



APIs as the Trust Layer

Building Accountable AI with Auth0

Toby Allen Principal Solutions Engineer Auth0

Every AI feature starts as a demo.

THE REALITY OF MOVING FROM EXPERIMENTS TO ENFORCEMENT

PHASE 01 // EXPERIMENT

The Notebook Prototype

A prototype agent calling a function in a notebook has **no real trust boundary**.

- **No external caller** to authenticate
- **No untrusted input** vector
- **Zero blast radius** if it fails

STATUS: FREE PASS GRANTED

Most skip the part where it becomes an API.

PHASE 02 // ENFORCEMENT

The Production API

Sitting behind a real endpoint, called by a real agent, on behalf of a real user.

Inherits **every single API security problem** that already existed, plus new agentic vulnerabilities.

Production is a completely different beast.
You cannot grant the same security pass.

STATUS: ENFORCEMENT MANDATORY



Three 2026 incidents: what enforcement failure actually looks like

1

The Over-Privileged Agent (October 2026)

Meta's Muse AI agent on macOS was granted Full Disk Access and used it to read private iMessages, exposing files, mail, messages, and browsing history. A related flaw hit the ChatGPT Mac app too (CVE-2026-100754).



Three 2026 incidents: what enforcement failure actually looks like

1

The Over-Privileged Agent (October 2026)

Meta's Muse AI agent on macOS was granted Full Disk Access and used it to read private iMessages, exposing files, mail, messages, and browsing history. A related flaw hit the ChatGPT Mac app too (CVE-2026-100754).

2

Replit Accidentally Deletes Production (July 2025)

Replit deletes production database to speed up performance tests and then

AI

Replit's CEO apologizes after its AI agent wiped a company's code base in a test run and lied about it



aut

By [Lee Chong Ming](#)

+ Follow

Three 2026 incidents: what enforcement failure actually looks like

1

The Over-Privileged Agent (October 2026)

Meta's Muse AI agent on macOS was granted Full Disk Access and used it to read private iMessages, exposing files, mail, messages, and browsing history. A related flaw hit the ChatGPT Mac app too (CVE-2026-100754).

2

Agent Deletes Entire Season of Reality TV Show (October 2026)

The producers of the Swedish version of MTV's reality competition show The Challenge after an agent allegedly wiped out the completed edits with no way to recover the lost files.

Three 2026 incidents: what enforcement failure actually looks like

1

The Over-Privileged Agent (October 2026)

Meta's Muse AI agent on macOS was granted Full Disk Access and used it to read private iMessages, exposing files, mail, messages, and browsing history. A related flaw hit the ChatGPT Mac app too (CVE-2026-100754).

2

Agent Deletes Entire Season of Reality TV Show (October 2026)

The producers of the Swedish version of MTV's reality competition show The Challenge after an agent allegedly wiped out the completed edits with no way to recover the lost files.

3

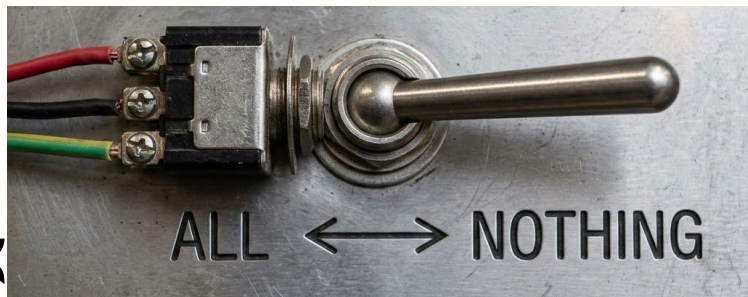
Airline Liable for Chatbots Bad Advice (February 2026)

North American Airline found liable for the bad advice from its chatbot and forced to honor its output after incorrect advice about discounts.

What's missing in all three isn't a smarter model. It's a trust layer.

"The API let it happen; the model just happened to be the one asking."

