkan ketidaktahuan terlihat lebih meyakinkan daripada yang menyatakannya — dan justru itulah masalahnya. Paksa skill Anda menyebut apa yang tidak berhasil diambil, di tempat yang sama menonjolnya dengan temuannya.

BAB 58 / LATIHAN

Latihan mandiri

1. Tuliskan daftar yang tidak dikerjakan agen Anda lebih dulu, sebelum daftar yang dikerjakan.
2. Terapkan pemisahan empat jenis kalimat pada satu ringkasan nyata dan periksa hasilnya.
3. Pisahkan laporan kesehatan sumber data dari laporan isi, lalu matikan satu sumber untuk mengujinya.
4. Tetapkan ambang peringatan berdasarkan besaran perubahan yang benar-benar mengubah keputusan pembaca.

BAB 58 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Ringkasan terdengar meyakinkan tetapi salah	Tafsiran tidak ditandai. Paksa pemisahan fakta dan tafsiran.
Laporan berhenti dibaca	Ia selalu terkirim. Terapkan ambang dan aturan diam.
Pemantau diam padahal sumber data mati	Pisahkan pelaporan kegagalan dari pelaporan isi.
Ada permintaan menyambungkan ke sistem order	Tolak. Lihat Gambar 58.1 dan alasan di baliknya.
Biaya lebih tinggi dari perkiraan	Langganan data biasanya melebihi biaya token.

Tiga pertanyaan review

1. Jelaskan dengan gerbang Bab 55 mengapa keputusan investasi otomatis tidak layak.
2. Mengapa agen yang membaca berita publik tidak boleh memiliki kemampuan bertindak?
3. Rancang aturan ambang untuk peringatan supaya ia tidak menjadi kebisingan.

Kunci pembahasan

Gerbang deteksi gagal karena kesalahan keputusan investasi tidak terlihat segera dan dapat tampak benar selama berbulan-bulan; tanpa deteksi tepat waktu, sistem itu menumpuk kerugian sebelum ada yang menyadarinya. Agen pembaca berita berhadapan dengan konten pihak luar yang dapat memuat instruksi, sehingga memberinya kemampuan bertindak berarti membiarkan penulis berita menentukan tindakan sistem Anda. Aturan ambang yang tidak berisik: tetapkan ambang berdasarkan besaran perubahan yang benar-benar mengubah keputusan pembaca, batasi jumlah peringatan per hari, gabungkan peringatan sejenis, dan laporkan kegagalan pengambilan data lewat jalur terpisah.

Bab ini membahas rekayasa sistem informasi dan bukan nasihat keuangan. Seluruh angka adalah data sintetis penyusun.

USE CASE BISNIS

Bab 59 Use case D — Back-office dokumen dan penagihan

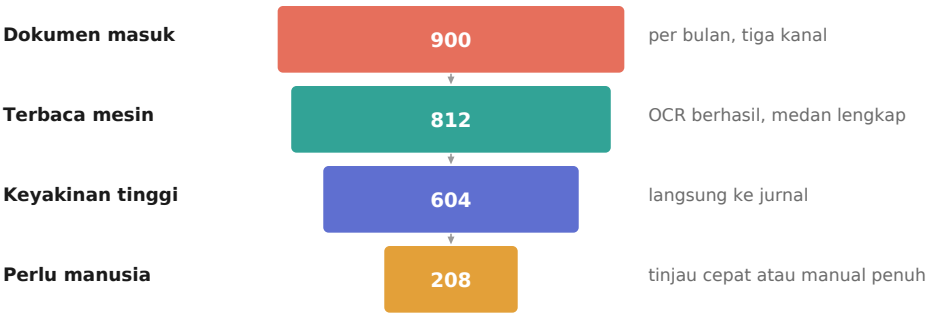
Sembilan ratus dokumen masuk setiap bulan: faktur pemasok, surat jalan, bukti pembayaran. Mereka datang lewat surel, WhatsApp, dan folder pindaian. Dua orang memasukkannya ke sistem secara manual, dan keduanya sudah lama bosan.

Tujuan belajar

Merancang alur ekstraksi dokumen dengan skor keyakinan sebagai gerbang rute, menetapkan ambang yang masuk akal, dan mengukur beban kerja yang tersisa. Pada akhir bab, pembaca dapat mengurangi pekerjaan manual tanpa memindahkan risiko ke akuntansi.

Skor keyakinan sebagai gerbang

Kesalahan paling umum pada use case ini adalah memperlakukan keluaran model sebagai benar atau salah. Yang berguna adalah tingkat ketiga: keluaran yang mungkin benar tetapi tidak cukup pasti untuk diteruskan tanpa mata manusia.



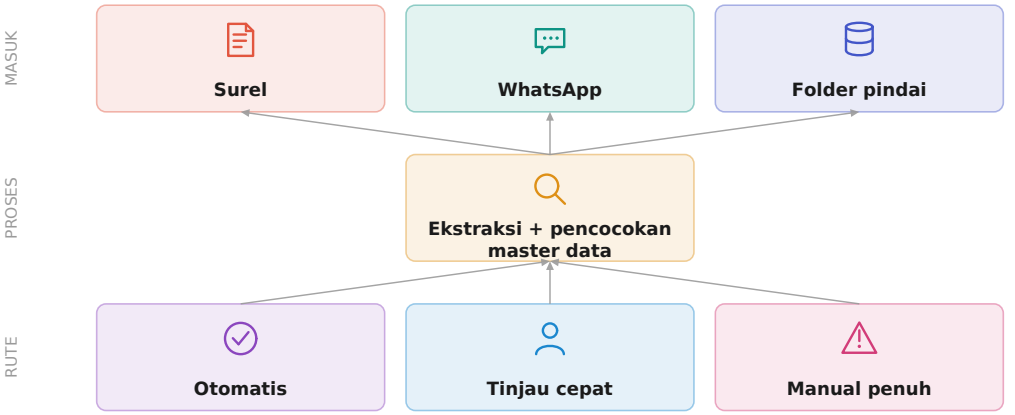
Gambar 59.1 — Corong bulanan sintetis. Angka yang menentukan bukan 900 melainkan 208: itulah beban yang tersisa untuk tim Anda.

Rute	Syarat	Perlakuan
Otomatis	Seluruh medan wajib terbaca dan cocok master data	Masuk sistem, sampel diperiksa mingguan
Tinjau cepat	Medan lengkap tetapi ada yang tidak cocok	Manusia mengonfirmasi dalam satu layar
Manual penuh	Medan wajib hilang atau dokumen tidak terbaca	Dikerjakan seperti sebelumnya

AMBANG ADALAH KEPUTUSAN BISNIS
Ambang antara ketiga rute adalah keputusan bisnis, bukan keputusan model. Ia ditetapkan berdasarkan biaya kesalahan di organisasi Anda, lalu disesuaikan dengan bukti — bukan diambil dari nilai bawaan siapa pun.

BAB 59 / ARSITEKTUR

Bentuk sistemnya



Gambar 59.2 — Pencocokan terhadap master data adalah yang mengubah tebakan model menjadi keyakinan yang dapat diukur.

Lapis tengah layak diperhatikan. Ekstraksi saja menghasilkan angka yang terlihat benar; pencocokan terhadap master data — daftar pemasok, nomor pesanan, kode produk — yang membuktikan apakah angka itu masuk akal. Tanpa pencocokan, skor keyakinan hanyalah rasa percaya diri model.

Tiga berkas keluaran

Berkas	Isinya	Pembacanya
jurnal.csv	Baris siap impor ke sistem akuntansi	Sistem
tinjau.csv	Baris beserta medan yang meragukan	Tim back-office
tolak.log	Dokumen yang tidak dapat diproses beserta alasannya	Pemilik proses

Berkas ketiga sering dilupakan dan paling berguna. Ia adalah bahan perbaikan: pola yang berulang di dalamnya menunjukkan pemasok mana yang formatnya perlu ditangani khusus, atau kanal mana yang kualitas pindaiannya buruk.

BAB 59 / KEAMANAN

Dokumen adalah konten tidak tepercaya

Dokumen dari pemasok adalah konten yang ditulis pihak luar. Bab 51 berlaku penuh di sini: agen yang membacanya harus berjalan tersandbox, tanpa exec, tanpa tool yang mengubah keadaan, dan tanpa akses tulis ke workspace.

```
{
  agents: { entries: { dokumen: {
    tools: { deny: ["exec", "process"] },
    sandbox: {
      mode: "all", scope: "session",
      workspaceAccess: "none",
      docker: { network: "none", readOnlyRoot: true, capDrop: ["ALL"] },
    },
  } } },
}
```

Agen pemroses dokumen dengan postur paling ketat dari Bab 52.

Data sensitif di dalam dokumen juga layak dipikirkan. Nomor rekening dan identitas dapat disunting dari hasil tool sebelum disimpan lewat mekanisme di Bab 40, sehingga ia tidak pernah masuk ke transkrip sesi.

BAB 59 / PRAKTIKUM

Menetapkan ambang rute dengan bukti

Praktikum ini menetapkan ambang antara tiga rute dokumen berdasarkan pengukuran, bukan berdasarkan nilai bawaan siapa pun. Ambang adalah keputusan bisnis.

Langkah 1 — jalankan fase bayangan pada dokumen nyata

Proses seratus dokumen nyata tanpa mengirim apa pun ke sistem akuntansi. Untuk tiap dokumen, catat skor keyakinan dan apakah hasil ekstraksinya benar menurut pemeriksaan manusia.

Langkah 2 — susun tabel kalibrasi

Rentang skor	Jumlah dokumen	Yang ternyata salah	Tingkat kesalahan
Sangat tinggi			
Tinggi			
Sedang			
Rendah			

Tabel ini sengaja kosong. Mengisinya dengan data Anda sendiri adalah satu-satunya cara menetapkan ambang yang bermakna, karena tingkat kesalahan bergantung pada kualitas dokumen dan kelengkapan master data Anda.

Langkah 3 — tetapkan ambang dari biaya kesalahan

```
# Pertanyaan yang menentukan ambang:
Berapa biaya satu baris salah yang masuk jurnal?
Berapa biaya satu dokumen yang tidak perlu ditinjau manusia?
Ambang yang tepat adalah titik di mana keduanya seimbang.
```

Ambang bukan angka teknis; ia hasil perbandingan dua biaya di organisasi Anda.

Langkah 4 — pasang pencocokan master data

Tanpa pencocokan terhadap daftar pemasok, nomor pesanan, dan kode produk, skor keyakinan hanya mencerminkan rasa percaya diri model terhadap pembacaannya sendiri. Pencocokan yang mengubahnya menjadi sesuatu yang dapat diverifikasi.

BAB 59 / STUDI KASUS

Ambang yang disalin dari presentasi vendor

Sebuah tim menetapkan ambang otomatis pada angka yang mereka lihat di sebuah presentasi. Angka itu tampak wajar dan datang dari sumber yang terlihat kredibel.

Dokumen mereka berbeda: sebagian besar berupa pindaian dari mesin faks pemasok kecil, dengan kualitas yang jauh di bawah contoh pada presentasi itu. Pada ambang tersebut, sekitar satu dari dua belas dokumen yang lolos ke rute otomatis ternyata salah.

Gambar 59.3 — Angka sintetis. Ambang yang lebih konservatif mengurangi otomasi seperempat dan mengurangi kesalahan lima per enam.

Setelah kalibrasi sendiri, proporsi otomatis turun tetapi kesalahan yang masuk jurnal turun jauh lebih tajam. Pekerjaan pembersihan yang hilang jauh melebihi pekerjaan tinjauan yang bertambah.

