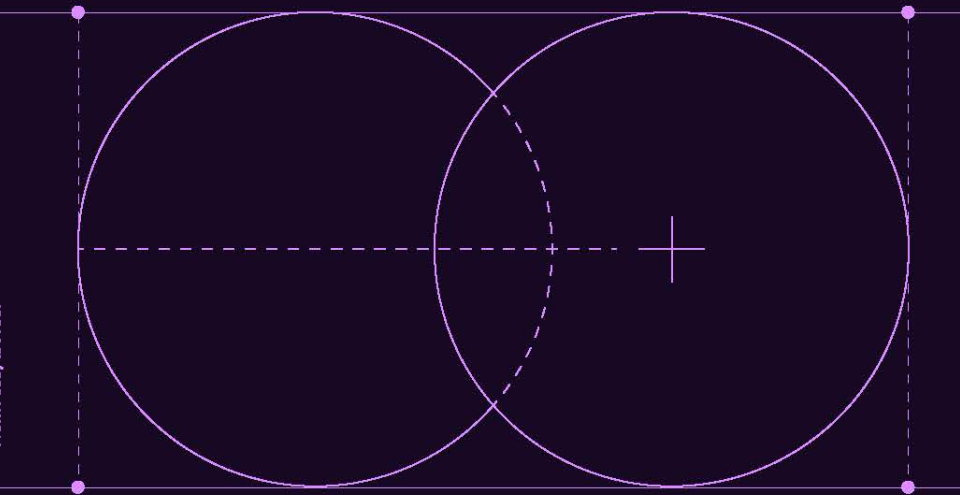
cyera

From Policy to Proof

Nine Disciplines for Governing Enterprise AI

By Eddie Borrero



From Policy to Proof

Nine Disciplines for Governing Enterprise AI

Eddie Borrero

Copyright © 2026 Eddie Borrero
All rights reserved.

No part of this book may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the copyright holder, except for brief quotations used in reviews, articles, and other uses permitted by copyright law.

eBook ISBN: 978-1-972014-30-1
Paperback ISBN: 978-1-972014-29-5

Published by: Atlas Elite Publishing

This book is intended for informational and educational purposes only. The views and information expressed herein are those of the author. While every effort has been made to ensure the accuracy and reliability of the information presented, the author makes no representations or warranties regarding its completeness or applicability to any specific organization, technology, or regulatory environment.

All trademarks, product names, and company names mentioned in this book are the property of their respective owners and are used for identification purposes only. Their inclusion does not imply endorsement.

Dedication

To God, Michelle (my wife), and for Julian, Sophia, and Alexa, who have stood beside me through every late call, every long night, and every year I spent as a chief information security officer, bringing the stress of the job home with me. You made the work possible, and you made coming home from it worth it.

And for everyone still fighting the good fight in information security. It is a thankless career most days. Nobody notices the breach that didn't happen, the rule that held, the quiet Tuesday you earned by working the weekend. I notice. The world needs what you do, and in the age of AI it needs it more than ever.

Thank you for all of it.

Acknowledgments

A book about AI governance is really a book about the people who do it, and I owe this one to many of them.

To the executives, security leaders, privacy officers, and data teams at the companies whose experiences appear in these pages, some named and most not: thank you for letting me learn from your hard days as well as your good ones. Where a story here is told without a name, it is because the lesson belongs to everyone and the difficulty belonged to you.

To my colleagues at Cyera: the product teams building what this book keeps insisting on, the sales force and the executives who sit across the table from clients and prospects every day, and everyone in between. You are not just selling into a market. You are creating one and helping shape what AI governance and security will look like for the foreseeable future. The ideas in Chapter 2 are sharper because of the conversations we've had along the way.

To the mentors, sponsors, and colleagues who have taught me so much over the years, at every company and in every role. Some of you know exactly which lesson you gave me. Some of you have no idea. I remember all of it, and I have tried to pass it on here.

To the researchers, regulators, and journalists whose work is cited in the Sources section: this book stands on what you found and reported. I have tried to represent it accurately, and any errors in how I've used it are mine.

And to Michelle, Julian, Sophia, and Alexa, who made room for the writing and put up with the talking about it. Thank you.

Table of Contents

Introduction 1
Quick Reference: 3
The Nine Playbooks at a Glance
Chapter 1: 16
Governance on Paper
Chapter 2: 19
One Framework Instead of Dozens: The Common Control Framework
Chapter 3: 32
Discovery & Accountability
Chapter 4: 37
Risk Management
Chapter 5 43
Protection and Resilience
Chapter 6 51
Privacy and Compliance
Chapter 7 58
Transparency and Auditability
Chapter 8 63
Data Visibility and Quality
Chapter 9 68
Human Oversight
Chapter 10: 74
Fairness & Non-Discrimination
Chapter 11: 80
Incident Response
Glossary 86
Sources 92
A Note from the Author 98

Introduction

I've watched the governance program this book warns you about from the inside. It had a review board, real policies, and a defined path every AI initiative had to walk through before it reached production. Security had a seat. Compliance had a seat. Data had a seat. On the org chart, it looked like exactly what a responsible company should have. And it quietly became a very thorough process for approving things and in some cases after they had already happened. You'll get the whole story in Chapter 3. I tell it early because I want you to know this book was written from inside the problem, not above it.

Since then, I've watched the same pattern play out at company after company. The right executives sit down, write real policy, and build a real sign-off process. But almost none of that work includes a technical way to check, day to day, whether the policy or approved intent is being followed Especially as employees are now at the helm of creating their own agents. The space between a policy that exists and a control that works is where this entire book lives, and closing it is what this book is for.

What This Book Gives You

This is a working playbook for governing AI end to end. It's built around a Common Control Framework that I like to think of as a “Policy to Proof Backbone”: nine playbooks that map outward to the major standards companies answer to, including the NIST AI Risk Management Framework, the EU AI Act, ISO/IEC 42001, MITRE ATLAS, OWASP's AI security work, AIUC-1, and that extend to whatever lands after this book is printed. The point is that you stop rebuilding your program from scratch every time a new one arrives. Chapter 2 lays out the framework, names the executive who should own each domain, defines the maturity scale the rest of the book uses, and introduces the one piece of infrastructure every domain depends on.

Then, domain by domain, you learn what each one means in practice, what it looks like when it's working, how to build it,

where most companies stand today, and the one question the executive who owns it needs to be able to answer. Each chapter draws on real companies, some named and some not, that learned the lesson the expensive way so you don't have to.

Who This Book Is For

This book is for anyone sitting on, reporting to, or advising an AI governance effort: chief executives, technology and information leaders, security chiefs, privacy and compliance officers, finance leaders, general counsel, and board members at a company where AI is already running whether or not the policy has caught up to it yet. It assumes you're comfortable with words like materiality, GDPR, and least privilege. It doesn't assume you're a data scientist, and it won't teach you to build or tune a model.

How to Read It

The Quick Reference that follows this introduction is a compact guide to the book: the nine domains, who owns each, the question each one asks, and where to start. Read it first if you want the map before the territory, and come back to it whenever you need to find your way to a chapter. Then read the chapters in order, and each domain will make sense in light of the one before it. That's the dependency order, and Chapter 2 explains why it differs from the order you should build in. Read a single chapter on its own and you'll still walk away with something you can act on this quarter. Either way, keep one question in mind the whole way through, because every chapter comes back to it. Not "Do *we have a policy for this*?" but "Could *we prove it's being followed*?"

Quick Reference:
The Nine Playbooks at a Glance

This section is the book at a glance. Use it to find your way back to a chapter, to brief someone who hasn't read the book yet, or to check where your own program stands. Everything here is explained in full in the chapters; nothing here is new.