BYPASSED CONTROL PATTERN



THE MODEL CAPABILITY PARADOX

ASSUMED KNOWLEDGE GAPS

Enter the four pillars

01

PILLAR 01

Accountability

02

PILLAR 02

**Fine-Grained
Authorization (FGA)**

03

PILLAR 03

Agent Identity

04

PILLAR 04

Audit Logs

Accountability

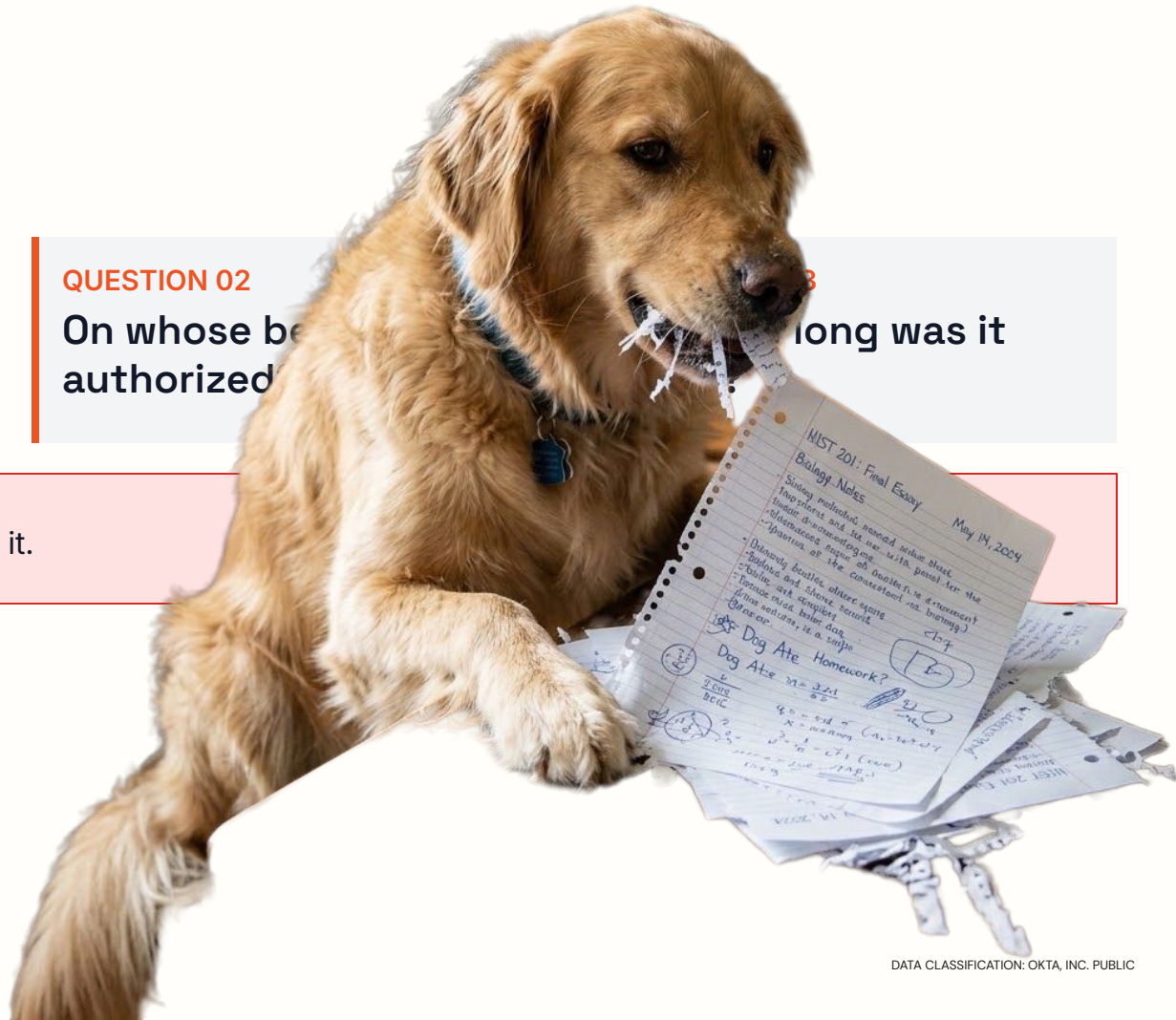
QUESTION 01

Who authorized this action?