ANGKA ORANG LAIN BUKAN ANGKA ANDA

Ambang yang benar untuk organisasi lain hampir pasti salah untuk Anda, karena ia bergantung pada kualitas dokumen dan kelengkapan master data. Kalibrasi seratus dokumen memakan satu hari dan berlaku bertahun-tahun.

BAB 59 / LATIHAN

Latihan mandiri

1. Jalankan fase bayangan pada seratus dokumen nyata dan isi tabel kalibrasi.
2. Hitung dua biaya pada praktikum langkah tiga untuk organisasi Anda.
3. Tetapkan ambang dari perbandingan kedua biaya itu, lalu tuliskan alasannya.
4. Periksa kelengkapan master data Anda; perbaiki di sana sebelum menyesuaikan ambang.

BAB 59 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Angka salah lolos ke jurnal	Ambang terlalu longgar, atau pencocokan master data belum ada.
Rute tinjau cepat membengkak	Master data tidak lengkap. Perbaiki sumbernya, bukan ambangnya.
Dokumen pemasok tertentu selalu gagal	Periksa tolak.log; polanya biasanya terlihat jelas di sana.
Data pribadi tersimpan di transkrip	Sunting hasil tool sebelum penyimpanan.
Tim tidak percaya pada hasil otomatis	Belum ada pemeriksaan sampel. Jalankan dan tunjukkan hasilnya.

Tiga pertanyaan review

1. Mengapa pencocokan master data mengubah sifat skor keyakinan?
2. Mengapa rute ketiga tetap dipertahankan meskipun ia berarti pekerjaan manual?
3. Rancang cara menaikkan ambang otomatis secara bertahap dengan bukti.

Kunci pembahasan

Tanpa pencocokan, skor hanya mencerminkan seberapa yakin model terhadap pembacaannya sendiri; dengan pencocokan, skor mencerminkan kesesuaian terhadap data yang sudah diketahui benar, yang dapat diverifikasi. Rute manual dipertahankan karena memaksa setiap dokumen melewati jalur otomatis akan memindahkan dokumen yang buruk ke dalam sistem, dan membersihkannya jauh lebih mahal daripada mengetiknya. Untuk menaikkan ambang: periksa sampel acak dari rute otomatis setiap minggu, catat tingkat kesalahannya, dan naikkan cakupan hanya setelah beberapa minggu berturut-turut menunjukkan tingkat kesalahan di bawah batas yang Anda tetapkan di muka.

Arsitektur mengikuti kemampuan yang didokumentasikan; seluruh angka adalah data sintetis penyusun.

USE CASE BISNIS

Bab 60 Use case E — Asisten penjualan dan CRM

Tim penjualan menghabiskan waktu untuk pekerjaan yang bukan menjual: mencatat hasil pertemuan, mencari riwayat pelanggan sebelum menelepon, menyusun tindak lanjut. Use case ini menyasar pekerjaan itu — dan sengaja tidak menyasar percakapan dengan pelanggan.

Tujuan belajar

Merancang asisten yang menyiapkan dan mencatat tanpa menggantikan penjual, menjaga kualitas data CRM, dan menghindari pola yang merusak hubungan pelanggan. Pada akhir bab, pembaca dapat mengembalikan waktu penjual tanpa mengotomatiskan penjualannya.

Apa yang dikerjakan dan tidak

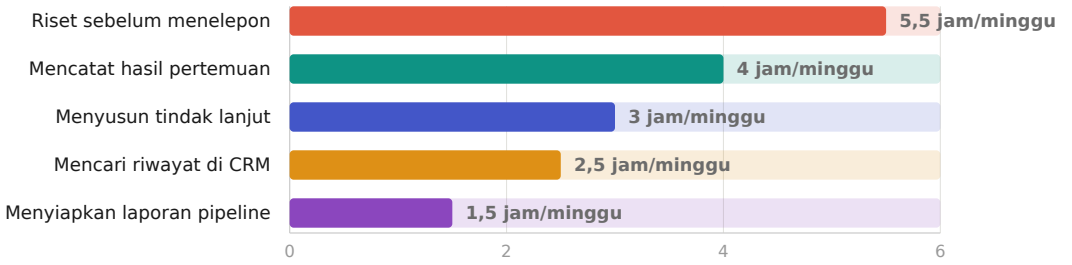
DIKERJAKAN AGEN		TETAP MANUSIA
Ya	**Riset sebelum pertemuan**	—
Ya	**Ringkasan catatan pertemuan**	—
Ya, sebagai draf	**Draf tindak lanjut**	Dikirim manusia
Diusulkan	**Pembaruan CRM**	Disetujui
—	**Percakapan dengan pelanggan**	Selalu
—	**Negosiasi harga**	Selalu

Gambar 60.1 — Batasnya jelas: agen menyiapkan dan merapikan, manusia berbicara dan memutuskan.

Dua baris terakhir bukan kehati-hatian berlebihan. Hubungan penjualan dibangun di atas kepercayaan pribadi, dan pelanggan yang menyadari ia berbicara dengan mesin tanpa diberi tahu akan mengingatnya lebih lama daripada efisiensi apa pun yang Anda peroleh.

BAB 60 / NILAI

Dari mana waktunya kembali



Gambar 60.2 — Waktu non-penjualan per penjual per minggu, sintetis. Total sekitar enam belas jam — hampir separuh minggu kerja.

Angka seperti ini biasanya mengejutkan manajemen dan tidak mengejutkan penjual. Nilai use case ini bukan menggantikan penjualan, melainkan mengembalikan sebagian dari enam belas jam itu ke pekerjaan yang benar-benar menghasilkan.

Arsitektur

Gambar 60.3 — Agen ini tidak pernah terhubung ke kanal pelanggan. Pemisahan itu sekaligus kontrol keamanan dan kontrol hubungan.

BAB 60 / KUALITAS DATA

Masalah yang sebenarnya: data CRM

Sebagian besar proyek asisten penjualan gagal bukan karena agennya, melainkan karena data CRM-nya buruk. Agen yang membaca catatan yang tidak lengkap akan menghasilkan ringkasan yang tidak lengkap, dan penjual akan berhenti memercayainya dalam dua minggu.

- ✓ Medan wajib CRM benar-benar diisi, bukan sekadar ada
- ✓ Agen mengusulkan pembaruan CRM setelah setiap pertemuan
- ✓ Usulan ditinjau penjual dalam satu ketukan, bukan formulir panjang
- ✓ Kualitas data diukur dan dilaporkan, bukan diasumsikan
- ⚠ Memakai agen untuk menutupi data CRM yang berantakan

Gambar 60.4 — Lima butir. Butir terakhir adalah harapan yang selalu gagal, dan biasanya menjadi alasan proyek dihentikan.

Justru di sinilah nilai tak terduga muncul: agen yang mengusulkan pembaruan CRM setelah setiap pertemuan memperbaiki kualitas data secara bertahap, karena mengisi CRM berubah dari pekerjaan terpisah menjadi satu ketukan konfirmasi.

BAB 60 / PRAKTIKUM

Mengembalikan waktu penjual tanpa menggantikan penjualan

Praktikum ini mengukur ke mana waktu penjual pergi, lalu menyasar bagian yang bukan menjual. Mengukur lebih dulu mencegah Anda mengotomatiskan hal yang salah.

Langkah 1 — ukur satu minggu

Minta tiga penjual mencatat waktu untuk lima kategori: riset sebelum menelepon, mencatat hasil pertemuan, menyusun tindak lanjut, mencari riwayat di CRM, dan menyiapkan laporan. Angka nyata hampir selalu mengejutkan manajemen dan tidak mengejutkan penjual.

Langkah 2 — periksa kualitas data CRM sebelum apa pun

Pemeriksaan	Cara	Bila gagal
Kelengkapan medan wajib	Hitung persentase terisi	Perbaiki data dulu; agen tidak akan menutupinya
Kemutakhiran	Periksa tanggal pembaruan terakhir	Tetapkan proses pembaruan
Konsistensi	Bandingkan beberapa catatan serupa	Sepakati format sebelum melanjutkan

Langkah 3 — batasi agen pada penyiapan dan pencatatan

```
# Yang dikerjakan agen
- riset sebelum pertemuan
- ringkasan catatan pertemuan
- draf tindak lanjut untuk ditinjau penjual
- usulan pembaruan CRM untuk dikonfirmasi

# Yang tetap manusia
- seluruh percakapan dengan pelanggan
- negosiasi harga dan komitmen apa pun
```

Agen ini tidak pernah disambungkan ke kanal pelanggan.

Langkah 4 — buat konfirmasi CRM menjadi satu ketukan

Usulan pembaruan yang menuntut membuka formulir panjang akan diabaikan. Usulan yang dapat dikonfirmasi dengan satu ketukan akan dipakai, dan di situlah kualitas data mulai membaik dengan sendirinya.

BAB 60 / STUDI KASUS

Asisten yang ditinggalkan karena datanya buruk

Sebuah tim penjualan memasang asisten riset. Pada minggu pertama, adopsinya tinggi: seluruh penjual memakainya sebelum menelepon. Ringkasannya cepat dan formatnya rapi.

Pada minggu ketiga, pemakaian turun tajam. Alasannya sama dari setiap penjual: ringkasan sering melewatkan hal penting, atau menyebutkan informasi yang sudah tidak berlaku. Mereka kembali membuka CRM sendiri, karena mereka tahu bagian mana yang dapat dipercaya dan bagian mana yang tidak.

Yang disalahkan	Penyebab sesungguhnya
Modelnya kurang pintar	Data masukan tidak lengkap
Promptnya kurang baik	Medan wajib CRM kosong pada 40 persen catatan
Agen tidak cocok untuk penjualan	Agen mencerminkan data yang ia baca
Proyek gagal	Proyek menyasar gejala, bukan penyebab

Setelah mereka memakai agen untuk mengusulkan pembaruan CRM setelah setiap pertemuan — satu ketukan konfirmasi — kelengkapan data naik bertahap. Enam minggu kemudian, kualitas ringkasan membaik dengan sendirinya dan adopsi kembali naik.

AGEN MENCERMINKAN DATANYA

Asisten penjualan tidak dapat menutupi data CRM yang buruk; ia mencerminkannya. Bila kualitas data adalah masalahnya, sasarlah itu lebih dulu — dan agen justru dapat menjadi alat untuk memperbaikinya.

BAB 60 / LATIHAN

Latihan mandiri

1. Ukur waktu non-penjualan tiga penjual Anda selama satu minggu.
2. Jalankan ketiga pemeriksaan kualitas data CRM dan catat angkanya.
3. Rancang alur usulan pembaruan CRM yang dapat dikonfirmasi dengan satu ketukan.
4. Tetapkan metrik keberhasilan yang sulit dimanipulasi, dan hindari menghitung jumlah pesan ke agen.

BAB 60 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Penjual berhenti memakai setelah dua minggu	Data CRM buruk, atau ringkasan tidak cukup akurat untuk dipercaya.
Ringkasan melewatkan hal penting	Sumber tidak lengkap. Periksa apakah riwayat surel ikut tersambung.
Usulan CRM diabaikan	Peninjauannya terlalu merepotkan. Ringkas menjadi satu ketukan.
Ada permintaan agen membalas pelanggan langsung	Tolak; itu use case yang berbeda dengan risiko hubungan yang berbeda.
Laporan pipeline tidak cocok dengan kenyataan	Kualitas data. Ukur dan laporkan, jangan tutupi dengan agen.