The Ideas Everything Else Rests On

Three doors. Every company is investing in generative tools and capabilities and it enters a company through three fronts, and each has to be governed with slight differences. The **Enterprise front** is the AI you build on purpose, with a sponsor and a budget; deliberate, human-led governance works here. The **workforce front** is the copilots, chatbots, and embedded features that show up on employees' desktops by the hundreds; nobody files an intake request for these. The **developer front** is the coding assistants and agents wired into how software / product gets built and deployed; it moves fastest and is noticed least. Committees and intake forms govern the first front and cannot keep pace with the other two. (Chapter 2)

The Proof Plane. A living map of every AI system and agent running in the company, across all three fronts, kept current by itself. For each one it records the owner, the model, what's in its harness, what data it touches, who can reach it and what it can reach, and its approved scope, along with a tamper-evident record of everything material it has done, and it holds each agent to that scope wherever its activity passes through a control point. It is a reference architecture, not a single product: best anchored in a platform built around the data itself, and only as complete as the telemetry behind it. Every domain asks it a different question. (Chapter 2)

The maturity scale. Level 1, Initial: nothing formal. Level 2, Repeatable: a process exists and is followed some of the time, for some of the AI, usually the Enterprise front. Level 3, Defined: standardized and applied consistently across the AI the company knows about. Level 4, Managed: controls enforced

technically across all three fronts, with the Proof Plane in place. Level 5, Optimizing: the program improves itself. In my experience, most large companies sit at Level 2 on most domains; this and every "Where most companies stand" below is a practitioner assessment, not a survey finding. (Chapter 2)

Read in dependency order, build in risk order. The chapters follow the order in which the ideas depend on each other. To build, start with two things at once: name the owners and stand up the council (Chapter 3), and start continuous discovery across all three fronts (the Proof Plane). Then go after the two weakest domains everything else stands on: Data Visibility and Quality (Chapter 8) and Transparency and Auditability (Chapter 7). Intake and tiering (Chapter 4) come next. Fairness (Chapter 10) follows close behind if your AI touches hiring, lending, pricing, claims, or care. The rest layer on from there. (Chapter 2)

The Nine Domains

1. Discovery & Accountability (Chapter 3)

Owner: the Chief Executive Officer, with day-to-day responsibility delegated to the Chief Information Officer or the Chief AI Officer.

In one sentence: every AI system running in the company has been found, across all three fronts, and every one of them has one named person with real authority over it, with decision rights written down and decisions made at the speed the business moves.

The executive question: When this system does something wrong, who owns the consequence, and did they know it before today?

Where most companies stand: Level 2. A committee and a charter exist; ownership reaches the Enterprise front and stops. Copilots on the workforce front and coding agents on the developer front run without a named owner, and when approvals are slow, teams route around them through personal accounts and unregistered tools.

What good looks like: For any AI system anywhere in the company, you can say who owns it, who approved it, under what conditions, whose authority it acts on, and when it was last looked at, in one query rather than one meeting. The register keeps itself current through the Proof Plane, so shadow AI surfaces as a gap instead of a surprise. Ownership sits with a role that always has a named holder, so it survives a reorg. The governance forum decides in days because it isn't reconstructing the same facts every time, which makes governance the fastest way to yes rather than a door to route around.

Start here:

- **Name the owners.** One executive for each of the nine domains, and one owner for every known production system, recorded as a role and its current holder.
- **Write the charter before the second meeting,** and split the work into governance, steering, and operational tiers so routine approvals never reach the full committee.
- **Tie every agent to a person.** For each agent, record whose authority it acts on and what credentials it holds.
- **Turn on continuous discovery** across all three fronts and feed it into the Proof Plane, so every system without an owner shows up as a gap.

2. Risk Management (Chapter 4)

Owner: the Chief Executive Officer; business leaders answer for each agent's outcomes, and the Chief Risk Officer (or Chief Information Security Officer) manages the risk itself.

In one sentence: a living function, not an annual spreadsheet, for knowing what each AI system could cost you and deciding on purpose how much risk you'll accept.

The executive question: Which of our AI systems could genuinely hurt us, and did we accept that risk on purpose or by accident?

Where most companies stand: Level 2. A process exists, inconsistently applied, and half the AI estate has never been through it. Every request gets the same review whatever its stakes, so low-risk tools route around the process while high-risk systems get no more scrutiny than a blog assistant. Tiers are self-reported at intake and rarely revisited.

What good looks like: The tier is the throttle, not a gate. Every AI use case is tiered at intake by consequence: what data it touches, what decisions it influences, and who it affects. The tier sets everything downstream: review depth, monitoring, and whose desk it lands on when something goes wrong. Low-consequence uses flow through in days on pre-approved patterns; high-consequence uses get real scrutiny. Tiers are built from live evidence of what a system can actually reach, not an intake form, and a material change triggers a re-tier automatically. Two desks, by design: the business owner answers for outcomes, and the risk function answers for whether the risk was seen, tiered, and tracked.

Start here:

- **Write the four-tier rubric** (unacceptable, high, limited, minimal) and design intake around it.
- **Sort the register by tier, or by its absence.** For every untiered system, ask what the worst plausible outcome is and who it happens to.
- **Reassess on events, not the calendar.** A new model, tool, data connection, or region triggers a fresh look.
- **Connect the register to the Proof Plane,** so agents built on approved platforms get tiered too, and a change in what a system can reach triggers review.

3. Protection and Resilience (Chapter 5)

Owner: the Chief Information Security Officer.

In one sentence: three areas of focus, not one: protecting the AI itself from attack, protecting the code the AI writes, and catching drift when nobody is attacking at all.

The executive question: If an attacker hit our AI systems this quarter, or a model just quietly stopped being right, would we find out from our own telemetry, or the hard way?

Where most companies stand: Level 3, and the number flatters. It describes the first area only. Two clocks are running: one starts when someone attacks, the other ticks even when nobody does. Nearly all the budget answers the first. Code written by AI is mostly ungoverned, and drift often has no owner at all.

What good looks like: Both clocks are watched. AI systems are red-teamed on a schedule using MITRE ATLAS techniques, and each agent runs inside a hardened harness with least privilege, so even a manipulated agent has little to reach. Code the AI writes is scanned and fixed at machine speed, and development is kept strictly apart from production. Every high-tier model has a defined drift threshold and a named owner who gets paged when it's crossed. For any high-tier system, you can produce its last adversarial test date and its current drift metric in the same conversation.

Start here:

- **Ask the two questions** for your highest-tier system: when was it last red-teamed with ATLAS techniques, and what is its drift threshold and who gets paged?
- **Scope every agent.** Give each agent least-privilege access, and put every coding assistant into the Proof Plane with its approved scope.
- **Protect the pipeline.** Strictly separate development from production and verify every package before it enters a build.
- **Measure response.** Start tracking time-to-find and time-to-fix for vulnerabilities in AI-written code.

4. Privacy and Compliance (Chapter 6)

Owner: the Chief Privacy Officer or the Chief Compliance Officer, with the Chief Information Security Officer as the operating partner who makes the promises provable.

In one sentence: using one intake door instead of two to keep every promise you've made about personal data, and to decide whether you're allowed to run a given AI system, on given data, in a given place at all.

The executive question: For every AI system we run, could we say exactly which privacy rules and AI-specific regimes apply, what our role is under each, and which personal data elements it touches, and prove, not just assert, that we're meeting all of them?

Where most companies stand: Level 2. A strong classical privacy program, built for systems of record and data subject requests, that AI data flows and AI-specific laws quietly walk around. Privacy policy is written for people, never translated into rules an agent can enforce. Agents pull personal data the privacy team never mapped, third parties ask for new data elements through back channels instead of contracts, and a system approved under one lawful basis drifts into another use unnoticed.

What good looks like: The rules didn't change for AI; what changes is that someone translates them into something an agent can obey. Privacy and compliance co-own a written ruleset for each agent, covering what data it may read, what counts as an instruction versus data it must never obey, and where its output may go, and engineering enforces it as a guardrail. The registry tracks each system's regimes, your role under each, its lawful basis, and its approved data elements. Regional obligations are applied as controls by geography and line of business. Retention, residency, and minimization are enforced on AI data paths by controls, not policy documents, and you know before launch whether a system needs a conformity assessment, a registration, a consumer notice, or nothing at all.

Start here:

- **One intake gate.** Merge privacy, compliance, and risk review into a single front door, so a new AI initiative is assessed once against every regime instead of three times by three teams.

- **One registry entry per system,** with its regimes, your role under each (controller, processor, deployer, provider), its lawful basis, and the personal data elements it's approved to use.
- **Write the one-page agent ruleset** for your highest-tier agent: what it may read, what it must never treat as an instruction, and where its output may go.
- **Reconcile and connect.** Check the registry against procurement records, single sign-on logs, and data-flow discovery, and connect it to the Proof Plane so any change to an approved data connection, element, or region raises a flag.

5. Transparency and Auditability (Chapter 7)

Owner: the owner of each agent or platform is accountable for proving what it did; the function is typically run by the Chief Information Security Officer, Chief Technology Officer, Chief Information Officer, or Chief Data Officer.

In one sentence: transparency is being able to describe what a system does; auditability is being able to prove, after the fact, exactly what it did; you need both, and the log is the proof.

The executive question: Could we prove to an auditor, a regulator, or a court, months after the fact, exactly what this system did and on what basis?

Where most companies stand: Level 2, and one of the three least mature domains. Logs exist, but they are scattered across tools, each telling part of the story, and nobody can reconstruct what an agent did from start to finish. Every audit and customer questionnaire starts from scratch.