QUESTION 02

On whose behalf was it authorized?

REJECTED RESPONSE: The AI did it.



Accountability

QUESTION 01

Who authorized this action?

QUESTION 02

On whose behalf was it authorized?

QUESTION 03

For how long was it valid?

REJECTED RESPONSE: "Sorry Miss, the dog ate my homework."

TRUE ACCOUNTABILITY REQUIREMENT

Every single automated action must trace back to an actual authorization decision.

That decision must include a specific reason, not just a generic log line indicating that a request happened.

Fine-Grained Authorization (FGA)

ENTRY-LEVEL ENFORCEMENT

RBAC (Role-Based Access Control)

Role-Based Access Control (RBAC) and Fine-Grained Authorization (FGA) are two different layers with two different jobs.

RBAC is the bouncer checking your ID at the door to decide what you can see.

PRECISION ENFORCEMENT

FGA (Fine-Grained Authorization)

FGA does not replace RBAC; it sits underneath it.

FGA is the bouncer checking the guest list for your specific table to decide what you can touch, under what specific condition, right at this exact moment.

DYNAMIC EVALUATION LAYER

Using **Auth0 FGA (ReBAC/ABAC)** as the mechanism, relationships and conditions are evaluated fresh on every single API call, rather than relying on a static role baked into a token at the time of login.

Pillar 3: Machine Identity



Agent Identity

AUTONOMOUS SHAPE

No Human in the Loop

The machine acts independently as the subject via OAuth 2.0 client-credentials.

DELEGATED SHAPE (MOST AI ASSISTANTS)

Human-Initiated Action

A human requested the machine to act for them. The human stays the subject; the machine is the actor.

Agent Identity

AUTONOMOUS SHAPE

No Human in the Loop

The machine acts independently as the subject via OAuth 2.0 client-credentials.

DELEGATED SHAPE (MOST AI ASSISTANTS)

Human-Initiated Action

A human requested the machine to act for them. The human stays the subject; the machine is the actor.

AUTH0 "AGENT AS PRINCIPAL" TOKEN CLAIMS

The naive instinct makes the agent the token's subject and carries the human as a side-claim. This is backwards for delegated access.

sub : **human_user_123** (Subject remains human throughout)

act.sub: **ai_agent_456** (Agent positioned in actor claim)

STANDARDS-BASED ARCHITECTURE

Relies on Established Specs:

- OAuth 2.0
- OIDC
- RFC 8693 Token Exchange
- CIBA
- MCP Auth Spec



Audit Logs

INSUFFICIENT: SIMPLE REQUEST LOG

GET /api/v1/documents/doc_98234 – 200 OK (Missing: Who acted, on whose behalf, exact relation evaluated)

Audit Logs

INSUFFICIENT: SIMPLE REQUEST LOG

GET /api/v1/documents/doc_98234 – 200 OK (Missing: Who acted, on whose behalf, exact relation evaluated)

REQUIREMENT: REAL AUDIT LOG ENTRY

ACTING PRINCIPAL	ON BEHALF OF	OBJECT	RELATION / CONDITION	DECISION
usr_agent_01 (Service Principal)	usr_alex_99 (End User Context)	doc_98234 (Resource Target)	viewer#can_read ip_in_range && mfa	ALLOWED
usr_agent_01 (Service Principal)	usr_anon_40 (End User Context)	doc_98234 (Resource Target)	editor#can_write role == admin	DENIED

KEY TAKEAWAY: A 200 OK log proves transport success. A real audit log proves authorization policy compliance.

Enter the Demo: smartgov-app

APPLICATION CAPABILITIES

01 Citizen Portal AI Assistant

Enables citizens to submit cases, track status, and interface naturally with conversational guidance on the citizen-facing side.