Tiga pertanyaan review

1. Mengapa agen penjualan sengaja tidak disambungkan ke kanal pelanggan?
2. Bagaimana asisten ini dapat memperbaiki kualitas data CRM alih-alih sekadar memakainya?
3. Rancang cara mengukur apakah proyek ini berhasil, tanpa memakai metrik yang mudah dimanipulasi.

Kunci pembahasan

Pemisahan itu melindungi dua hal sekaligus: hubungan pelanggan yang bergantung pada kepercayaan pribadi, dan keamanan, karena agen ini memiliki akses luas ke data pelanggan yang tidak seharusnya dapat dijangkau dari kanal eksternal. Kualitas data membaik ketika pengisian CRM berubah dari tugas administratif terpisah menjadi konfirmasi satu ketukan atas usulan yang sudah disusun agen. Metrik yang sulit dimanipulasi: waktu yang dilaporkan penjual untuk pekerjaan non-penjualan, kelengkapan medan wajib CRM yang diukur otomatis, dan proporsi pertemuan yang punya catatan dalam dua puluh empat jam — bukan jumlah pesan yang dikirim ke agen.

Arsitektur mengikuti kemampuan yang didokumentasikan; seluruh angka adalah data sintetis penyusun.

USE CASE BISNIS

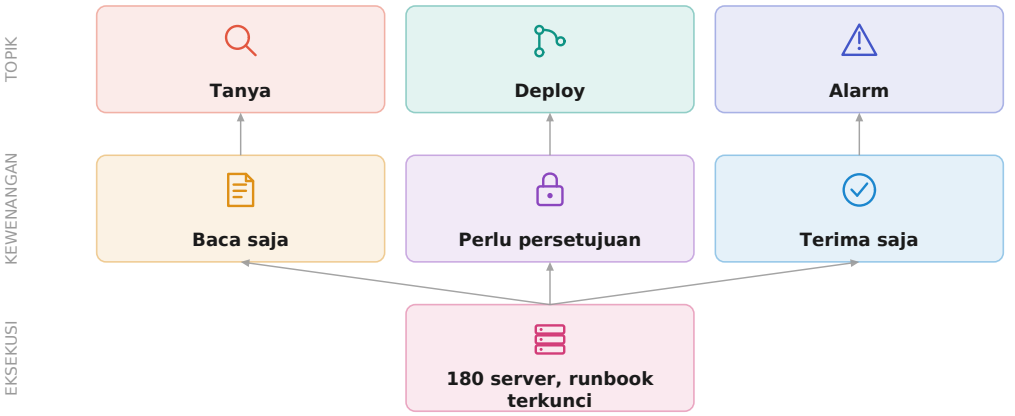
Bab 61 Use case F — Operasi internal tim IT

Tim tujuh orang menjaga seratus delapan puluh server. Pertanyaan status, permintaan restart, dan penelusuran insiden datang sepanjang hari lewat Telegram. Use case ini berbeda dari lima sebelumnya: di sini agen benar-benar menjalankan perintah.

Tujuan belajar

Memakai topik forum sebagai tingkat kewenangan, menyusun runbook yang dapat dieksekusi, dan menjaga jejak audit. Pada akhir bab, pembaca dapat memberi agen kemampuan eksekusi dengan batas yang dapat dipertanggungjawabkan.

Tiga topik, tiga kewenangan



Gambar 61.1 — Perpindahan tingkat kewenangan berarti pindah ruang, bukan menambah argumen perintah.

Rancangan ini memanfaatkan pewarisan setelan topik dari Bab 29. Topik Tanya mewarisi setelan grup dengan skill baca saja; topik Deploy menimpanya dengan daftar izin sempit dan persetujuan wajib; topik Alarm hanya menerima dan tidak memiliki tool eksekusi sama sekali.

```
{
  channels: { telegram: { groups: {
    "-100xxxxxxxxx": {
      allowFrom: [ /* 7 anggota tim */ ],
      topics: {
        "11": { skills: ["status"] },
        "12": { skills: ["runbook"], allowFrom: [ /* 3 senior */ ] },
        "13": { skills: [], requireMention: false },
```

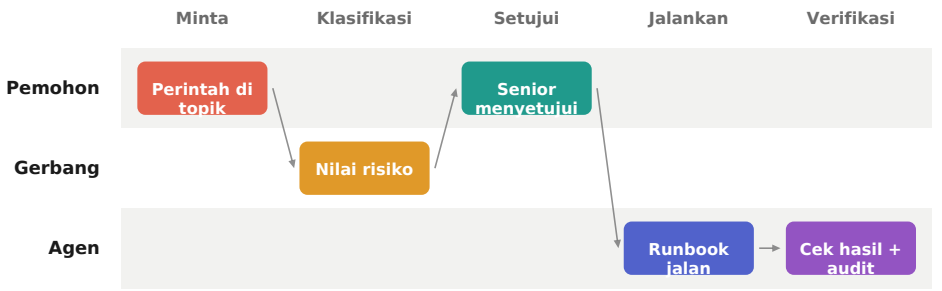
```
      },
    },
  } } },
}
```

Topik Deploy memakai daftar izin yang lebih sempit daripada grupnya.

BAB 61 / RUNBOOK

Runbook yang dapat dieksekusi

Runbook yang berguna untuk agen berbeda dari runbook untuk manusia. Ia harus menyebut kondisi berhenti, bukan hanya langkah; harus menyebut cara memverifikasi keberhasilan; dan harus menyebut apa yang dilakukan ketika verifikasi gagal.



Gambar 61.2 — Lima tahap. Tahap keempat tidak pernah dimulai tanpa jejak dari tahap ketiga.

```
## Runbook: restart layanan aplikasi
**Prasyarat:** Layanan terdaftar di inventaris; bukan jam puncak
**Berhenti bila:** Ada insiden terbuka pada layanan yang sama
**Langkah:** ...
**Verifikasi:** Endpoint kesehatan mengembalikan 200 dalam 60 detik
**Bila verifikasi gagal:** Jangan ulangi. Panggil penanggung jawab.
```

Empat medan di luar langkah itulah yang membedakan runbook agen dari catatan biasa.

JANGAN PERNAH MENGULANG OTOMATIS

Baris terakhir penting. Agen yang mengulang langkah ketika verifikasi gagal adalah cara tercepat mengubah gangguan kecil menjadi insiden besar. Tuliskan larangan mengulang secara eksplisit.

BAB 61 / HASIL

Mengukur dan mengaudit

180 server	7 orang	3 orang	100 %
Dalam cakupan	Tim operasi	Berwenang menyetujui deploy	Tindakan tercatat di audit

Gambar 61.3 — Angka cakupan sintetis. Metrik keempat tidak boleh kurang dari seratus persen; bila kurang, ada jalur yang tidak tercatat.

Jejak audit OpenClaw mencatat metadata beserta identitas jalannya dan tanda terima keputusan, sebagaimana dibahas di Bab 54. Untuk tim operasi, ini yang mengubah pertanyaan “siapa merestart layanan itu” dari perdebatan menjadi kueri.

BAB 61 / PRAKTIKUM

Menulis runbook yang aman dijalankan agen

Runbook untuk agen berbeda dari runbook untuk manusia. Manusia berhenti ketika sesuatu terasa aneh; agen tidak, kecuali Anda menuliskan kapan ia harus berhenti.

Langkah 1 — tulis empat medan di luar langkah

```
## Runbook: <nama>
**Prasyarat:** keadaan yang harus benar sebelum memulai
**Berhenti bila:** keadaan yang membatalkan seluruh prosedur
**Verifikasi:** cara membuktikan langkah berhasil
**Bila verifikasi gagal:** JANGAN ulangi. Panggil penanggung jawab.
```

Medan keempat mencegah gangguan kecil berlipat menjadi insiden besar.

Langkah 2 — petakan topik ke kewenangan

Topik	Skill	Daftar izin	Persetujuan
Tanya	Baca saja	Seluruh tim	Tidak perlu
Deploy	Runbook	Senior saja	Wajib
Alarm	Kosong	Seluruh tim	Tidak ada eksekusi

Langkah 3 — uji bahwa topik benar-benar membatasi

Minta seseorang di luar daftar senior menjalankan perintah deploy di topik Deploy. Ia harus ditolak meski ia ada di daftar izin grup. Bila tidak, penimpaan per topik Anda belum bekerja.

Langkah 4 — buktikan tidak ada jalur yang lolos audit

```
openclaw tasks audit
# jalankan sejumlah perintah uji lewat tiap jalur,
# lalu cocokkan jumlahnya dengan catatan audit
```

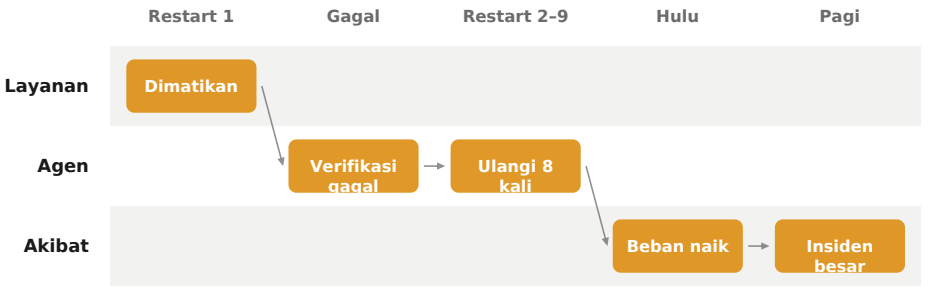
Metrik ini tidak boleh kurang dari seratus persen; bila kurang, ada jalur yang tidak tercatat.

BAB 61 / STUDI KASUS

Runbook yang mengulang sampai merusak

Sebuah tim menulis runbook restart layanan dengan langkah yang jelas dan verifikasi yang benar: endpoint kesehatan harus mengembalikan status sehat dalam enam puluh detik. Runbook itu tidak menyebut apa yang dilakukan bila verifikasi gagal.

Suatu malam, layanan gagal naik karena masalah basis data di hulu. Verifikasi gagal. Agen menyimpulkan bahwa langkahnya belum berhasil dan mengulanginya. Ia mengulang sembilan kali dalam dua belas menit.



Gambar 61.4 — Satu gangguan hulu yang seharusnya kecil, diperbesar oleh pengulangan otomatis.

Restart berulang menambah beban pada basis data yang sedang bermasalah, dan yang semula gangguan satu layanan menjadi insiden yang menyentuh beberapa layanan lain. Penanggung jawab baru dipanggil pagi harinya.

> **KALIMAT YANG WAJIB ADA DI SETIAP RUNBOOK**
> Tuliskan larangan mengulang secara eksplisit di setiap runbook. Agen yang mengulang langkah ketika verifikasi gagal adalah cara tercepat mengubah gangguan kecil menjadi insiden besar.

BAB 61 / LATIHAN

Latihan mandiri

1. Tulis satu runbook lengkap dengan keempat medan, termasuk larangan mengulang.
2. Petakan topik Anda ke kewenangan dan uji bahwa penimpaan per topik benar-benar membatasi.
3. Jalankan uji cakupan audit dan pastikan tidak ada jalur eksekusi yang tidak tercatat.
4. Periksa runbook yang sudah ada di organisasi Anda dan tambahkan medan yang hilang.