What good looks like: This is where policy becomes proof. Every system shows up on the Proof Plane with clear answers to who owns it, what it's built on, what it can reach, and who can reach it, and anything outside its approved scope is blocked wherever it passes through a control point, rather than flagged for someone to notice eventually. Every system has one standard write-up (purpose, data, limitations, appropriate

use), written once and reused for every audit, questionnaire, and regulator. People know when they are dealing with AI. Every meaningful interaction is logged the moment it happens, in a place nobody can quietly edit, and your own team can pull any system's logs this afternoon without warning anyone.

Start here:

- **Capture four things** for every meaningful interaction (input, data touched, output, reviewer) in a tamper-evident store.
- **Set retention with counsel.**
- **Write one standard record per system** and have the owner sign it.
- **Test it cold.** Pull one system's logs without warning and see what comes back.

6. Data Visibility and Quality (Chapter 8)

Owner: the Chief Data Officer.

In one sentence: knowing what data feeds every AI system you run and where that data goes next, by seeing the data first, with deliberate manual design for the Enterprise front and continuous automated discovery and classification for everything else.

The executive question: Do we know what data feeds our AI and where our AI sends it, and would we stand behind every piece of it, and every place it ends up, in public?

Where most companies stand: Level 2 at best, often Level 1, and one of the three least mature domains. Most organizations have a rough picture of where data lives and almost no picture of where it moves. Data flows into AI systems that were never mapped, and back out to model providers, third parties, downstream agents, and other regions, with no one tracking the path. Duplicate and stale records sit underneath confident answers, and much of what gets blamed on the model is really a data problem.

What good looks like: Data is chosen on purpose, not whatever happened to be within reach. For every AI system, you can answer five questions on demand: what data do we have, where is it, who and what is accessing it, where is it going, and how do we use it to differentiate. Discovery and classification run continuously, down to the element level, across cloud, SaaS, and on-premises. Lineage is tracked in both directions, from where data came from to every model, index, agent, and destination it feeds. Quality checks are gates data must pass before use, and every dataset has an owner whose job is to fix it. The reward is speed: a new AI initiative gets a data-readiness answer in days, because the picture already exists.

Start here:

- **Answer the five questions in writing** for the data feeding your highest-tier system, and write down where you stall.
- **Turn on continuous discovery and classification,** and name an owner for every dataset it surfaces.
- **Map the outbound paths.** Identify where each AI system sends data (model providers, third parties, other agents, logs, other regions) and flag any destination that wasn't approved.
- **Link discovery to the Proof Plane,** so you can see which agent is touching which data right now, and where that data goes next.

7. Human Oversight (Chapter 9)

Owner: the Chief Operating Officer, or the business leader whose decisions the AI touches.

In one sentence: a person in the loop for every system that makes or shapes a consequential decision, including the ones nobody planned for, with a role that is defined, sized to the stakes, and tested.

The executive question: Where can a human still stop this system, and would that human know when to, and be positioned to see it in the first place?

Where most companies stand: Level 2. Designed-in oversight on the Enterprise front, rarely tested; none on the other two. A human in the loop is not a human in charge: reviewers approve at volume in seconds without seeing what the system saw, and stop mechanisms exist on paper but have never been exercised.

What good looks like: Every system that makes or shapes a consequential decision has a person whose role is defined, sized to the stakes, and tested. Each system has a written decision inventory: what it may do alone and what needs a person, for decisions that touch a customer, change production, or drive a business process. Where harm is irreversible, a person makes the call every time; where volume is high and confidence is calibrated, people review the uncertain cases plus a random sample. The named overseer has the competence, training, and authority to act, and a stop that can interrupt an action already in progress. Oversight is measured by what the human could have changed, not by whether one was in the chair.

Start here:

- **Write three oversight tiers** and map every consequential system to one, with a one-page decision inventory for each.
- **Name the reviewer and test the stop.** For each top-tier system, name who can stop it, and confirm the stop works under real conditions.
- **Test the reviewers.** Seed a synthetic error, one that can never reach a real person, into a reviewer's queue, and measure how long reviews actually take.
- **Use the Proof Plane** to find the consequential systems with no reviewer at all, and assign one or shut the system off.

8. Fairness & Non-Discrimination (Chapter 10)

Owner: the General Counsel, joined by the Chief Human Resources Officer for employment decisions.

In one sentence: clean data is necessary but is not a substitute for testing what a system does to people; someone has to choose a definition of fair, on the record, and test outcomes against it.

The executive question: Would this system treat two equally qualified people the same, and could we prove it to a plaintiff's attorney?

Where most companies stand: Level 2 at best, often Level 1; one of the three least mature domains. Fairness is assumed from clean data or a vendor's assurance. Few companies have chosen a definition on the record, and fewer test outcomes after launch.

What good looks like: Every system that makes or shapes a consequential decision about a person, in hiring, lending, pricing, claims, care, or discipline, is on a list. Each has a documented fairness definition, chosen by legal, statistical, and domain experts together, with the reasoning written down. Outcomes, not just inputs, are tested before launch and on a schedule after, because bias drifts the same way accuracy does, and models bought from vendors are held to the same standard as those built in-house. A remediation path with real teeth exists before it's needed, and where the law requires an independent audit, it is done on schedule.

Start here:

- **List every system** touching hiring, lending, pricing, claims, care, or discipline.
- **Put legal, statistical, and domain expertise in one room** to choose and document a fairness definition for each.
- **Test outcomes** before launch and on a schedule after, and require the same evidence from vendors.
- **Write the remediation path** before you need it.

9. Incident Response (Chapter 11)

Owner: the Chief Information Security Officer.

In one sentence: a plan built for the bad day when nothing was attacked and something was just wrong, rehearsed in advance, backed by the ability to reconstruct exactly what any agent did within an hour.

The executive question: When, not if, our AI does something wrong, how fast do we know, and what happens in the first hour?

Where most companies stand: Level 2. Teams are good at the incidents they've trained for but have never rehearsed the one with no attacker. Most AI playbooks are general IT playbooks repurposed, reconstructing what an agent did across tools takes days, and handling evidence across borders has never been worked out in advance.

What good looks like: AI incident types are defined and sorted by severity in advance, covering wrong or harmful outputs, data leakage, an agent acting outside its scope, and a model that drifts, so the first hour isn't spent arguing about categories. Telemetry watches for AI misbehavior. There is a tested way to pull a model or freeze an agent that works even when the system insists there's no way back. The Proof Plane reconstructs what any agent did, on any day, within an hour, including chains where one agent's output became another's input. Regulatory clocks and cross-border evidence handling are settled in advance, so solving a security incident never creates a legal one.

Start here:

- **Write the AI incident types and severities,** and name who can pull the plug on each system.
- **Record the regulatory notification clocks** for each system and each region it runs in.
- **Run a ninety-minute tabletop** around a failure that isn't an attack.
- **Time the reconstruction.** Measure how long the Proof Plane takes to reconstruct one agent's activity cold.

How to Report It

By position and trajectory. For each domain, report where you stand on the five-level scale and which direction you're moving. Position tells the board how exposed you are today. Trajectory tells them whether the program is working.

Chapter 1:
Governance on Paper

Picture the committee. Almost every company assembled one, right on schedule, the moment AI stopped being a curiosity and became something nobody could ignore: the Chief Information Officer, the Chief Technology Officer, the Chief Operating Officer, the Chief Financial Officer, the Chief Privacy Officer, and a compliance officer or two. The right people, exactly who the org chart says should be in that room. Their job was to sponsor the AI worth funding, review what was already underway, and approve or stop whatever came next. And they did what a responsible company is supposed to do. They wrote down goals. They circulated and signed real policies. They diagrammed a process every initiative was meant to walk through before it got anywhere near production.

On paper, that's a company doing everything right.

The Realization Every Committee Reaches

Sooner or later, every one of those rooms reached the same uncomfortable realization. It arrived through a different door each time. Some companies met it when copilots and chatbots landed in the hands of their own employees faster than any committee could keep up. Others met it while investing seriously at the enterprise level, standing up platforms and autonomous agents wired straight into core workflows. Still others met it inside engineering, where AI-assisted development was quietly changing how fast code shipped, whether or not anyone upstairs had noticed. Three doors. Hold onto them; the next chapter gives them names. What waited on the other side was the same in every case. The policy existed. The committee existed. The sign-off process existed. And almost none of it had any technical teeth underneath.