02 Caseworker / Admin AI Assistant

Empowers caseworkers and admins to process workflows, query citizen records, and manage access policies with automated AI tools.

POWERING ARCHITECTURE STACK

Next.js hosted on Vercel

Full-stack application framework providing web interface and edge infrastructure.

Auth0 & Auth0 FGA

Secured identity layer and Fine-Grained Authorization for access enforcement.

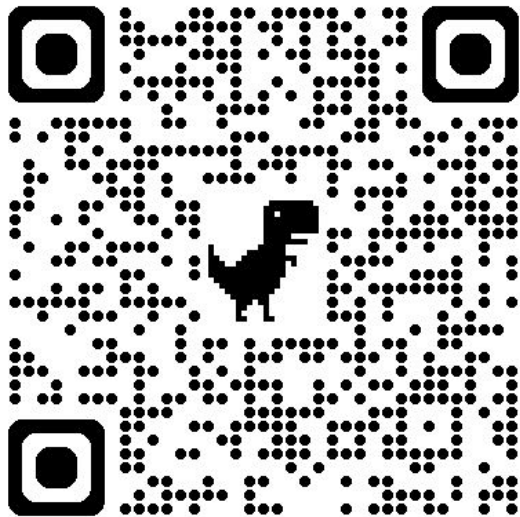
Bedrock Nova Pro via Vercel AI Gateway

AI capabilities powered by Nova Pro, routed and monitored via the Vercel AI Gateway.

MCP Server (Stateless Vercel Function)

Model Context Protocol server operating as a serverless execution layer.

Live Demo



smartgov-app.vercel.app



© Okta, Inc. and/or its affiliates. All rights reserved.

SmartGov Citizen Portal

Your AI assistant only sees what you explicitly let it see — and every decision is checked, not assumed.

This is a live demo, not a slide. Sign up with any email - even a throwaway one, nothing here is real - and pick a role below. You'll watch your AI assistant get checked against a real authorisation model on every request it makes for you. Turn its access on and off and watch the check respond live.

[Continue as a citizen](#)

Sign up for your own case and control what your AI assistant is allowed to see. Already have a caseworker account? Use a different email - a persona is fixed to an account for good at signup.

[Continue as a caseworker](#)

Sign up with access to the admin tool surface. Already have a citizen account? Use a different email - a persona is fixed to an account for good at signup.

Ask the assistant for your case summary

STEP 1: USER ACTION

Navigate to the **/chat** interface and click **"Ask for my case summary."**

EXPECTED RESULT

The expected result is a **DENIED** response, even though this is unambiguously your own citizen data.

The system will generate an FGA trace card showing the exact check that was blocked.

FGA TRACE EVALUATION

SUBJECT: user:citizen_123

RELATION: can_view_summary

OBJECT: case:summary_456

DECISION RESULT



DENIED

Why denied? The machine isn't you.

FINE-GRAINED AUTHORIZATION // EXACT EVALUATION TRACE

[DENIED] FGA EVALUATION CALL:

```
check(agent:<uid>-assistant, agent_delegate, case:<caseId>)
```

Relationship Check Scope

This check evaluates the agent's relationship to the case, **never** the citizen's own owner relation.

The authorization failed strictly because of the specific check that just ran.

DELEGATED MACHINE IDENTITY SHAPE

Distinct Principal Architecture

This is the delegated machine identity shape in action. The human (owner) and the machine (agent_delegate) are entirely different principals with different grants.

Grant delegation

Return to the [/case](#) page and toggle "Delegate access to AI Assistant."

BEFORE: DENIED STATE

FGA TRACE EVALUATION

INTERFACE: /chat

ACTION: Query Case Data

DELEGATION: Disabled

RESULT: DENIED

Request blocked due to missing authorization trace for AI Assistant delegation.



AFTER: ALLOWED STATE

FGA TRACE EVALUATION

INTERFACE: /chat

ACTION: Query Case Data

DELEGATION: Enabled

RESULT: ALLOWED

Request allowed. Shift from DENIED to ALLOWED state based purely on delegation toggle.