BAB 61 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Siapa pun di tim bisa men-deploy	Topik Deploy memakai daftar izin grup. Persempit per topik.
Agen mengulang perintah yang gagal	Runbook tidak melarang pengulangan. Tuliskan eksplisit.
Tidak bisa menjawab siapa melakukan apa	Ada jalur yang melewati audit. Temukan dan tutup.
Persetujuan menjadi formalitas	Terlalu sering. Perluas allowlist perintah rutin yang aman.
Alarm bercampur dengan percakapan	Pisahkan ke topik sendiri tanpa tool eksekusi.

Tiga pertanyaan review

1. Mengapa topik lebih baik daripada argumen perintah untuk membedakan kewenangan?
2. Mengapa runbook agen harus menyebut kondisi berhenti dan larangan mengulang?
3. Rancang cara membuktikan bahwa tidak ada jalur eksekusi yang melewati audit.

Kunci pembahasan

Topik membuat kewenangan menjadi tempat, sehingga seseorang harus berpindah ruang secara sadar — sementara argumen perintah dapat diketik siapa saja yang tahu sintaksisnya, dan tidak meninggalkan sinyal sosial apa pun. Kondisi berhenti mencegah agen bertindak pada keadaan yang tidak diantisipasi penulis runbook, dan larangan mengulang mencegah kegagalan tunggal berlipat menjadi insiden. Untuk membuktikan tidak ada jalur yang lolos audit: daftar seluruh tool yang dapat mengeksekusi, uji tiap jalur di lingkungan uji, lalu cocokkan jumlah percobaan dengan jumlah catatan audit.

Arsitektur mengikuti kemampuan yang didokumentasikan; seluruh angka adalah data sintetis penyusun.

USE CASE BISNIS

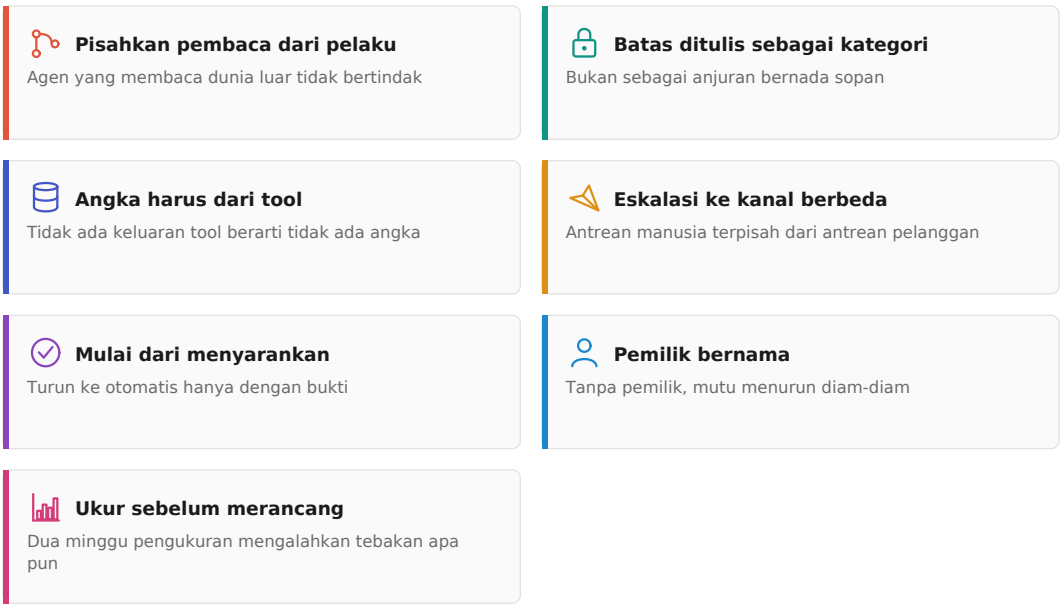
Bab 62 Pola yang berulang di keenam use case

Enam use case, enam domain yang berbeda, dan sejumlah keputusan yang sama persis. Bab penutup Bagian IX mengangkat pola itu supaya Anda dapat memakainya pada use case yang tidak ada di dalam buku ini.

Tujuan belajar

Mengenali tujuh pola yang berulang, menerapkannya pada kasus baru, dan mengenali tanda bahwa sebuah rancangan menyimpang darinya. Pada akhir bab, pembaca dapat merancang use case sendiri tanpa memulai dari nol.

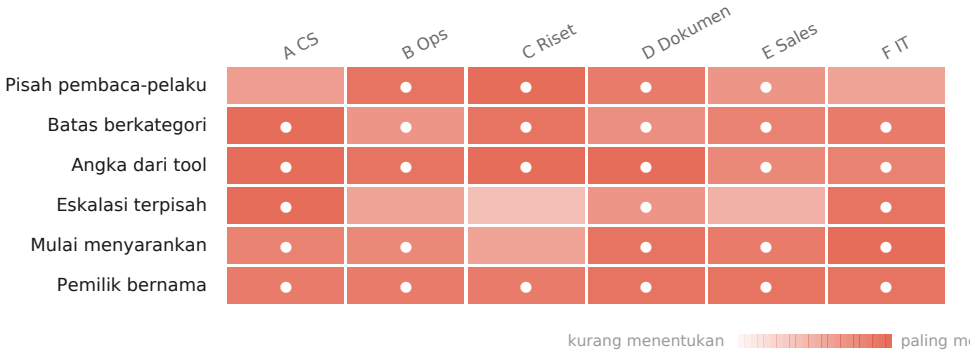
Tujuh pola



Gambar 62.1 — Tujuh pola yang muncul di keenam use case. Rancangan yang melanggar salah satunya biasanya gagal pada titik itu juga.

BAB 62 / PEMETAAN

Di mana tiap pola muncul



Gambar 62.2 — Intensitas tiap pola per use case. Baris ketiga dan keenam gelap di hampir seluruh kolom; keduanya berlaku universal.

Dua baris yang gelap merata layak diperhatikan. Aturan bahwa angka harus berasal dari tool berlaku di mana pun agen menyebut angka, dan kebutuhan pemilik bernama berlaku di mana pun sebuah proses harus bertahan lebih dari satu kuartal.

BAB 62 / MENERAPKAN

Menerapkan pada use case baru

Gambar 62.3 — Lima langkah dari ide ke penerapan. Langkah pertama adalah yang paling sering dilewati dan paling mahal untuk dilewati.

Dua minggu pengukuran — berapa banyak permintaan, jenis apa saja, berapa lama ditangani sekarang, berapa yang datang di luar jam kerja — akan mengubah rancangan Anda lebih banyak daripada membaca seluruh buku ini. Tanpa itu, Anda merancang untuk masalah yang dibayangkan, bukan yang dimiliki.

Tanda bahwa rancangan menyimpang

- ⚠ Satu agen melayani kanal pelanggan sekaligus sistem internal
- ⚠ Batas ditulis sebagai 'sebaiknya hindari' alih-alih kategori tegas
- ⚠ Tidak ada seorang pun yang namanya tertulis sebagai pemilik
- ⚠ Rencana langsung menuju otomasi penuh pada bulan pertama
- ⚠ Angka target ditetapkan tanpa pengukuran awal

Gambar 62.4 — Lima tanda bahaya. Menemukan salah satunya dalam rencana Anda adalah kesempatan, bukan kegagalan.

BAB 62 / PRAKTIKUM

Menerapkan tujuh pola pada ide Anda sendiri

Praktikum ini mengubah tujuh pola menjadi daftar periksa yang dapat Anda jalankan pada rancangan apa pun, termasuk yang tidak ada di dalam buku ini.

Langkah 1 — tuliskan rancangan Anda dalam enam baris

```
Tujuan   :
Pembaca  : agen mana yang membaca konten dari luar
Pelaku   : agen mana yang memiliki tool yang mengubah keadaan
Batas    : kategori yang diserahkan ke manusia
Sumber   : dari mana angka berasal
Pemilik  : nama orang, bukan nama tim
```

Enam baris ini memetakan langsung ke enam dari tujuh pola.

Langkah 2 — periksa tiap pola terhadap rancangan itu

Pola	Pertanyaan periksa
Pisahkan pembaca dari pelaku	Apakah baris Pembaca dan Pelaku berbeda?
Batas berkategori	Apakah baris Batas dapat diperiksa mesin?
Angka dari tool	Apakah baris Sumber menyebut tool, bukan pengetahuan model?
Eskalasi terpisah	Apakah antrean manusia terpisah dari antrean pelanggan?
Mulai dari menyarankan	Apakah tahap pertama melibatkan manusia?
Pemilik bernama	Apakah baris Pemilik berisi nama orang?
Ukur sebelum merancang	Apakah Anda sudah punya angka dua minggu?

Langkah 3 — tandai pola yang gagal dan perbaiki di sana

Rancangan yang melanggar sebuah pola biasanya gagal pada titik itu juga. Memperbaiki

sekarang memakan menit; memperbaikinya setelah peluncuran memakan minggu.

Langkah 4 — simpan sebagai templat

Enam baris ini akan Anda pakai lagi untuk use case berikutnya. Menyimpannya sebagai templat organisasi membuat percakapan rancangan berikutnya dimulai dari tempat yang jauh lebih maju.

BAB 62 / STUDI KASUS

Use case ketujuh yang dirancang dalam satu jam

Sebuah perusahaan telah menjalankan enam use case selama setahun. Ketika permintaan ketujuh datang, tim yang menanganinya sudah berganti separuh; hanya dua orang yang terlibat sejak awal.

Alih-alih memulai dari nol, mereka memakai templat enam baris dan daftar tujuh pola. Rancangan awal selesai dalam satu jam, dan tiga masalah yang biasanya baru muncul saat pilot tertangkap di ruang rapat.

USE CASE PERTAMA		USE CASE KETUJUH
Tiga minggu	**Waktu rancangan**	Satu jam
Pilot	**Masalah ditemukan saat**	Rancangan
Terlewat	**Pemisahan pembaca-pelaku**	Baris kedua dan ketiga
Ditentukan belakangan	**Pemilik**	Ditulis di awal
Tidak ada	**Pengukuran**	Sudah berjalan

Gambar 62.5 — Selisih yang berasal dari pola yang tertulis, bukan dari orang yang lebih berpengalaman — separuh tim itu baru.

Yang membuat perbedaannya bukan pengalaman individu, melainkan pengetahuan yang sudah berpindah dari kepala orang ke dokumen. Itulah yang dimaksud tata kelola di Bab 66.

MENGAPA MENULISKANNYA LAYAK

Pola yang tertulis bertahan melewati pergantian orang; pengalaman yang hanya ada di kepala tidak. Menuliskan tujuh pola Anda sendiri adalah investasi yang membayar pada use case berikutnya.

BAB 62 / LATIHAN

Latihan mandiri

1. Tuliskan rancangan use case Anda dalam enam baris, lalu periksa terhadap tujuh pola.
2. Perbaiki setiap pola yang gagal sebelum melanjutkan ke konfigurasi apa pun.
3. Simpan enam baris itu sebagai templat organisasi Anda.
4. Tambahkan pola kedelapan yang berasal dari pengalaman organisasi Anda sendiri.

BAB 62 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Rancangan terasa rumit sekali	Biasanya karena satu agen diminta melakukan terlalu banyak. Pisahkan.
Sulit menentukan batas	Tulis sebagai kategori yang dapat diperiksa, bukan sebagai nuansa.
Target tidak realistis	Ditetapkan tanpa pengukuran awal. Ukur dulu dua minggu.
Proyek kehilangan momentum	Tidak ada pemilik bernama, atau nilainya tidak pernah ditunjukkan.
Use case baru tidak cocok pola mana pun	Periksa empat gerbang Bab 55; kemungkinan ia memang belum layak.