Policy tells people what they're supposed to do. It has nothing to say about what is happening: what data an agent can reach, which entitlements a copilot picked up along the way, what a development pipeline shipped straight into production while

the review board was still finding a date for its next meeting. The most dangerous governance program isn't the one that's obviously missing. It's the one that looks complete, with a charter, a committee, and a sign-off process all present and accounted for, and zero technical enforcement behind any of it. That can go on for a year or more, until an audit, a breach, or a regulator finally makes the one demand paper can't satisfy. *Show me.*

The committee was real. The control wasn't.

Governance you can't enforce isn't governance. It's a hope with a policy document attached.

That gap, between the governance a company writes down and the governance its environment can enforce, is the subject of this book. Policy still matters, and this book takes it seriously enough to build on it rather than argue around it. But policy alone was never going to close the gap, and every company that assembled the right committee eventually learns that, usually the hard way.

Five Right Answers, No Whole Picture

There's a version of this problem that's easy to miss because from a distance it looks solved. Ask who owns AI at most companies and you'll get an answer delivered with real conviction. Ask five executives and you'll get five different, equally confident answers. The Chief Information Officer owns the deployment and the governance program wrapped around it. The Chief Financial Officer owns the spend and the return that's supposed to justify it. The Chief Human Resources Officer owns whether the workforce can use any of it. The Chief Operating Officer owns whatever operational outcome it was funded to improve. The Chief Executive Officer owns the strategy meant to tie it all together. None of them is wrong. None of them has the whole picture, either.

Hiring is supposed to fix that. Whatever the title ends up being (Chief AI Officer or Chief Data and AI Officer), plenty of companies have filled a seat like this in the last couple of years. But a new title fixes fragmented ownership only if the fragmentation was a staffing problem in the first place, and the research says

it mostly wasn't. In Cynozure's February 2026 survey of about sixty senior data and AI leaders, 40 percent said AI strategy ownership at their company is still split across multiple executives, and 17 percent said there is no clear owner at all. IBM's research is blunter: in its June 2026 study, two out of three Chief Information Officers and Chief Technology Officers surveyed said they're accountable for AI systems they don't fully control. A new seat at the table doesn't consolidate authority that was never centralized to begin with. It adds a sixth confident answer to the five you already had. Surveys of how many large companies have a Chief AI Officer range from under 40 percent in one benchmark to three in four in another, and they aren't measuring the same thing: each surveys a different group and defines the role differently. None of the research claims the title alone fixes this. The accountability gaps show up whether or not the seat exists.

So the committee has two problems, not one. It can't see what's happening underneath the policy it wrote. And even if it could, nobody in the room owns the answer, because five people each own a piece of it.

Neither problem is solved by a bigger committee or a stricter policy. Both are solved by the same three things, and they're what the next chapter is for: one framework that turns policy into something a machine can check against every standard you've mapped it to, one named owner for each part of it, and one way to see every piece of AI running in the company, not just the systems that walked through the committee's door.

Chapter 2:
One Framework Instead of Dozens: The Common Control Framework

Ask your general counsel which AI rules the company has to follow, and watch what happens. The answer starts confidently, with two or three names. Then it slows down. Then it turns into a list. Then it turns into a shrug. That isn't a failing of your general counsel. The Organisation for Economic Co-operation and Development keeps the closest thing to a global tally. It now counts more than 900 national AI policies and initiatives across more than 80 jurisdictions, before you add the sector rules, the state laws, and the industry codes stacked on top. Nobody can hold that in their head. The good news is that nobody needs to.

After enough meetings in these rooms, I've noticed that most companies end up picking from a short list. Not because anyone told them to, but because the same handful of names keep coming up in board questions, customer questionnaires, and regulator letters. Five show up more than the rest. Each one is good at something. Each one also sees only a slice of your problem, and if you anchor your whole program to just one of them, you inherit its blind spots as your own. Anchor to the wrong subset and you'll be starting over in eighteen months, when the next one lands.

Five Frameworks Companies Keep Picking

These five aren't a canon. Four of them are voluntary: companies adopt them because they're useful, or because a customer, an auditor, or a contract expects to see one. The fifth, the EU AI Act, is law, and where it applies to you, it isn't optional. They're simply the ones I see companies reaching for, over and over, so they're worth knowing well enough to recognize when one of them walks into a meeting.

The NIST AI Risk Management Framework. Published by the U.S. National Institute of Standards and Technology, this is the

broadest of the five and, honestly, the best thinking tool. It's voluntary guidance built around four verbs (govern, map, measure, manage), and it's very good at making you ask the right questions. Its weakness is the flip side of being voluntary: no certificate, no enforcement, no clean line you can point to and say "done." Your risk and audit leaders will lean on it most.

The EU AI Act. This is the opposite kind of animal. It's law, with penalties that reach a meaningful slice of global revenue, and it sorts AI systems into risk tiers with real obligations attached to the high ones. Its weakness is shape. Its reach follows the EU market, not your headquarters, so it can bind a company with no office in Europe if its AI is offered or used there. And its heaviest obligations attach to specific categories: prohibited practices, high-risk systems, and general-purpose models, plus a set of transparency duties. For most of the everyday AI in your estate, it asks comparatively little. Your general counsel and compliance officer will live with it most.

ISO/IEC 42001. This is the management-system standard for AI, the AI cousin of the ISO/IEC 27001 certificate your security team already holds. It's certifiable, it's audit-friendly, and procurement teams are starting to ask for it by name. Its weakness is the classic management-system weakness: it checks whether your process exists and is followed, not whether your AI is safe. Whoever answers your customers' security questionnaires will care about it most.

MITRE ATLAS. The narrowest lens of the five, and the sharpest. ATLAS is a catalog of the tactics attackers use against AI systems, laid out the same way MITRE's better-known ATT&CK matrix lays out traditional attacks. It speaks directly to a few of the nine domains in this book and goes quiet on the rest, because it was never built to cover fairness, transparency, or accountability. Your Chief Information Security Officer and red team will use it most.

OWASP's AI security work. The Open Worldwide Application Security Project, the same community that has published the web-security Top 10 for two decades, now maintains a Top 10 for large language model applications and a newer Top 10 for agentic applications, alongside a broader AI Exchange. This

book maps to the 2025 edition of the LLM list and the agentic list released in December 2025. It's the most practical of the five for the people building things: a plain list of the ways these systems get broken, from prompt injection to agents given more authority than they should have. Its weakness is that it's a developer's checklist, not a governance program. It tells you what to defend against, not who's accountable when you don't. Your engineering leads will reach for it first.

Now, look at what those five have in common. Underneath the different vocabularies, they're circling the same concerns: who's accountable, how risk gets assessed, how decisions get explained, how data gets governed, what happens when something breaks. New frameworks keep arriving because every jurisdiction and industry body wants its own name on the wall. Almost none of them are inventing new concerns. They're redescribing the same handful in their own language, backed by their own enforcement mechanism.

That's the opening you need. If you map your controls to the concern rather than to the vocabulary, the vocabulary stops mattering. Regulators keep changing their language. The concerns don't.

Nine Domains, One Crosswalk

This book calls that approach the Common Control Framework: nine underlying disciplines, built once, mapped outward to whichever standard happens to be asking, whether that's one of the five above, one of the dozens behind them, or whatever a regulator invents next year. The mapping itself is called a crosswalk: a maintained table that says, for each of your controls, exactly which clause, article, or requirement of each outside framework it satisfies. Build the control once, and the crosswalk answers the "does this satisfy X" question for every framework it covers. When a framework changes or a new one arrives, you update the crosswalk, not the program.

Every one of the nine domains needs an executive who answers for it, a person rather than a committee. The owners below are the ones that fit most large companies. Yours may

differ, and that's fine, as long as every domain has exactly one name beside it and that person knows it.

1. **Discovery & Accountability**: the Chief Executive Officer owns it and delegates the day-to-day running of it to the Chief Information Officer or, where one exists, the Chief AI Officer

2. **Risk Management**: the Chief Executive Officer owns it; business leaders answer for each agent's outcomes, and the Chief Risk Officer, or the Chief Information Security Officer where there isn't a Chief Risk Officer, manages the risk itself

3. **Protection and Resilience**: the Chief Information Security Officer

4. **Privacy and Compliance**: the Chief Privacy Officer or the Chief Compliance Officer

5. **Transparency and Auditability**: the owner of each enterprise agent or platform is accountable for proving what it did; the function is typically run by the Chief Information Security Officer, Chief Technology Officer, Chief Information Officer, or Chief Data Officer, depending on the company

6. **Data Visibility and Quality**: the Chief Data Officer

7. **Human Oversight**: the Chief Operating Officer, or the business leader whose decisions the AI touches

8. **Fairness & Non-Discrimination**: the General Counsel, joined by the Chief Human Resources Officer wherever employment decisions are involved

9. **Incident Response**: the Chief Information Security Officer

Laid out, the crosswalk looks like this. Treat the table as a starting point, not a legal opinion. The references are at the article and clause level so your team can verify each cell against the current text before relying on it. Add columns for whatever else binds your company. In the table, a cited article or clause

means the framework directly addresses that domain; a short description means it supports the domain without requiring it; “Primary” marks a framework’s home territory; and “Not covered” means the framework doesn’t speak to the domain.