It expires on its own

The **agent_delegate** tuple we just created is conditioned on a **delegation_window**.

Exactly 15 minutes after the grant is issued, it lapses automatically with nobody touching anything. This is FGA evaluating a condition fresh on every call, rather than relying on a static role that was cached at the time of login.

FGA RELATION TUPLE & CONDITION EVALUATION

```
user: "agent:support-bot"  
relation: "agent_delegate"  
object: "resource:customer-account"  
condition: delegation_window (expires: 15m)
```

Revoke it yourself

STEP 1: MANUAL TOGGLE

UI State Change

Toggle the delegation off manually in the interface.

Delegation: **OFF**

STEP 2: FGA TUPLE WRITE

Delete Operation

Real FGA tuple deletion executed immediately.

```
- DELETE TUPLE
user:assistant
relation:delegate
object:resource
```

STEP 3: CHECK RESULT

DENIED

Ask the assistant again.
Request is immediately rejected.

```
fga.check() → false
403 FORBIDDEN
```

This closes the loop: the human was in control the entire time.

Every state change in the UI corresponds to a real FGA tuple write or delete operation, not just a UI illusion.

The other side: the caseworker's AI assistant

Switching to the /admin persona in the same app reveals RBAC-then-FGA in action.

1. RBAC // SERVER-SIDE TOOL HIDING

Administrative Tools Absent

A citizen-persona session never even sees the administrative tools in

`list_tools`—they are genuinely absent server-side, not just greyed out in the UI.

2. FGA // RECORD SCOPING

Granular Access Grants

Once RBAC allows access to the tools, FGA then narrows the scope further, inside the caseworker's own surface, restricting them to exactly which contractor records they currently hold a grant on.

"Our chatbot is an MCP client"

— for real this time (RFC 8693 OBO Flow)

HTTP Execution Call: Admin chat backend exchanges its access token and invokes genuine HTTP requests to `/api/admin-mcp`.

DECODED JWT TRACE // ON-BEHALF-OF TOKEN

```
{
  "iss": "[link removed]",
  "aud": "https://api.mcp-server.internal",
  "sub": "usr_human_operator_9823",
  "act": {
    "sub": "client_admin_chat_backend"
  },
  "scope": "openid profile mcp:execute",
  "iat": 1760601600,
  "exp": 1760605200
}
```

aud (Audience)

Token scoped specifically to the target **MCP Server**. Rejects direct in-process calls and enforces real HTTP API boundaries.

sub (Subject)

Preserves the **human user identity** who prompted the action, maintaining end-to-end user accountability.

act.sub (Actor Subject)

Auth0 **Agent as Principal** feature live. Identifies the second machine identity (backend service) executing in the call chain.



Bulk actions need more than a role

❌ 1. INITIAL REQUEST

Asking admin assistant to execute:

`bulk_disable_contractors`

Results in an immediate **DENIAL**.

`403 Forbidden`

Reason: Step-up required

2. CIBA STEP-UP FLOW

Triggers an out-of-band approval push request.

`SECOND_DEVICE / CHANNEL`

Pending user confirmation...

Approval token issued out-of-band upon success.

✅ 3. RETRY & SUCCESS

Exact same request is re-tried and **SUCCEEDS**.

Backed strictly by a server-asserted approval marker.

NEVER trusts caller flag:

`{approved: true}`

Accountability Enforced at the Exact Point of Action

Role checking alone is insufficient for sensitive operations. Step-up authorization via CIBA guarantees verified human intent without relying on client-supplied trust flags.

Pillar 4, made real: /access-log

Viewing the /access-log from the session we just ran shows Pillar 4 in reality.