Tiga pertanyaan review

1. Mengapa aturan 'angka harus dari tool' berlaku hampir universal?
2. Mengapa pola 'mulai dari menyarankan' sering ditolak manajemen, dan bagaimana menjawabnya?
3. Ambil satu ide use case Anda sendiri dan jalankan lima langkah pada Gambar 62.3.

Kunci pembahasan

Angka adalah tempat kesalahan model paling merusak kepercayaan dan paling mudah diperiksa, sehingga aturan itu memberi perlindungan besar dengan biaya penerapan yang hampir nol. Manajemen menolak pola menyarankan karena ia terlihat tidak menghemat apa pun; jawabannya adalah menunjukkan bahwa fase menyarankan menghasilkan data kesalahan yang diperlukan untuk membenarkan otomasi penuh nanti — tanpa fase itu, otomasi penuh adalah taruhan tanpa bukti. Jawaban pertanyaan ketiga bergantung pada pembaca, tetapi hasilnya sering berupa kesadaran bahwa pengukuran dua minggu mengubah prioritasnya.

Pola pada bab ini adalah sintesis dari keenam use case di Bagian IX dan praktik penyusun.

BIAYA, SKALA, DAN TATA KELOLA

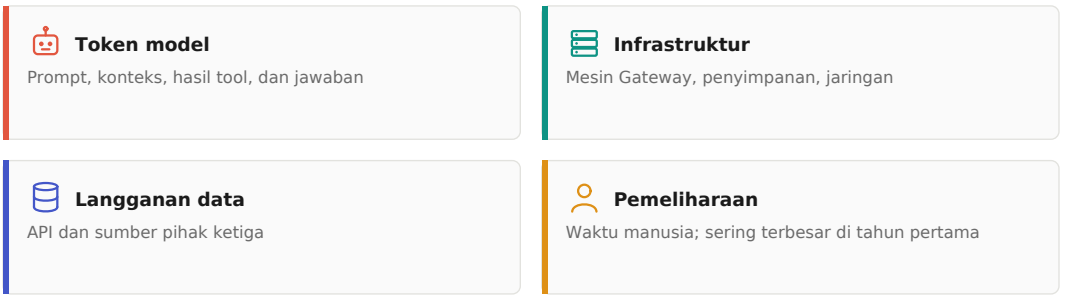
Bab 63 Model biaya token dan infrastruktur

Pertanyaan pertama dari siapa pun yang memegang anggaran adalah berapa biayanya. Bab ini memberi model perhitungan yang dapat Anda jalankan sendiri, beserta dua angka yang paling sering salah diisi.

Tujuan belajar

Menghitung biaya token dari parameter yang dapat diukur, mengenali komponen biaya di luar token, dan menghindari kesalahan estimasi yang umum. Pada akhir bab, pembaca dapat menyusun anggaran yang bertahan saat diperiksa.

Empat komponen



Gambar 63.1 — Empat komponen. Komponen keempat paling sering hilang dari perhitungan dan paling sering menjadi yang terbesar.

Model yang dapat dijalankan

```
# biaya.py — estimasi biaya token bulanan per use case

def estimasi(percakapan_per_hari, hari_kerja,
             giliran_per_percakapan,
             token_masuk_per_giliran, token_keluar_per_giliran,
             harga_masuk_per_juta, harga_keluar_per_juta,
             rasio_cache_hit=0.0, diskon_cache=0.9):
    """Harga dalam satuan mata uang per juta token."""
    giliran = (percakapan_per_hari * hari_kerja
               * giliran_per_percakapan)
    masuk = giliran * token_masuk_per_giliran
    keluar = giliran * token_keluar_per_giliran

    masuk_cache = masuk * rasio_cache_hit
    masuk_penuh = masuk - masuk_cache

    biaya = (masuk_penuh / 1e6 * harga_masuk_per_juta
             + masuk_cache / 1e6 * harga_masuk_per_juta
```

```
              * (1 - diskon_cache)
            + keluar / 1e6 * harga_keluar_per_juta)

    return {
        "giliran_per_bulan": int(giliran),
        "token_masuk": int(masuk),
        "token_keluar": int(keluar),
        "biaya_bulanan": round(biaya, 2),
        "biaya_per_percakapan": round(
            biaya / (percakapan_per_hari * hari_kerja), 4),
    }
```

Ganti seluruh harga dengan tarif penyedia dan model yang benar-benar Anda pakai.

DUA ANGKA YANG MENENTUKAN SEGALANYA

Dua parameter paling menentukan sekaligus paling sering salah diisi. Yang pertama token_masuk_per_giliran: ia mencakup prompt sistem, indeks skill, skema tool, riwayat percakapan, dan hasil pemanggilan tool — bukan hanya pesan pengguna. Yang kedua rasio_cache_hit: caching prompt adalah alasan utama percakapan panjang tetap terjangkau.

Apa pun yang memutus konteks lampau, menukar toolset, atau membangun ulang prompt sistem di tengah percakapan akan membatalkan cache itu. Karena itu, keputusan desain seperti menyalakan skill baru di tengah sesi punya konsekuensi biaya yang tidak terlihat pada tagihan sampai akhir bulan.

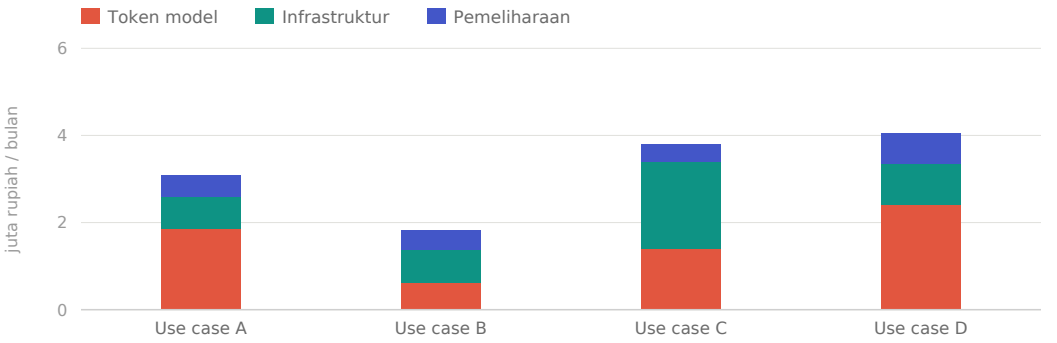
BAB 63 / MENGUKUR

Ukur, jangan menebak

Selisih antara tebakan dan pengukuran pada parameter token sering lebih dari tiga kali lipat. Jalankan tiga puluh percakapan nyata pada fase bayangan, baca angka sebenarnya dari pelacakan pemakaian atau dari jejak observability, lalu isikan ke model ini.

```
openclaw status          # ringkasan pemakaian
openclaw gateway usage-cost
```

Pemakaian dan biaya dapat dibaca langsung dari Gateway, bukan diperkirakan.



Gambar 63.2 — Susunan biaya sintetis empat use case. Token bukan selalu komponen terbesar; pada use case C, langganan data yang menang.

BAB 63 / PRAKTIKUM

Mengisi model biaya dengan angka yang diukur

Praktikum ini mengganti setiap tebakan pada model biaya dengan angka yang diukur. Selisih antara keduanya sering lebih dari tiga kali lipat.

Langkah 1 — jalankan fase bayangan tiga puluh percakapan

```
openclaw status
openclaw gateway usage-cost
```

Baca angka sebenarnya, jangan memperkirakan dari panjang pesan.

Langkah 2 — isi parameter satu per satu

Parameter	Cara mendapatkannya	Kesalahan umum
percakapan_per_hari	Hitung dari riwayat kanal	Memakai perkiraan
giliran_per_percakapan	Rata-rata dari sesi nyata	Mengira satu
token_masuk_per_giliran	Baca dari pemakaian	Menghitung pesan pengguna saja
token_keluar_per_giliran	Baca dari pemakaian	Jarang salah
rasio_cache_hit	Bandingkan giliran pertama dan berikutnya	Mengira nol

Langkah 3 — hitung tiga skenario

Hitung biaya pada volume sekarang, dua kali lipat, dan lima kali lipat. Skenario ketiga sering memperlihatkan bahwa rancangan Anda perlu diubah sebelum volumenya sampai ke sana.

Langkah 4 — tambahkan komponen di luar token

```
# Yang sering hilang dari perhitungan:
- mesin Gateway dan penyimpanan
- langganan data dan API pihak ketiga
- waktu manusia untuk meninjau dan merawat
```

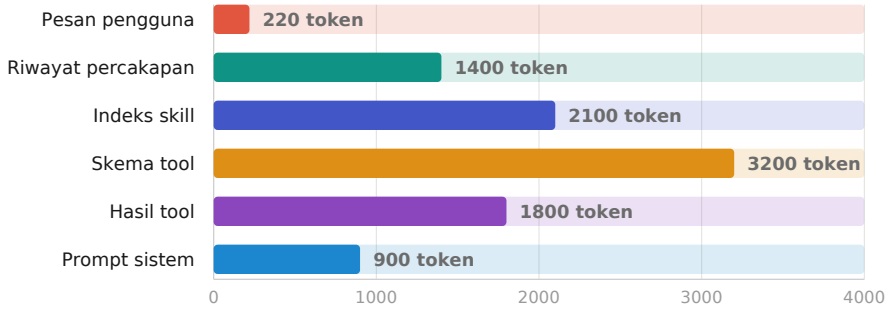
Komponen ketiga sering yang terbesar di tahun pertama dan paling sering lupa dimasukkan.

BAB 63 / STUDI KASUS

Anggaran yang meleset tiga kali lipat

Sebuah tim menyusun anggaran dengan memperkirakan token masuk dari panjang pesan pengguna. Angkanya terlihat masuk akal dan disetujui tanpa pertanyaan.

Tagihan bulan pertama tiga kali lipat dari anggaran. Setelah diperiksa, token masuk sebenarnya jauh lebih besar: ia memuat prompt sistem, indeks empat puluh skill, skema seluruh tool yang terlihat, riwayat percakapan, dan hasil pemanggilan tool.



Gambar 63.3 — Susunan token masuk sintetis untuk satu giliran. Pesan pengguna adalah komponen terkecil.

Perbaikannya bukan menaikkan anggaran. Mereka mempersempit indeks skill per agen dan memfilter tool MCP, dan biaya turun hampir separuh tanpa satu pun kemampuan hilang.

> **PERIKSA SUSUNANNYA SEBELUM MENAIKKAN ANGGARAN**
> Ketika biaya melebihi perkiraan, periksa susunan token masuk sebelum menaikkan anggaran. Pada sebagian besar kasus, dua perubahan konfigurasi dari Bab 36 dan 42 menyelesaikan sebagian besarnya.

BAB 63 / LATIHAN

Latihan mandiri

1. Jalankan fase bayangan dan isi kelima parameter dengan angka yang diukur.
2. Hitung tiga skenario volume dan tentukan pada volume berapa rancangan Anda perlu berubah.
3. Petakan susunan token masuk Anda seperti Gambar 63.3 dan cari komponen terbesarnya.
4. Tambahkan komponen pemeliharaan ke anggaran Anda dengan angka jam yang realistis.

BAB 63 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Biaya jauh di atas estimasi	token_masuk_per_giliran ditebak terlalu rendah. Ukur dari pemakaian nyata.
Biaya melonjak setelah menambah skill	Indeks skill masuk konteks setiap giliran. Persempit per agen.
Cache tidak pernah membantu	Ada yang memutus konteks atau menukar toolset di tengah percakapan.
Anggaran ditolak keuangan	Komponen pemeliharaan hilang dari perhitungan.
Tidak tahu biaya per use case	Pisahkan agennya supaya pemakaian dapat diatribusikan.