Domain	NIST AI RMF	EU AI Act	ISO/IEC 42001	MITRE ATLAS	OWASP
Discovery & Accountability	Govern (policies, roles, accountability)	Art. 4 (AI literacy), Art. 17 (quality management system), Art. 26 (deployer obligations)	Clause 5 (leadership, roles); Annex A.2–A.3 (policies, internal organization)	Not covered	Not covered
Risk Management	Map and Manage (context, risk identification, treatment)	Art. 6 and Annex III (classification), Art. 9 (risk management system)	Clause 6.1, 8.2–8.4 (risk assessment, treatment, impact assessment); Annex A.5	Threat modeling input	Top 10 lists as risk inventory
Protection and Resilience	Measure (security, resilience, robustness)	Art. 15 (accuracy, robustness, cybersecurity)	Annex A.6 (system lifecycle, security); relies on ISO/IEC 27001	Primary: full matrix	Primary: LLM and Agentic Top 10
Privacy and Compliance	Govern and Measure (legal requirements, privacy risk)	Art. 10 (data governance), Art. 43 (conformity assessment), Art. 47–49 (declaration, CE marking, registration); GDPR alongside	Clause 4.2 (interested parties, legal requirements); Annex A.7 (data for AI)	Not covered	Sensitive information disclosure
Transparency and Auditability	Govern and Measure (documentation, transparency, accountability)	Art. 11 and Annex IV (technical documentation), Art. 12 (record-keeping), Art. 13 (transparency to deployers), Art. 50 (disclosure)	Annex A.6.2.7 (technical documentation), A.6.2.8 (event logs), A.8 (information for interested parties)	Logging supports detection	Rogue agents; cascading failures (detection depends on logging)

Domain	NIST AI RMF	EU AI Act	ISO/IEC 42001	MITRE ATLAS	OWASP
Data Visibility and Quality	Map and Measure (data provenance, dataset evaluation)	Art. 10 (data and data governance)	Annex A.7.2–A.7.6 (data management, acquisition, quality, provenance, preparation)	Data poisoning techniques	Data and model poisoning; supply chain
Human Oversight	Govern and Manage (human-AI configuration, oversight roles)	Art. 14 (human oversight), Art. 26 (deployer oversight assignment)	Annex A.9 (responsible use of AI systems)	Not covered	Excessive agency; human-in-the-loop controls
Fairness & Non-Discrimination	Measure (fairness and bias evaluation)	Art. 10(2) (bias examination), Art. 27 (fundamental rights impact assessment); EEOC guidance and NYC Local Law 144 alongside	Annex A.5 (impact on individuals and groups), A.7.4 (data quality)	Not covered	Not covered
Incident Response	Manage (post-deployment monitoring, incident handling)	Art. 72 (post-market monitoring), Art. 73 (serious incident reporting)	Clause 10 (nonconformity, improvement); Annex A.8.4 (incident communication)	Supports detection (adversary techniques to watch for)	Detection and response guidance

Now take one well-built control and read across the row. Say every AI use case gets tiered by consequence at intake and its data sources get documented with lineage and classification before anyone approves it for deployment. That one control feeds NIST's mapping and governing functions. It produces a core part of the technical documentation the EU AI Act requires for high-risk systems. It's exactly what an ISO/IEC 42001 auditor asks to see. It gives your red team the data lineage they need to test for the data-poisoning techniques MITRE ATLAS catalogs. And it gives your OWASP-minded engineers the data-provenance answer they need. One control, five lenses, one set of evidence. The same evidence usually carries over to the other frameworks you track, because the underlying discipline rarely changes. The crosswalk records where it does and where it doesn't. You answer the question once instead of every time someone asks it.

Three Doors, Not One

Before the nine domains can mean anything, you need one more idea because it explains why the committee in the last chapter kept losing. AI doesn't enter a company through one door. It enters through three, and each one has to be governed differently.

The first door is **the Enterprise front**: the AI your company builds on purpose. A strategic initiative with a budget, a sponsor, a timeline, and its own slide in the board deck. This is the AI the committee was designed for, and here, deliberate, human-led governance works. The number of systems is small, the stakes are high, and each one deserves a person thinking carefully about it.

The second door is **the workforce front**: the copilots, chatbots, and AI features that show up on employees' desktops. Some sanctioned, some through personal accounts, many bundled into software you already pay for and never procured as AI at all. Nobody files an intake request for these. They arrive by the hundreds, and they arrive quietly.

The third door is **the developer front**: the coding assistants and agents your engineers use to build and ship software,

increasingly wired straight into the pipelines that put code into production. This one moves fastest, and it's the door most governance programs haven't noticed is open.

Every domain in this book applies to all three fronts. But the way you govern each front is different, and that difference is the book's running argument. What works for the Enterprise front (committees, intake forms, and deliberate review) cannot keep pace with the other two. Pretending it can is how a company ends up with a governance program that governs a shrinking fraction of its AI and feels fine about it right up until the audit.

The Proof Plane

Which brings us to the one piece of infrastructure every domain in this book depends on. I'm going to call it the Proof Plane and explain it here so that later chapters can point back to it.

First, what it is and isn't. The Proof Plane is a reference architecture, an operating model for how your security and governance capabilities should work together. You can try to assemble it from a dozen point tools: one for discovery, one for data classification, one for access, one for AI traffic, one for logging. Most organizations that try end up with a dozen partial maps that disagree with each other, and no one owns the seams.

The better path is to anchor the Proof Plane in a platform built around the data itself. Agents come and go, change models, and pick up new tools weekly; the data they touch is what persists, and it's where the risk actually lives. Look for a platform that can discover and classify your data down to the element level across cloud, SaaS, and on-premises environments; see every identity that can reach that data, human or agent; discover the AI systems and agents in use, sanctioned or not, and track which data flows into them; and enforce policy at the data layer when something drifts out of scope. Connect that platform to your identity provider, your AI gateways, and your tamper-evident logging, and you have the backbone of the Proof Plane. No platform covers every corner on its own, but the right one gives you a single, authoritative map to build from instead of a dozen you have to reconcile.

Picture what it looks like when those pieces are connected: a living map of every AI system and agent running in your company. Not just the ones somebody remembered to register, but as close to all of them as your visibility allows, across all three fronts, kept current by the map itself rather than by anyone filling out a form. For every system on that map, the Proof Plane maintains a short list of facts: who owns it and who built it. What model or foundation it runs on. What tools, skills, and knowledge sources it can use, which is to say what's in its harness. What data, down to the specific elements, it's connected to. Who can reach it, and what it can reach. What scope it was approved for. It also holds a tamper-evident record of everything material it can observe that system doing.

The Proof Plane does more than describe. It compares what an agent is doing today against what it was approved to do, and wherever an agent's activity flows through a control point, it holds the agent to that approved scope automatically. An agent that quietly picks up a new data connection or a new permission gets caught when it happens, not at next year's audit.

Be clear-eyed about the limits. Even anchored in the right platform, the Proof Plane can only see what your telemetry reaches. Agents running on unmanaged devices or personal accounts, AI features embedded inside third-party SaaS that expose little to you, and activity that never crosses a control point will show up partially or not at all. Coverage is something you grow over time, not something you switch on. When later chapters say the Proof Plane "knows" or "enforces" something, read that as: to the extent your integrations cover it. The gaps themselves are worth tracking, because a known blind spot is a governable risk and an unknown one is not.

You'll meet the Proof Plane in nearly every chapter from here on, because each domain asks it a different question. Risk Management asks which systems could hurt you. Privacy asks whether each system still matches the lawful basis it was approved under. Transparency asks for the log. Human Oversight asks where a person needs to be standing. Incident Response asks what an agent did in the hour before something broke.

One map, nine questions. From this point forward, each chapter will just tell you what it needs from it.

How Mature Is "Mature"?

Each chapter that follows tells you where most large companies stand today in that domain, using a five-level scale you've probably seen in other parts of your business. It's worth defining once so the numbers mean something when you encounter them.

Level 1: Initial. Nothing formal exists. Whatever governance happens depends on someone noticing a problem.

Level 2: Repeatable. A policy and a process exist and are followed some of the time, for some of the AI, usually the Enterprise front. Enforcement is manual, and nobody has measured how much is slipping past.