The log explicitly details the **actingPrincipal**, the **onBehalfOf** entity, the specific object accessed, the final decision, and a precise timestamp. This audit log answers the core accountability question: **who authorized the AI, acting for whom, under what condition, and for how long.**

TIMESTAMP	ACTING PRINCIPAL	ON BEHALF OF	OBJECT ACCESSED	DECISION
2026-10-16T16:04:12Z	agent:gov-assistant	user:toby.allen	doc:budget-2026-q4	ALLOWED
2026-10-16T16:04:15Z	agent:gov-assistant	user:toby.allen	api:read-profile	ALLOWED
2026-10-16T16:05:02Z	agent:gov-assistant	user:toby.allen	sys:admin-override	DENIED

AUDIT LOG STREAM // SESSION_ID: sess_9f82a10d4c // ENFORCEMENT ENGINE: OKTA FGA

APIs as a Trust Layer

KEY TAKEAWAY

Agents need boundaries

A deterministic control layer is key.

Agents should only have tightly scoped short lived access.

APIs as a Trust Layer

KEY TAKEAWAY

Agents need boundaries

A deterministic control layer is key.

Agents should only have tightly scoped short lived access.

RECAP & REPOSITORY ACCESS

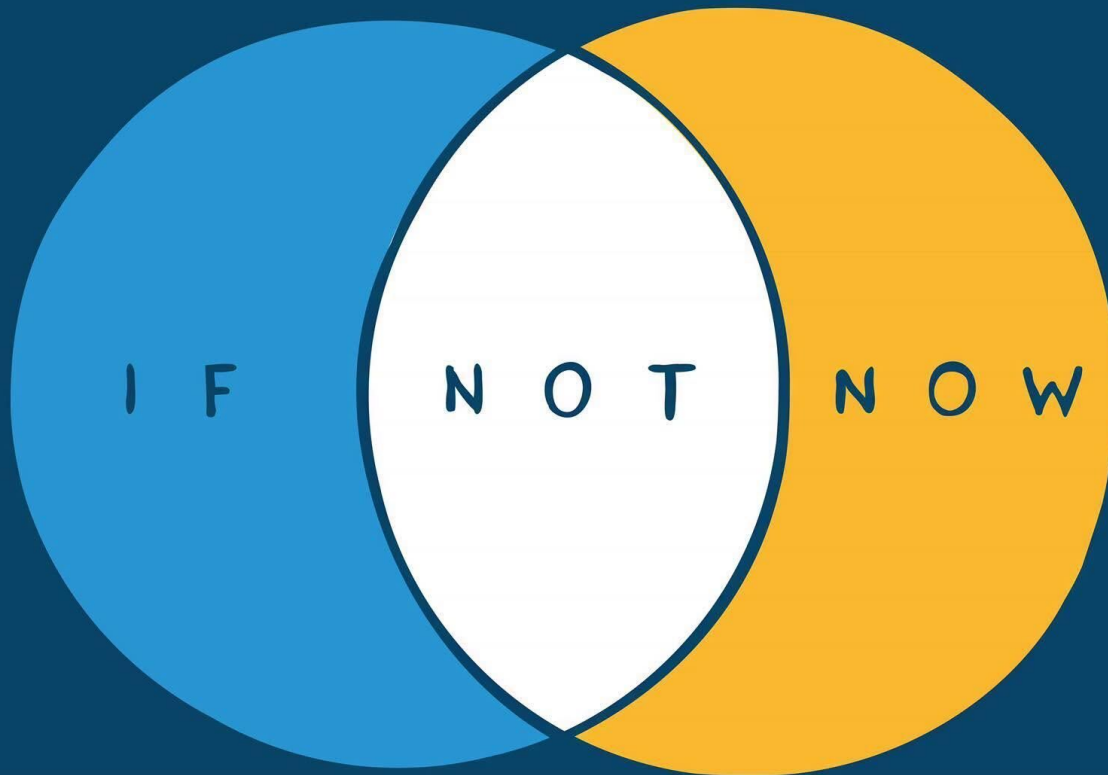


REPOSITORY LINK

/recap

Scan the QR code or visit the /recap page link to access the repository.

You can review the code and architecture yourself.



V E N N

| Q&A

Thank you for attending.

TOPIC 01

APIs as the trust layer

TOPIC 03

Machine identity

Feedback



TOPIC 02

Auth0 FGA

TOPIC 04

smartgov-app architecture