Tiga pertanyaan review

1. Mengapa token masuk hampir selalu jauh lebih besar daripada token keluar?
2. Sebutkan dua keputusan desain di bab-bab sebelumnya yang berdampak langsung pada biaya.
3. Susun rencana pengukuran dua minggu untuk mengisi seluruh parameter model biaya.

Kunci pembahasan

Token masuk memuat prompt sistem, indeks skill, skema seluruh tool yang terlihat, riwayat percakapan, dan hasil pemanggilan tool, sementara token keluar hanya berisi jawaban — perbandingan sepuluh banding satu adalah hal biasa. Dua keputusan yang berdampak langsung: mempersempit daftar skill per agen dari Bab 36, dan memfilter tool MCP dari Bab 42; keduanya menurunkan token masuk pada setiap giliran. Rencana pengukuran: jalankan agen pada fase bayangan tanpa mengirim balasan ke pelanggan, catat jumlah percakapan dan giliran per hari, baca token masuk dan keluar dari pemakaian Gateway, lalu hitung rasio cache dari selisih antara giliran pertama dan giliran berikutnya dalam satu percakapan.

Model perhitungan adalah susunan penyusun; seluruh angka adalah data sintetis. Perintah pemakaian merujuk dokumentasi OpenClaw, ditinjau 17 September 2026.

BIAYA, SKALA, DAN TATA KELOLA

Bab 64 Kapasitas, batas, dan titik jenuh

Setiap sistem punya titik di mana menambah beban berhenti menambah hasil. Bab ini menyebutkan batas yang akan Anda temui, urutan kemunculannya, dan cara mengenalinya sebelum ia menjadi insiden.

Tujuan belajar

Mengenali batas yang muncul lebih dulu, merancang untuk melewatinya, dan menerima batas yang tidak dapat dilewati. Pada akhir bab, pembaca dapat memperkirakan sampai skala berapa rancangannya bertahan.

Urutan batas muncul

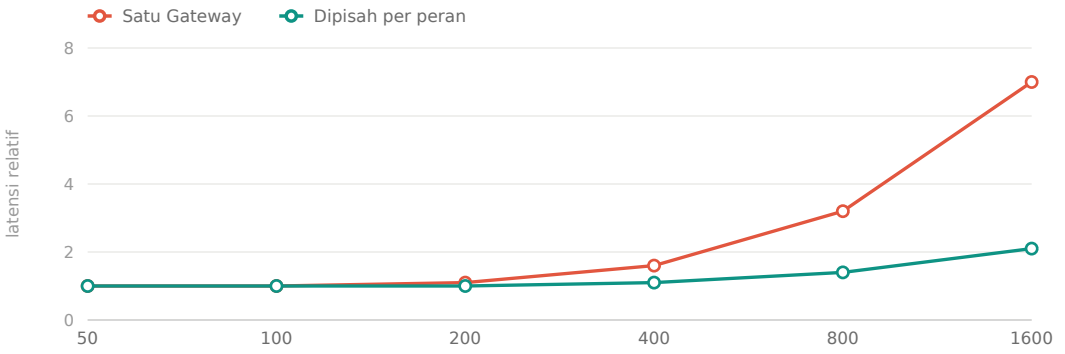
Batas	Gejala saat tercapai	Penanganan
Rate limit penyedia	Giliran gagal atau tertunda pada jam sibuk	Antrean sisi aplikasi, perutean penyedia, atau naik tier
Flood control platform	Kanal menolak kiriman; ledger mencoba ulang	Kurangi pesan progres; gabungkan balasan
Satu proses Gateway	Latensi naik saat banyak percakapan paralel	Pisahkan profil ke host berbeda per agen atau per peran
Jendela konteks	Pemadatan makin sering; mutu jawaban turun	Reset sesi berkala, persempit indeks skill, pangkas riwayat
Kapasitas manusia untuk meninjau	Audit mingguan mulai dilewati	Kurangi cakupan agen sampai rasio audit kembali bermakna

BATAS YANG PALING SERING TERCAPAI LEBIH DULU

Baris terakhir layak dibaca dua kali. Otomasi yang melampaui kemampuan tim mengawasinya bukan penghematan, melainkan pemindahan risiko ke masa depan. Tentukan berapa banyak percakapan yang dapat diaudit per minggu, lalu sesuaikan cakupan agar rasio audit tetap bermakna.

BAB 64 / SKALA

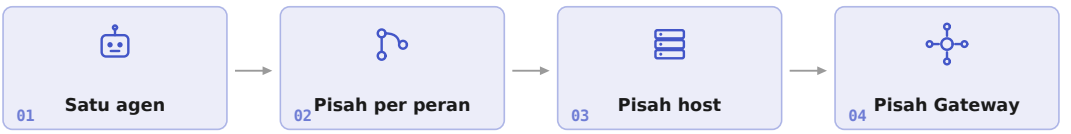
Sampai skala berapa rancangan bertahan



Gambar 64.1 — Kurva sintetis latensi relatif terhadap pesan per hari. Pemisahan per peran menunda titik jenuh, tidak menghapusnya.

Pola yang digambarkan kurva ini berlaku umum: pemisahan menunda titik jenuh dengan biaya kompleksitas operasional. Memisahkan terlalu dini menghasilkan tiga mesin yang harus dirawat untuk beban yang sebenarnya muat di satu; memisahkan terlalu lambat menghasilkan latensi yang dirasakan pengguna.

Urutan pemisahan yang masuk akal



Gambar 64.2 — Naik satu tingkat hanya ketika tingkat sebelumnya benar-benar menjadi kendala yang terukur.

Pemisahan Gateway punya alasan kedua di luar kapasitas, dan sering kali alasan itulah yang menentukan: Bab 49 menyebut bahwa batas kepercayaan yang berbeda menuntut Gateway yang berbeda. Bila Anda sudah harus memisahkan karena kepercayaan, kapasitas ikut terpecahkan.

BAB 64 / PRAKTIKUM

Menemukan batas Anda sebelum ia menemukan Anda

Praktikum ini mencari batas secara sengaja di lingkungan uji, supaya Anda mengenali gejalanya sebelum ia muncul pada jam sibuk.

Langkah 1 — tetapkan batas kapasitas manusia lebih dulu

```
# Pertanyaan yang menentukan cakupan Anda:
Berapa percakapan yang dapat diaudit tim Anda per minggu?
Berapa persen dari volume yang harus diaudit agar rasionya bermakna?
Cakupan maksimum = hasil bagi keduanya.
```

Batas ini biasanya tercapai lebih dulu daripada batas teknis mana pun.

Langkah 2 — kenali gejala tiap batas

Batas	Gejala pertama	Yang sering disalahkan
Rate limit penyedia	Giliran tertunda pada jam tertentu	Jaringan
Flood control platform	Kanal menolak kiriman	Kanal rusak
Satu proses Gateway	Latensi naik bertahap	Model lambat
Jendela konteks	Mutu jawaban menurun pada sesi panjang	Model memburuk
Kapasitas audit	Audit mingguan mulai dilewati	Tim kurang disiplin

Langkah 3 — tentukan ambang pemisahan Anda

Tuliskan dua angka: latensi yang masih dapat Anda janjikan, dan jumlah peran yang kebijakan tool-nya sudah berbeda. Yang mana pun tercapai lebih dulu adalah pemicu pemisahan Anda.

Langkah 4 — uji pemulihan pada beban

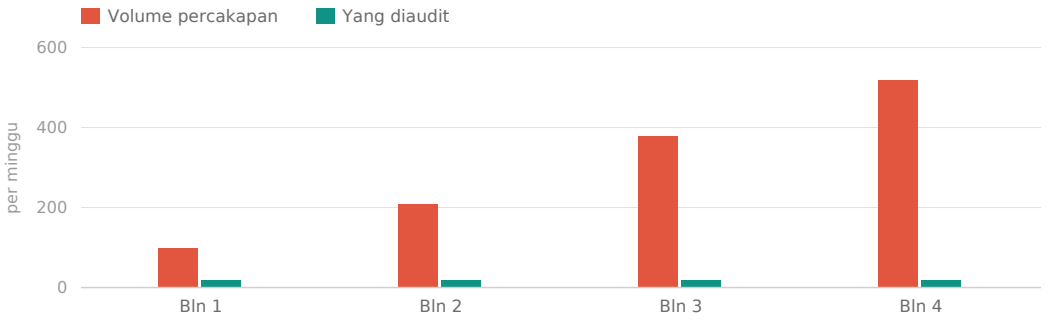
Jalankan lonjakan simultan di lingkungan uji, lalu restart Gateway di tengahnya. Amati apakah giliran dilanjutkan, tugas latar dipulihkan, dan pengiriman dikuras. Mengetahui ini sebelum go-live mengubah sikap Anda pada hari insiden.

BAB 64 / STUDI KASUS

Otomasi yang melampaui kemampuan mengawasinya

Sebuah tim memperluas cakupan agen dari satu kanal menjadi empat dalam tiga bulan. Setiap perluasan berhasil secara teknis: latensi stabil, biaya terkendali, tidak ada kegagalan sistem.

Audit mingguan mereka tetap memeriksa jumlah percakapan yang sama seperti ketika masih satu kanal. Rasio audit terhadap volume turun dari dua puluh persen menjadi kurang dari empat persen, dan tidak ada yang memperhatikan karena angka absolutnya tidak berubah.



Gambar 64.3 — Angka sintetis. Jumlah audit yang tetap terlihat seperti konsistensi, padahal ia adalah penurunan cakupan.

Kesalahan yang lolos ditemukan tiga bulan kemudian lewat keluhan pelanggan, bukan lewat audit. Pada volume itu, empat persen sampel sudah tidak cukup untuk menangkap pola kesalahan yang jarang tetapi berdampak.

> **METRIK YANG MENYEMBUNYIKAN PENURUNAN**
> Ukur audit sebagai rasio terhadap volume, bukan sebagai jumlah absolut. Jumlah yang tetap pada volume yang tumbuh adalah penurunan pengawasan yang tidak terlihat pada laporan mana pun.

BAB 64 / LATIHAN

Latihan mandiri

1. Hitung cakupan maksimum Anda dari kapasitas audit tim, bukan dari kapasitas teknis.
2. Untuk tiap batas pada tabel praktikum langkah dua, tuliskan gejala yang akan Anda kenali.
3. Tetapkan dua angka pemicu pemisahan Anda dan tuliskan mana yang lebih mungkin tercapai lebih dulu.
4. Ubah metrik audit Anda dari jumlah absolut menjadi rasio terhadap volume.

BAB 64 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Giliran gagal pada jam tertentu saja	Rate limit penyedia. Antrekan atau routing ke penyedia lain.
Kanal menolak kiriman	Flood control platform. Kurangi pesan progres dan gabungkan balasan.
Mutu jawaban menurun pada percakapan panjang	Pemadatan makin sering. Reset berkala dan pangkas riwayat.
Latensi naik seiring jumlah pengguna	Satu proses menjadi kendala. Pisahkan per peran lebih dulu.
Audit mingguan mulai dilewati	Cakupan melampaui kapasitas tinjauan. Kurangi cakupan.

Tiga pertanyaan review

1. Mengapa kapasitas manusia untuk meninjau sering menjadi batas pertama yang tercapai?
2. Apa risiko memisahkan Gateway terlalu dini?
3. Tentukan ambang yang akan memicu Anda memisahkan agen per peran.