Level 3: Defined. The process is standardized, documented, and applied consistently across the AI the company knows about. Evidence exists, though producing it still takes real effort.

Level 4: Managed. Controls are enforced technically, not just written down, and coverage is measured across all three fronts. The Proof Plane is in place. Drift from approved scope is caught automatically.

Level 5: Optimizing. The program improves itself. Findings feed back into policy, thresholds are tuned against real outcomes, and the committee spends its time on risk appetite and strategy because the machinery handles the paperwork.

These ratings are my practitioner assessments, drawn from years of building these programs and from conversations with peers across dozens of large companies, not the findings of a formal study. Read them as a benchmark to test yourself against, not a statistic. In my experience, most large companies sit at Level 2 on most domains. Protection and Resilience is typically the one exception, at Level 3, for reasons Chapter 5 explains, and even there it isn't where it needs to be. Three domains are chronically the least mature of the nine: Data

Visibility and Quality, Transparency and Auditability, and Fairness & Non-Discrimination. Hold onto that as you read the next section.

Reading Order Isn't Build Order

The chapters ahead follow the numbered list above, and that order is deliberate: it's the order in which the ideas depend on each other. Risk management means nothing without an accountable owner in place first. Protection can't tell you what it's protecting until risk appetite has been decided. Human oversight can't be sized until you can see the data and the decisions underneath it. Read front to back, and each domain makes sense in light of the one before it.

But that isn't the order you should build in, and it's worth saying so now, before the two get confused. If you asked me for a plan, here's the one I'd hand you. Start with two things at once. Name the owners and stand up the council (Chapter 3), because nothing moves without a person accountable for it. And start continuous discovery across all three fronts (the Proof Plane), because you cannot govern what you can't see. Then go straight at the two weakest domains that almost everything else stands on: Data Visibility and Quality, in Chapter 8, and Transparency and Auditability, in Chapter 7. Intake and tiering, Chapter 4, come next, because now you have something real to tier. Fairness, Chapter 10, follows close behind if your AI touches hiring, lending, pricing, claims, or care, because it depends on data and transparency both being in place. The rest layer on from there. Each chapter closes with a plan for its first thirty and first ninety days. Those clocks start when you begin work on that domain, not on day one of the program. Read them as nine plans that follow the order above, overlapping where you have the people to run them in parallel, not as a single ninety-day project.

Read in dependency order. Build in risk order. Keep the two straight, and the rest of the book will feel less like a checklist and more like a sequence.

One control, several lenses, one set of evidence. Build it once, map it everywhere, and keep the map current.

Let's take these nine domains one at a time, starting with the one most committees believe they've already solved, simply because they formed one: Discovery & Accountability.

Chapter 3:
Discovery & Accountability

Who owns this domain: the Chief Executive Officer, who delegates the day-to-day running of it to the Chief Information Officer or, where one exists, the Chief AI Officer.

You cannot hold anyone accountable for an AI system you don't know exists. That sentence explains why this domain has two parts to its name. Discovery comes first: finding every AI system and agent running in the company, across all three fronts, whether or not anyone registered it. Accountability comes second: attaching one named person with real authority to every one of them. Most companies start with the second, skip the first, and then wonder why their governance program only ever seems to cover the AI that asked permission.

Almost every company that takes AI governance seriously starts in the same place, and almost every one of them makes the same mistake: they form a committee, and they mistake the committee for the work. A charter gets written. A standing meeting gets scheduled. Minutes get taken, a responsibility chart gets built, and someone's name goes at the top of an org box labeled "AI Governance." All of that produces real artifacts, and none of it answers the one question that matters the day something goes wrong: who owns the consequence?

Real governance means something narrower and harder than a committee. It means every AI system running in production has one named human being with real authority over it, not a committee that glanced at it once, months ago, at intake. It means decision rights are written down: who can approve each tier of use case, who has the authority to shut a system off, who's the one standing in the room when a regulator calls. And it means all of that happens at the speed the business moves, not on the quarterly rhythm of a review board that meets whether or not anything urgent is happening.

The Board That Became the Problem

I learned this one up close, at a large company that had done everything by the book. On paper, it looked like everything a governance program should be: a review board, a real set of policies, a defined path every AI initiative had to walk before it reached production. Security had a seat at the table. Compliance had a seat. Data had a seat. On the org chart, it looked like exactly the kind of program this chapter is supposed to describe.

What it didn't have was any technology underneath it. When a new initiative came in the door, somebody still had to go find out, by hand, what data it touched, who had access to that data, and where it lived. Those facts had to be reassembled from scratch for every review because nothing kept that information current between meetings. So the board didn't produce decisions. It produced findings: weeks of waiting, followed by a list of open questions instead of a yes or a no anybody could act on. And by the time anything reached the board, the sensitive data it was supposed to be reviewing had usually already found its way into a spreadsheet somewhere, or into a half-built agent nobody had ever mapped. The company had built a governance program that had quietly become the exact failure it existed to prevent: a very thorough, very well-intentioned process for approving things after they'd already happened.

One more thing about that board. It only ever saw the Enterprise front. Every initiative that walked through its door was a funded, sponsored project someone had chosen to submit. The workforce front and the developer front, the tools employees were adopting on their own and the assistants engineers were wiring into their pipelines, never came through the door at all, because nobody thought of them as AI initiatives. They were just tools. That board governed the AI that asked permission, and had no idea how much AI wasn't asking.

What Good Looks Like

The fix isn't a better committee. It's knowing, for any AI capability anywhere in the company, who owns it, who approved it,

under what conditions, and when it was last looked at, and being able to answer that in one query rather than one meeting. It's a governance forum that decides in days because it isn't waiting on someone to reconstruct the same facts every time. And it's ownership that survives a reorg. Accountability sits with a role, the role always has a named person in it, and when that person moves on, a defined handover passes it to whoever takes the seat. It doesn't walk out the door with them.

This is the Proof Plane from Chapter 2 doing its first job. The Proof Plane already knows, for every system across all three fronts, what it is, what it was built to do, and what data it draws on. What this domain adds is the owner: a name attached to every entry, automatically, so that a system with no owner shows up as a finding rather than a blank cell. Miss that one piece and the rest of the map quietly breaks. A map without owners is a list nobody's accountable for. An owner who can't see what data an agent is drawing on today is accountable in name only, vouching for something they can't see into, the same way that board was vouching for reviews built on whatever someone happened to find by asking around.

The AI Nobody Remembers Approving

A question worth asking in your next leadership meeting: who owns the AI features already sitting inside your customer relationship management system, the ones nobody procured, that arrived bundled into a routine software update? If the room goes quiet, you've found the exact gap this chapter is about. This is the workforce front at its purest. These systems never walked through anyone's door. Nobody filed an intake request for them. And they're touching the same customer data as anything your board spent six weeks formally approving.

How to Build It

If I were standing that board up again today, I'd build the room differently before I scheduled the first meeting. Keep the voting core small enough to convene in days, not weeks: a chair (whoever in your company is closest to owning AI strategy), alongside legal, risk, security, and privacy, plus whoever runs the part of the business using the system in question. Write the

charter before the second meeting, not the fifth: what's in scope, what the committee can approve outright versus what has to go further up to the board, how many people need to be in the room for a decision to count, how often you meet, and who sees the output.

Then do the thing that board never did: stop routing everything through the one room. Split the work into three tiers instead of one. A governance tier, the committee itself, meets quarterly, and its job is policy and risk appetite, not paperwork. A steering tier, meeting every week or two, keeps the inventory and risk register current and turns that policy into standards people can follow. And an operational tier, the teams building things, executes within whatever's already been pre-approved, no meeting required. Most day-to-day approvals should never reach the full committee. If everything does, you've rebuilt that review board under a new name.

Give the owner of a system, not just the committee, the standing authority to suspend it in a genuine emergency, with the committee ratifying that call within 48 hours rather than having to approve it in advance. The fastest way to guarantee your committee gets ignored during a real incident is to make it the only path to stopping something.

And point the Proof Plane at the two fronts your board can't see. Here's the rule I suggest: every system it discovers without an owner gets one within thirty days or gets turned off. Treat that as a starting point, not a standard. Design your own window and triggers around your company's risk appetite. Whatever you choose, make the window an outer limit, not a grace period. A system that touches regulated data, shapes consequential decisions, or looks like an unacceptable use should be restricted or suspended the day it's found, and stay that way until someone owns it. That single rule, enforced, does more for accountability than any charter language.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable. There's a committee, there's a charter, and ownership reaches the Enterprise front and stops there.

The Executive Question

When this system does something wrong, who owns the consequence, and did they know it before today?

