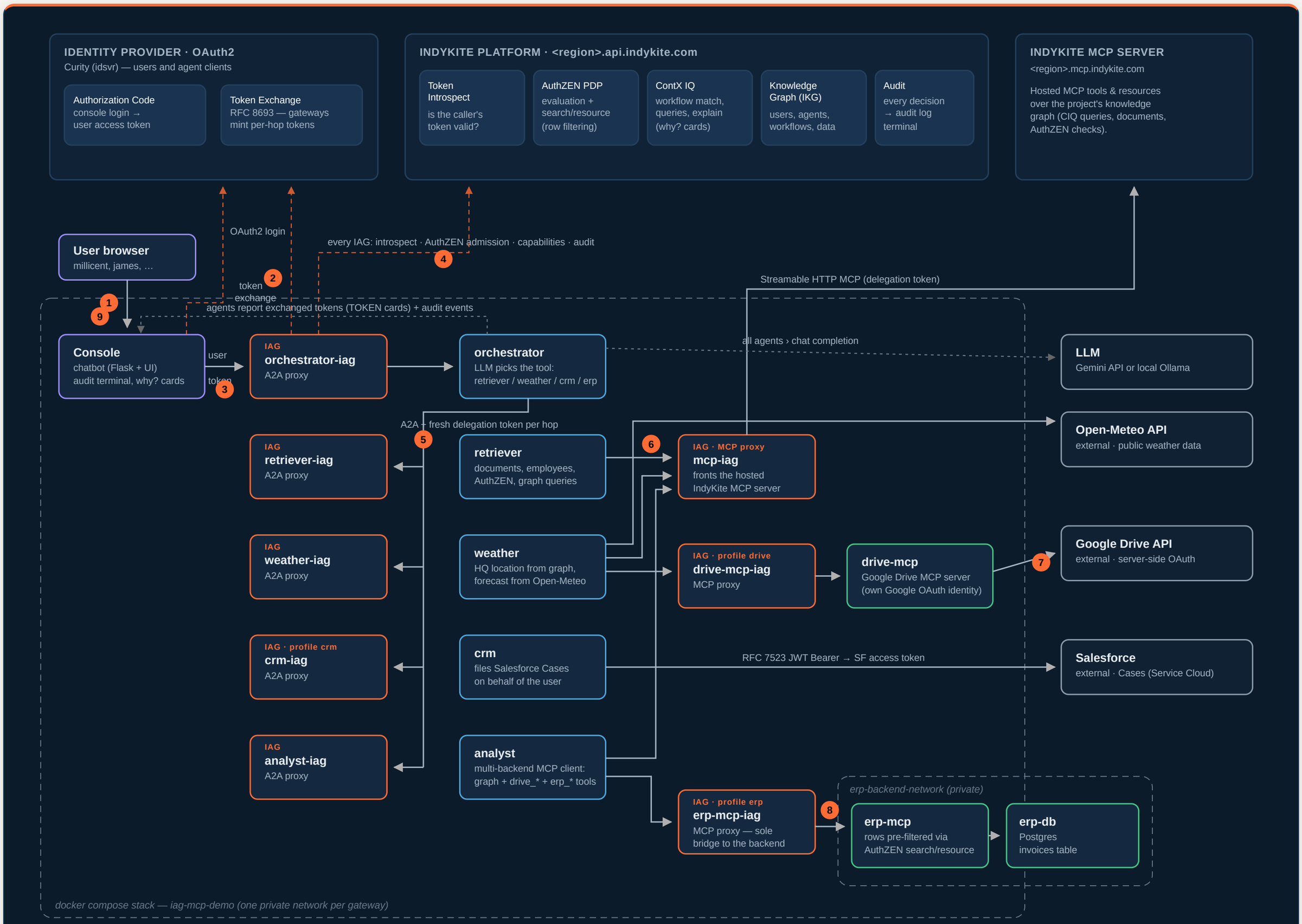
Demo architecture

Every hop between containers goes through an IndyKite Agent Gateway (orange): the caller's token is introspected, the call is admitted against the knowledge graph and limited to the tools the caller may use, a fresh delegation token is minted for the next hop, and an audit event is emitted. Greyed-out groups are optional compose profiles that are currently off.



The flow

1. **Login & prompt**: the user logs in at the console through the IdP's OAuth2 Authorization Code flow, comes back with a user access token, and sends a prompt.
2. **IdP token services**: the same IdP that authenticated the user also runs the **token-exchange service** (RFC 8693) that every gateway calls to mint a fresh delegation token — user as subject, agent chain in `act` - for the next hop.
3. **Console → orchestrator**: each prompt is forwarded with the *user's* token to **orchestrator-iag**.
4. **Gateway checks (every hop)**: the gateway introspects the token, exchanges it (step 2), asks the platform who the caller is (`whoami`) and sends one AuthZEN `CAN_TRIGGER` evaluation: a policy admits the call when a Workflow the caller can trigger has exactly the token's delegation chain as its path of agents. In front of an MCP server it also limits the tools, resources and prompts to those the caller may use (`CAN_USE`). Every decision emits an audit event.
5. **Orchestrator routes**: the LLM picks a tool; the A2A call goes to the chosen agent's gateway, where the same checks run again on the new chain.
6. **Agents fetch data**: retriever/analyst/weather use MCP through **mcp-iag** to the hosted IndyKite MCP server; the analyst additionally reaches Google Drive via **drive-mcp-iag** and the ERP via **erp-mcp-iag**.
7. **External parties**: Google Drive (server-side OAuth identity), Salesforce (the crm agent swaps its IndyKite delegation token for a Salesforce access token via RFC 7523 JWT Bearer — a second, independent trust domain), Open-Meteo, and the configured LLM.
8. **Data-plane authorization**: downstream services can ask the platform what the calling user may see before answering - e.g. **erp-mcp** asks AuthZEN `search/resource` which resources the user may `CAN_VIEW` and returns only those rows. The backend lives on a private network whose sole bridge is its gateway.
9. **Back to the console**: agents report every minted token as a TOKEN card and gateways' decisions appear in the audit-log terminal; deny cards offer a *why?* graph explanation (ContX IQ).

Legend

- IndyKite Agent Gateway (IAG) container
- Agent container (A2A / LangChain)
- Console / user
- Data-plane service (MCP server, database)
- External party (SaaS / public API)
- Control plane: OAuth2, introspection, AuthZEN, audit
- Network boundary (compose stack / private backend)

Optional profiles

- **drive**: Google Drive MCP behind a gateway
- **crm**: Salesforce Cases on behalf of the user
- **erp**: Postgres invoices, graph-filtered rows