Kunci pembahasan

Kapasitas teknis tumbuh dengan menambah mesin, sedangkan kapasitas meninjau tumbuh hanya dengan menambah orang — dan menambah orang justru yang ingin dihindari proyek otomasi, sehingga batas itu tercapai lebih dulu. Memisahkan terlalu dini menambah mesin, konfigurasi, dan permukaan keamanan yang harus dirawat tanpa manfaat yang sepadan; setiap Gateway tambahan adalah pemasangan yang harus diperbarui, diaudit, dan dicadangkan. Ambang pemisahan yang masuk akal biasanya adalah ketika dua peran mulai berbeda kebijakan tool-nya, atau ketika latensi yang dirasakan pengguna melewati batas yang Anda janjikan — mana pun yang datang lebih dulu.

Kurva dan angka pada bab ini adalah pemodelan sintetis penyusun untuk menggambarkan bentuk hubungannya, bukan hasil pengukuran.

BIAYA, SKALA, DAN TATA KELOLA

Bab 65 Checklist go-live dan matriks risiko

Bab ini adalah yang akan Anda buka pada minggu terakhir sebelum peluncuran. Isinya sengaja berbentuk daftar, karena pada minggu itu tidak ada yang punya waktu membaca argumen.

Tujuan belajar

Menjalankan dua minggu terakhir sebelum go-live, menyusun matriks risiko dengan pemilik, dan menyiapkan rencana mundur. Pada akhir bab, pembaca memiliki peluncuran yang dapat dibatalkan tanpa drama.

Dua minggu terakhir

Gambar 65.1 — Latihan mundur di H-3 adalah satu-satunya cara mengetahui rencana mundur Anda benar-benar bisa dijalankan.

Latihan mundur layak ditegaskan. Banyak tim menuliskan rencana mundur dan tidak pernah mencobanya, lalu menemukan pada hari insiden bahwa kembali ke proses manual menuntut akses yang sudah dicabut, atau pengetahuan yang sudah dilupakan orang.

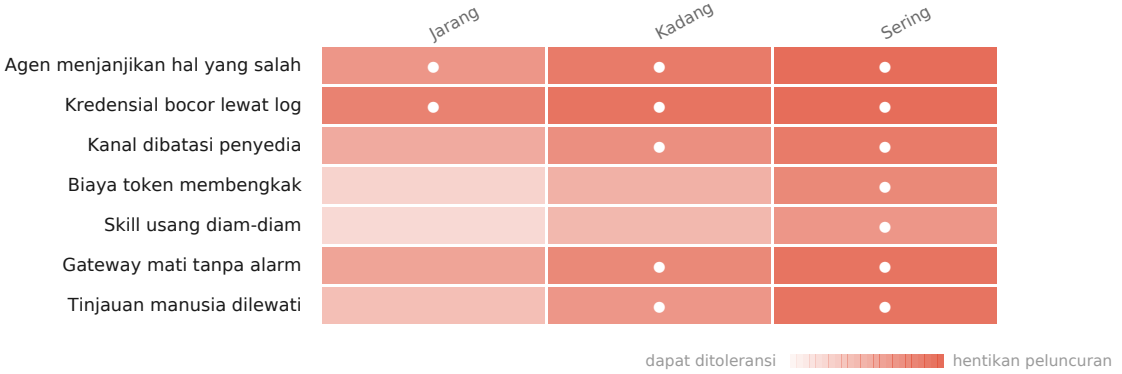
Checklist go-live

- Baseline keamanan Bab 50 lulus seluruhnya
- openclaw security audit bersih dari temuan kritis
- Cadangan sudah diuji pulih ke mesin lain
- Kebijakan tool dan sandbox diverifikasi dengan percobaan nyata
- Jalur eskalasi punya penerima bernama pada jam kerja
- Rencana mundur pernah dilatih, bukan hanya ditulis
- Metrik keberhasilan disepakati sebelum peluncuran
- Ada orang yang berwenang menghentikan tanpa menunggu rapat

Gambar 65.2 — Delapan butir. Butir terakhir sering hilang dan paling menentukan kecepatan respons pada jam pertama insiden.

BAB 65 / RISIKO

Matriks risiko



Gambar 65.3 — Kolom adalah frekuensi, warna adalah keparahan gabungan. Setiap kotak paling gelap harus punya pemilik bernama.

Risiko	Deteksi	Mitigasi
Janji yang salah	Sampel percakapan mingguan	Kategori eskalasi yang tegas; aturan angka-dari-tool
Kredensial bocor	Audit keamanan berkala	Rujukan rahasia; triage untuk berbagi diagnostik
Kanal dibatasi	Ledger pengiriman dan pemutus arus	Kurangi pesan progres; siapkan kanal cadangan
Biaya membengkak	Pemakaian harian dibandingkan anggaran	Persempit indeks skill dan filter tool
Tinjauan dilewati	Rasio audit terhadap volume	Kurangi cakupan sampai rasio kembali bermakna

BAB 65 / PRAKTIKUM

Menjalankan dua minggu terakhir

Praktikum ini adalah jadwal yang Anda jalankan, bukan yang Anda baca. Setiap butir menghasilkan bukti yang dapat ditunjukkan.

Langkah 1 — H-14 sampai H-7

```
# H-14: kunci akses
openclaw pairing list <kanal>     # tutup permintaan tertunda
openclaw security audit --deep    # simpan hasilnya sebagai bukti

# H-7: uji beban
# jalankan lonjakan simultan, lalu restart di tengahnya
```

Simpan keluaran tiap perintah; ini bahan yang akan diminta saat peninjauan.

Langkah 2 — H-3, latihan mundur

Yang dilatih	Cara	Yang dibuktikan
Mematikan agen	Hentikan Gateway	Layanan berhenti bersih
Kembali manual	Jalankan proses lama	Akses lama masih ada
Memberi tahu pengguna	Kirim pesan pemberitahuan	Jalurnya berfungsi
Waktu pemulihan	Catat berapa lama seluruhnya	Ekspektasi realistis

Langkah 3 — H-1, pembekuan

Tidak ada perubahan skill, konfigurasi, atau plugin. Pembekuan memastikan bahwa bila sesuatu berubah perilakunya pada hari peluncuran, penyebabnya adalah beban nyata — bukan perubahan yang baru dimasukkan seseorang.

Langkah 4 — hari H, pantau melekat

```
openclaw logs --follow
openclaw health
# periksa sampel percakapan tiap jam pada hari pertama
```

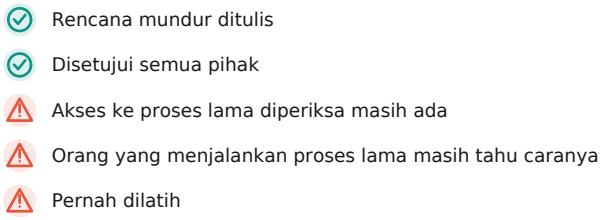
Empat puluh delapan jam pertama adalah tempat sebagian besar masalah muncul.

BAB 65 / STUDI KASUS

Rencana mundur yang tidak dapat dijalankan

Sebuah tim menuliskan rencana mundur yang rinci: matikan agen, kembalikan proses manual, beri tahu pelanggan. Dokumennya rapi dan disetujui semua pihak. Tidak ada yang pernah mencobanya.

Empat bulan setelah go-live, sebuah masalah menuntut mereka kembali ke proses manual selama beberapa hari. Rencana itu gagal pada langkah kedua: akses ke sistem lama sudah dicabut ketika proses lama dianggap tidak dipakai lagi.



Gambar 65.4 — Dua butir terpenuhi dan tiga tidak. Yang gagal semuanya adalah butir yang hanya dapat dibuktikan dengan mencoba.

Memulihkan akses memakan dua hari, dan dua orang yang dulu menjalankan proses manual sudah pindah bagian. Yang seharusnya menjadi kemunduran terkendali menjadi insiden tersendiri.

> **MENGAPA H-3 TIDAK BOLEH DILEWATI**
> Rencana mundur yang belum pernah dilatih adalah dokumen, bukan rencana. Latihan H-3 memakan setengah hari dan menguji hal yang tidak dapat diuji dengan membaca.

BAB 65 / LATIHAN

Latihan mandiri

1. Susun jadwal dua minggu terakhir Anda dengan tanggal konkret dan penanggung jawab tiap butir.
2. Jalankan latihan mundur lengkap dan catat waktu sebenarnya untuk tiap tahap.
3. Periksa apakah akses ke proses lama Anda masih ada dan masih ada yang tahu caranya.
4. Tetapkan siapa yang berwenang membatalkan peluncuran, dan pada kriteria apa.

BAB 65 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Peluncuran mundur berulang kali	Kriteria go-live tidak disepakati di muka. Tetapkan dan tulis.
Rencana mundur gagal saat dibutuhkan	Tidak pernah dilatih. Jadwalkan latihan sebelum go-live berikutnya.
Risiko tidak ada yang menangani	Kotak matriks tanpa pemilik bernama. Tunjuk sekarang.
Insiden lambat ditangani	Tidak ada yang berwenang menghentikan tanpa rapat.
Tidak tahu apakah peluncuran berhasil	Metrik tidak disepakati sebelum peluncuran.

Tiga pertanyaan review

1. Mengapa pembekuan perubahan di H-1 bernilai meski terasa berlebihan?
2. Mengapa setiap kotak risiko paling gelap harus punya pemilik bernama, bukan tim?
3. Susun kriteria go-live Anda sendiri dalam lima butir yang dapat diperiksa.

Kunci pembahasan

Pembekuan memastikan bahwa bila sesuatu berubah perilakunya pada hari peluncuran, penyebabnya adalah beban atau pemakaian nyata — bukan perubahan yang baru dimasukkan seseorang; tanpa itu, diagnosis pada jam pertama menjadi jauh lebih sulit. Pemilik bernama diperlukan karena risiko yang dimiliki tim pada praktiknya tidak dimiliki siapa pun; nama membuat seseorang merasa bertanggung jawab memeriksa. Kriteria go-live bergantung konteks, tetapi lima butir yang hampir selalu pantas: audit keamanan bersih, cadangan teruji pulih, rencana mundur terlatih, penerima eskalasi bernama, dan metrik keberhasilan yang disepakati tertulis.

Checklist dan matriks pada bab ini adalah susunan penyusun, dibangun di atas kemampuan yang didokumentasikan di Bagian VIII.

BIAYA, SKALA, DAN TATA KELOLA

Bab 66 Tata kelola jangka panjang dan serah terima

Bab terakhir membahas hal yang paling sering menentukan apakah sebuah penerapan masih hidup dua tahun lagi: bukan arsitekturnya, melainkan siapa yang merawatnya dan bagaimana pengetahuannya berpindah.

Tujuan belajar

Menetapkan ritme perawatan, menyiapkan serah terima yang tidak bergantung pada satu orang, dan mengetahui kapan sebuah penerapan sebaiknya dihentikan. Pada akhir bab, pembaca memiliki rencana yang bertahan setelah ia sendiri pindah pekerjaan.