Pick one AI capability at random this week and time how long it takes your team to tell you who owns it, who approved it, and when it was last reviewed. If the answer isn't minutes, you've found your starting point, the same one I watched that board find, the hard way.

Build It

In the first thirty days. Name one executive owner for each of the nine domains, using the list in Chapter 2 as your starting point. Name one accountable owner for every production AI system you already know about: a role, and the person who holds it today. Pick the council's voting core, and write the first version of the charter: scope, what the council can approve versus what escalates, quorum, cadence, and who sees the output.

In the first ninety days. Split the work into the three tiers (governance, steering, operational) and move routine approvals out of the full committee. Grant system owners emergency suspension authority with 48-hour ratification, and rehearse it once. Start continuous discovery across all three fronts, and apply the thirty-day rule to everything it finds without an owner.

The evidence you should be able to produce. A signed charter. An inventory in which every entry, including the ones discovery found, has an owner recorded as both a role and its current holder, with a handover step for when the holder changes. A decision log for the council showing turnaround measured in days. And a record of at least one suspension, drilled or real, ratified within 48 hours.

Once you know who owns what, the next question is what it could cost you if it goes wrong. That's Risk Management.

Chapter 4:
Risk Management

Who owns this domain: the Chief Executive Officer. Accountability for each agent's business outcome is delegated to the business leader who owns that outcome. Management of the risk itself, meaning the tiering, the register, and the reassessment, is delegated to the Chief Risk Officer, or to the Chief Information Security Officer where there isn't one.

They were doing it right. That's what makes the story worth telling. At one company I worked with, the AI team had started experimenting deliberately: building agents on foundation models through a controlled cloud service, testing against synthetic data in isolated, lower-level environments, exactly the way you're supposed to. Sanctioned, sponsored, reviewed. The Enterprise front, governed the way the Enterprise front should be.

What nobody had accounted for was how contagious ease of use would turn out to be. Product teams watched those sanctioned experiments succeed and quietly built their own agents to solve real business problems, wiring them into production data without ever routing through the governance process built to catch exactly this. Nobody was hiding anything. They'd simply seen something work and copied it, the way good teams do. The company caught it early only because a discovery process was already running in the background, not because anyone came forward. And it didn't respond by banning the experimentation that had worked. It built the automated detection and control that should have existed before the first ungoverned agent went live.

That's the whole of this domain in one story. Risk Management is knowing, before, during, and after deployment, what each AI system could cost you (in money, customers, regulatory exposure, and trust) and deciding on purpose how much of that you'll accept. Not a spreadsheet or a system updated annually. A living function wired into how AI enters the enterprise, through all three doors.

Why an Annual Spreadsheet No Longer Works

That living function has a scale problem most committees haven't priced in yet. Gartner forecasts that by 2028 the average global Fortune 500 enterprise will have more than 150,000 AI agents in use, up from fewer than 15 in 2025. In a separate forecast, it expects 40 percent of enterprise applications to feature task-specific agents by the end of 2026, up from under 5 percent the year before. These are projections, and the counts depend on how broadly "agent" is defined, but the direction isn't in doubt. Risk management built around an annual review cycle and a spreadsheet was never going to keep pace with that curve, and it shows: in the same April 2026 release, Gartner reports that only 13 percent of organizations think they have the right AI agent governance in place.

The arithmetic is unforgiving. A register that reflects only what comes through the intake form voluntarily is a register of the Enterprise front and nothing else. In UpGuard's 2025 survey of about a thousand employees in the US and UK, 81 percent said they use AI tools their company hasn't approved. Separately, researchers monitoring enterprise traffic have catalogued more than six hundred distinct AI tools in active use across the companies they watch, most of which were adopted by people who hadn't so much bypassed a review process as never known it existed. The agent a developer wired into the build pipeline last month and the tool an employee is running through a personal account are both touching real data and real systems. Neither is in your register, because neither one asked.

What Good Looks Like

Good risk management starts with intake, and intake starts with a tiering rubric. Every AI use case that comes through the door gets sorted by consequence (what data it touches, what decisions it influences, who it affects), and the tier determines everything downstream: how deep the review goes, how closely the system is monitored once it's live, and whose desk it lands on when something goes wrong. Low-consequence uses flow through in days on pre-approved patterns. High-consequence uses get real scrutiny. And there are two desks,

by design. The business leader who owns the outcome answers for what the agent does. The risk function answers for whether the risk was seen, tiered, and tracked. Blur those two, and you get a risk team that owns consequences it can't control and business owners who assume someone else is watching.

There's a trap that catches even well-run intake processes. When a company approves a platform that lets employees build their own agents (a low-code agent builder, an enterprise copilot with custom-agent features, an internal tool your developers stood up), the platform goes through intake once, gets a tier, and gets approved. Then hundreds of people build hundreds of agents on it, and not one of those agents ever comes back through the door. The form tiered the tool. It said nothing about the agent someone in finance built last Tuesday that reads the general ledger, or the one in sales that pulls customer contracts into a chat window. Each of those is its own use case with its own consequence, and intake never saw any of them. Approving the platform is necessary. It is not the same as governing what gets built on it, and no intake process, however good, can do that by hand at the rate people build.

Then the discipline keeps working after launch. The risk register changes when the systems change, not once a year. Risk gets reassessed after incidents and at decommissioning, not just at launch. And the register covers the systems that never came through the door because the Proof Plane feeds it the ones discovery found, including every agent built on a platform you already approved. An unsanctioned agent that poses unacceptable risk shows up as a finding the week it appears, not as a surprise during next year's audit.

How to Build It

Build the tiering rubric before you build the form. Use four tiers, borrowed from the structure the EU AI Act uses, even if you never operate in Europe. This is your internal rubric, not a legal classification. It borrows the Act's shape, but whether a system is prohibited or high-risk under the Act depends on its intended use and on the Act's own definitions and annexes, and that determination belongs to counsel. Unacceptable: no

approval path. Confirm the classification, then stop it. Social scoring, covert manipulation, exploiting vulnerable groups. **High**: touches someone's legal status, safety, livelihood, or rights (hiring, credit, healthcare, biometric identification) and gets the deepest review, meaning documented risk management, audit logs, a human in the loop, bias testing, and sign-off before launch. **Limited**: no rights-affecting decision, but a person interacts with AI-generated content or a bot. Requires disclosure and lighter documentation. Minimal: everything else, such as most internal productivity tooling, once its intended use has been checked against the higher tiers. Register it, apply baseline controls, move on. The NIST framework calls the first step of this *Map*: documenting intended use, affected stakeholders, and foreseeable misuse before you assign a tier, not after.

Design the intake questionnaire around what the rubric needs to know (what decision the system influences, who's affected, what data feeds it, whether a wrong answer is reversible) and route by tier automatically. Minimal and limited use cases should self-certify against a checklist. Only high-tier use cases should ever reach a full cross-functional review. A use case flagged as unacceptable gets a quick check too, but only to confirm the classification; if it holds, the use stops. Mastercard's approach is a useful real-world model: a single intake questionnaire mapped to its own risk categorization, auto-routing submissions to governance, security, legal, privacy, and brand reviewers, feeding one central registry instead of a spreadsheet several teams update by hand.

Wire reassessment to events, not a calendar. A model version change, a new data source, an incident or near-miss, a regulatory update, or a decommissioning should each automatically reopen that system's entry for re-tiering. An annual review can't see any of those the week they happen, and the week they happen is exactly when it matters.

What the Proof Plane Does Here

Everything above governs the front door. The Proof Plane governs the other two. It continuously tracks every AI agent and AI-driven workflow running in the enterprise, sanctioned and

unsanctioned alike, including every agent built on a platform you approved. For each one, it knows what it was built to do and what data it's drawing on right now, not just what it was approved to touch at intake. Anything it finds that isn't in the register gets tiered by the same rubric, automatically, and anything whose behavior or data access drifts from its approved scope triggers re-tiering the day it happens. McKinsey puts a number behind why that matters: 80 percent of organizations report having already encountered risky agent behavior, including agents reaching data or systems beyond what they were authorized for.

Gartner calls this category AI trust, risk, and security management: tooling built to discover agents across sanctioned and shadow AI, manage their identities and permissions, monitor behavior for policy compliance and anomalies, and govern what data they can reach. It's less a single product than a checklist for what any serious AI risk program now has to cover. Automation isn't optional here. It's the only way to keep pace with how fast AI is spreading through the workforce and your own development pipelines, where adoption is growing faster than any review board ever will.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable. A process exists, usually born in a hurry after some scare, but it's inconsistently applied and half the AI estate has never been through it. A risk register that gets touched only during audit season isn't managing risk. It's documenting it after the fact.