Ritme perawatan

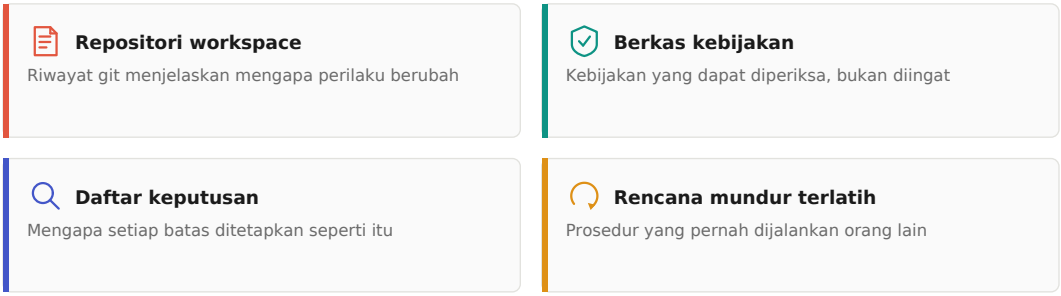
Irama	Pekerjaan
Harian	Periksa kesehatan dan antrean; tindak lanjuti eskalasi yang menggantung
Mingguan	Audit sampel percakapan; bandingkan biaya terhadap anggaran
Bulanan	Jalankan audit keamanan; tinjau temuan dan penekanannya
Kuartalan	Tinjau daftar skill dan plugin; hapus yang tidak dipakai
Tahunan	Uji pemulihan penuh; tinjau ulang kelayakan tiap use case

Irama kuartalan sering dilewati dan paling berdampak. Skill dan plugin menumpuk; sebagian menjadi usang tanpa ada yang menyadarinya, dan setiap entri yang terlihat menambah biaya pada setiap giliran seperti dibahas di Bab 36.

BAB 66 / SERAH TERIMA

Pengetahuan yang tidak menguap

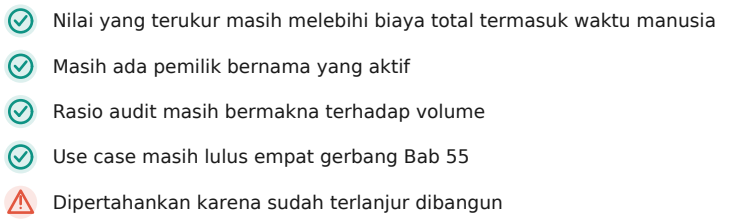
Penerapan yang hanya dipahami satu orang adalah utang yang jatuh tempo pada hari orang itu pergi. Empat artefak berikut membuat pengetahuannya dapat berpindah.



Gambar 66.1 — Empat artefak serah terima. Yang ketiga paling sering hilang dan paling mahal untuk disusun ulang.

Daftar keputusan layak ditegaskan. Konfigurasi menjelaskan apa, tetapi tidak menjelaskan mengapa. Tanpa catatan alasan, penerus Anda akan melonggarkan batas yang Anda tetapkan dengan alasan kuat — bukan karena ceroboh, melainkan karena tidak ada yang memberitahunya bahwa batas itu pernah menyelamatkan sesuatu.

Kapan menghentikan



Gambar 66.2 — Lima pemeriksaan tahunan. Butir terakhir adalah alasan paling umum sebuah sistem bertahan lebih lama daripada kegunaannya.

Menghentikan penerapan yang tidak lagi membayar dirinya sendiri bukan kegagalan; ia keputusan yang sama seriusnya dengan memulainya. Buku ini menutup dengan anjuran itu karena kemampuan menghentikan sesuatu adalah yang membuat Anda bebas mencobanya.

BAB 66 / PRAKTIKUM

Menyiapkan serah terima yang tidak bergantung pada Anda

Praktikum terakhir buku ini menghasilkan empat artefak yang membuat penerapan Anda bertahan setelah Anda pindah pekerjaan. Kerjakan sekarang, bukan pada minggu terakhir.

Langkah 1 — pastikan workspace punya riwayat yang bermakna

```
cd ~/.openclaw/workspace
git log --oneline -20
```

Pesan komit yang menjelaskan perilaku apa yang diperbaiki jauh lebih berguna daripada pesan yang menyebut berkas apa yang disunting.

Langkah 2 — tulis daftar keputusan

```
# Format tiap entri:
Tanggal   :
Keputusan :
Alasan    :
Alternatif yang ditolak dan mengapa :
Yang akan membuat keputusan ini ditinjau ulang :
```

Baris terakhir adalah yang membedakan daftar keputusan dari catatan sejarah.

Langkah 3 — jalankan satu siklus penuh bersama penerus

Irama	Yang dilatih bersama	Siapa yang menjalankan
Harian	Periksa kesehatan dan eskalasi	Penerus
Mingguan	Audit sampel dan bandingkan biaya	Penerus
Bulanan	Audit keamanan dan tindak lanjutnya	Penerus
Latihan mundur	Seluruh prosedur	Penerus, Anda mengamati

Langkah 4 — jalankan lima pemeriksaan tahunan dengan jujur

Nilai yang terukur masih melebihi biaya total termasuk waktu manusia; masih ada pemilik bernama yang aktif; rasio audit masih bermakna; use case masih lulus empat gerbang Bab 55. Bila salah satu gagal, pertimbangkan menghentikannya.

BAB 66 / STUDI KASUS

Sistem yang hidup lebih lama daripada kegunaannya

Sebuah perusahaan menjalankan agen back-office selama dua tahun. Pada tahun pertama, ia menghemat sekitar delapan puluh jam kerja per bulan. Angka itu menjadi dasar persetujuan proyeknya.

Pada tahun kedua, proses bisnisnya berubah. Sebagian besar dokumen kini datang dalam format terstruktur langsung dari sistem pemasok, sehingga ekstraksi tidak lagi diperlukan. Agen tetap berjalan, tetap dirawat, dan tetap dibayar.

TAHUN PERTAMA		TAHUN KEDUA
900 per bulan	**Dokumen yang butuh ekstraksi**	70 per bulan
80 per bulan	**Jam kerja yang dihemat**	6 per bulan
Tidak berubah	**Biaya berjalan**	Tidak berubah
Tidak berubah	**Waktu perawatan**	Tidak berubah
Nilai jelas	**Alasan dipertahankan**	Sudah terlanjur dibangun

Gambar 66.3 — Nilai yang menguap tanpa satu pun kegagalan teknis. Baris terakhir adalah alasan yang paling sering dipakai dan paling lemah.

Pemeriksaan tahunan menemukannya. Agen dihentikan, dan perawatannya dialihkan ke use case lain yang sedang tumbuh. Tidak ada yang gagal; sebuah sistem hanya selesai masa kegunaannya, dan seseorang cukup jujur untuk mengatakannya.

KALIMAT PENUTUP BUKU INI

Menghentikan penerapan yang tidak lagi membayar dirinya bukan kegagalan, melainkan keputusan yang sama seriusnya dengan memulainya. Kemampuan menghentikan adalah yang membuat organisasi berani mencoba.

BAB 66 / LATIHAN

Latihan mandiri

1. Periksa apakah keempat artefak serah terima Anda lengkap, lalu lengkapi yang kurang.
2. Mulai daftar keputusan hari ini dengan tiga keputusan terakhir yang Anda ambil, termasuk alasannya.
3. Jadwalkan satu siklus perawatan penuh bersama orang yang akan menggantikan Anda.
4. Jalankan lima pemeriksaan tahunan pada setiap penerapan Anda, dan bersiaplah menerima jawaban yang tidak Anda harapkan.

BAB 66 / EVALUASI

Diagnosis dan pertanyaan review

Gejala	Pemeriksaan dan koreksi
Tidak ada yang tahu mengapa sebuah batas ada	Daftar keputusan tidak dipelihara. Mulai sekarang, bukan nanti.
Skill dan plugin menumpuk	Tinjauan kuartalan dilewati. Jadwalkan dan tunjuk pelaksananya.
Penerapan bergantung satu orang	Empat artefak serah terima belum lengkap.
Sistem dipertahankan tanpa nilai jelas	Jalankan lima pemeriksaan tahunan dengan jujur.
Pemulihan belum pernah diuji setahun ini	Jadwalkan; cadangan yang tidak diuji bukan cadangan.

Tiga pertanyaan review

1. Mengapa daftar keputusan lebih berharga daripada dokumentasi konfigurasi?
2. Mengapa kemampuan menghentikan penerapan membuat organisasi lebih berani mencoba?
3. Susun rencana serah terima untuk penerapan Anda, dengan asumsi Anda pergi bulan depan.

Kunci pembahasan

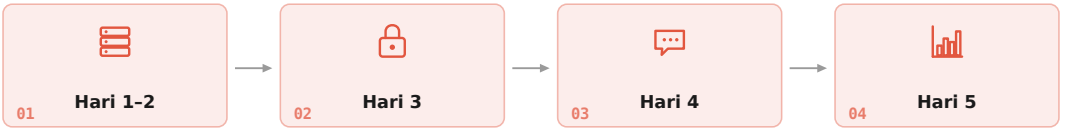
Konfigurasi dapat dibaca langsung dari berkasnya, sedangkan alasan di balik sebuah batas hanya ada di kepala orang yang menetapkannya; tanpa catatan, batas itu akan dilonggarkan oleh orang berikutnya yang menghadapi tekanan untuk menyelesaikan sesuatu dengan cepat. Kemampuan menghentikan menurunkan biaya kesalahan, dan biaya kesalahan yang rendah adalah syarat organisasi berani mencoba hal baru. Rencana serah terima yang layak: pastikan keempat artefak lengkap, jalankan satu siklus perawatan penuh bersama penerus Anda, biarkan ia yang menjalankan latihan mundur berikutnya, dan serahkan daftar keputusan sebagai dokumen terakhir yang Anda tulis.

Kerangka tata kelola pada bab ini adalah sintesis praktik penyusun.

PENUTUP

Apa yang sebaiknya Anda kerjakan besok

Enam puluh enam bab adalah jumlah yang mudah membuat orang menutup buku dan tidak melakukan apa-apa. Halaman ini menyempitkannya menjadi satu minggu kerja.



Empat langkah minggu pertama. Urutannya disengaja: aman dulu, berguna kemudian.

1. Hari satu dan dua: pasang aplikasi macOS, jalankan first run, dan biasakan diri dengan dasbor. Bab 3 sampai 5.
2. Hari tiga: tegakkan lantai keamanan sebelum kanal apa pun dibuka. Bab 13, lalu jalankan openclaw security audit.
3. Hari empat: sambungkan satu kanal saja — yang paling Anda butuhkan — dengan daftar izin sempit. Bab 26 atau 28.
4. Hari lima: ukur. Catat berapa banyak permintaan yang datang, jenisnya apa, dan berapa lama ditangani sekarang. Bab 55.

SATU SARAN TERAKHIR

Yang paling sering menghentikan orang bukan kesulitan teknis, melainkan mencoba membangun terlalu banyak sekaligus. Satu kanal, satu agen, satu use case yang kecil dan dapat diukur — itu lebih bernilai daripada rancangan besar yang tidak pernah berjalan.

Tentang buku ini

Seluruh perintah, nama berkas, dan kunci konfigurasi berasal dari dokumentasi resmi OpenClaw yang ditinjau pada 17 September 2026 terhadap v2026.9.4. Angka bisnis di Bagian IX dan X adalah data sintetis, ditandai di tempatnya, dan disusun agar bentuk perhitungannya dapat Anda tiru — bukan agar angkanya dipakai langsung.

OpenClaw bergerak cepat. Bab 14 mengajarkan cara memverifikasi sendiri, dan itu adalah bab yang paling tidak akan menua. Ketika sebuah perintah di buku ini tidak lagi cocok dengan mesin Anda, percayai mesin Anda.

Disusun oleh Romi Nur Ismanto — hello@rominur.com — www.rominur.com

Edisi Bahasa Indonesia, 17 September 2026.