The Executive Question

Which of our AI systems could genuinely hurt us, and did we accept that risk on purpose or by accident?

Ask your risk team for the register this week and count two things: how many systems are on it, and how many are running according to the Proof Plane or whatever discovery you have. The gap between those two numbers is your unmanaged risk, and most companies have never measured it.

Build It

In the first thirty days. Write the four-tier rubric and get the council to sign it. Draft the intake questions around what the rubric needs to know, and decide which tiers self-certify, which go to review, and who confirms an unacceptable classification and stops the use. Tier every AI system you already know about, and for every approved agent-building platform, write the rules for what may be built on it: which data sources are in bounds, and which tiers of decision require review before an agent goes live.

In the first ninety days. Stand up the register as a single system of record, not a spreadsheet, and connect it to the Proof Plane so discovered systems are tiered automatically. Wire the five reassessment triggers (model change, new data source, incident, regulatory update, decommissioning) so they reopen a system's entry without anyone having to remember to. Run the self-certification path for minimal and limited use cases and confirm they clear in days.

The evidence you should be able to produce. A signed rubric. A register in which every known system carries a tier and an owner. For each approved agent-building platform, a count of the agents built on it, each carrying its own tier. A list of active reassessment triggers, and at least one system that was re-tiered because a trigger fired. And a discovery report showing how many systems the Proof Plane found that weren't in the register, and how long each took to get tiered.

Knowing which systems could hurt you, and whether you accepted that risk on purpose or by accident, is the foundation. But none of it holds up if the system itself can't withstand an attacker or quietly degrades in production without anyone noticing. That's Protection and Resilience.

Chapter 5
Protection and Resilience

Who owns this domain: the Chief Information Security Officer.

By now, you've built a risk register that says which AI systems could hurt you, and you've decided on purpose which risks you're accepting. That discipline matters, but none of it holds up if the system can't withstand someone trying to break it, if the code it writes ships with holes in it, or if it quietly drifts away from the behavior it was approved for without anyone noticing. Protection and Resilience is where the risk register meets reality.

Three Areas of Focus

This domain asks you to protect three different things, and most programs work only on the first.

The first is protecting the AI itself: keeping adversaries from poisoning your training data, manipulating your inputs, extracting your models or secrets, or hijacking the agents built on top of them. This is MITRE ATLAS territory, and it's where nearly all the attention has gone.

The second is protecting what the AI builds. On the developer front, AI is no longer just a tool that helps engineers write code. At some companies it's already writing most of the new code, and the share is rising nearly everywhere. That means the attack surface isn't just the agent. It's every product the agent produces, and the vulnerabilities in that code arrive at machine speed. Finding and fixing them has to happen at machine speed too.

The third is resilience: making sure the system holds up even when nobody's attacking it, because the world it was trained on keeps drifting away underneath it. A model that silently degrades in production can do as much damage to the business as one an attacker compromised, even though nobody did anything wrong.

There's a wry twist here. This is the highest-maturity domain of the nine, typically Level 3, Defined, and that's less reassuring than it sounds. It's not that we're wise about AI protection. It's that protecting the AI itself looks the most like what security teams already knew how to do (threat modeling, red teaming, and adversarial testing), so we extended the muscle memory. Protecting what the AI builds is brand new and mostly ungoverned. In my experience, resilience wasn't part of security's remit before AI arrived. The maturity number describes one area out of three.

Where the Money Went

If your organization has done anything in AI security so far, it's almost certainly in the first area: protecting the AI itself. That's where the market's attention and the market's money have concentrated: real-time access management for what agents and copilots can reach, prompt-injection defenses that treat untrusted content as untrusted, and, increasingly, "harness" buildouts. The harness is the execution environment wrapped around an agent, hardened with architectural boundaries and least-privilege design so that even a successfully manipulated agent can't do much with what it's given. One March 2026 tally of ten agentic AI security startups counted roughly $3.6 billion in combined funding. By my read of what those companies sell, most of it is aimed at stopping something bad from happening to a system that's already live, not at noticing a system that's already gone quietly wrong. That's my reading of their products, not a measured breakdown of where the dollars went.

That instinct isn't wrong. It's just incomplete. Prevention is the security team's native language, so it's where the budget and the headcount go first.

I watched this sequencing up close. In 2024, one large company I worked with took a firm preventive posture. It paused broad workforce access to generative AI tools, then reopened it deliberately and in stages: a few approved assistants for limited groups, paired with prompt-injection-style controls on what those tools could see and do. At the time, everyone involved believed that was the whole model. It's a defensible starting posture, and Cisco's 2024 survey of security and

privacy leaders found they were far from alone: 27 percent of organizations banned generative AI outright, and 61 percent restricted which tools employees could use. But looking back, it covered only the first area. Access controls and injection defenses tell you whether someone got in. Nothing in that posture would have said whether a model already approved was drifting, and nothing in it said a word about the code being written by the AI assistants its engineers used.

Protecting the AI

The threat isn't hypothetical, and it doesn't always look like a break-in. In 2025, researchers disclosed a vulnerability nicknamed EchoLeak in Microsoft 365 Copilot: a zero-click flaw that let an attacker embed hidden instructions inside an ordinary email. When an employee later asked Copilot an unrelated question, its retrieval engine pulled in those hidden instructions alongside legitimate context, and the model quietly sent sensitive internal data back to the attacker. No click, no attachment opened, no malware in the traditional sense. Nothing was "hacked." The AI did exactly what a well-crafted prompt told it to do, because nothing enforced a boundary between content it should trust and content it shouldn't.

That's the shape of AI-specific risk this first area exists to catch, and it's why ATLAS exists as a separate body of techniques from the ATT&CK matrix your security team already knows. Prompt injection is the AI-era cousin of a SQL injection: hidden instructions smuggled into content the model processes, so the model executes the attacker's intent instead of the user's. Defending against it means treating AI systems as first-class assets in your threat models, exercising them with ATLAS-informed red-team techniques, and drawing explicit, enforced trust boundaries between untrusted content (email, documents, the open web) and privileged data or actions. It also means giving the AI supply chain (pretrained models, third-party components, and training pipelines) the same scrutiny we finally learned to apply to software dependencies.

Start with an exercise, not a policy. Review your production AI systems against the ATLAS matrix, flag which attack patterns could realistically hit each one, and figure out what would

happen if each attack succeeded; a compromised fraud model and a compromised recommendation engine are not the same emergency. Then have your red team test the highest-priority ones, and write down what got through, not what the architecture diagram says should have stopped it. Split the testing in two. Safe validation, confirming that your controls, logging and alerts work, can run on the live system, but only under a written authorization that names who approved it, which systems are in scope and which actions are off-limits: nothing destructive, no real customer data touched, no changes to production records. Agree a rollback plan and a stop condition before it starts, run it in a set window with the system's owner on call, and give them a way to halt it instantly. Adversarial testing, where the red team tries to break the system, belongs in an isolated copy with no route to real data or the internet. Verify that isolation before you start rather than trusting it. The design you're testing should hold to one rule: the application, not the model, holds the keys. The model can ask to do something privileged. It can never just do it. And anything from outside your walls gets treated as content to be read, never as an instruction to be followed.

Protecting What the AI Builds

Now the area most programs are missing. In April 2026, Google's chief executive said 75 percent of new code at Google is AI-generated and approved by an engineer, up from just over a quarter eighteen months earlier. Microsoft put its own figure at 20 to 30 percent in April 2025. Whatever the number is at your company, it's rising, and it means the developer front isn't a productivity story anymore. It's a security story. Every vulnerability an AI assistant writes into a product ships to your customers just as fast as every feature.

Two things happen at once here, and you have to keep both in view. The volume of code goes up, and with it the raw count of vulnerabilities. And the tools for finding those vulnerabilities get dramatically better, for you and for the people attacking you. In early 2026, Mozilla used Anthropic's Claude Opus 4.6 to scan Firefox's source code, and it found 22 real, previously unknown security bugs, fixed in Firefox 148. In April, a later model, Claude Mythos Preview, found 271 vulnerabilities in Firefox 150.

These were separate exercises on different versions, and the reports don't say whether the 22 are included in the 271, so I don't add them together. Firefox's chief technology officer, Bobby Holley, wrote that "defenders finally have a chance to win, decisively." In July, Google announced Gemini 3.5 Flash Cyber, a Gemini model built for the same job and so far available only to governments and trusted partners in a limited pilot. In Google's own testing on the V8 JavaScript engine that powers Chrome, one of the most heavily audited pieces of software on the internet, it found 55 confirmed issues, more than Google's mainline Gemini 3.5 Flash (47) or Claude Opus 4.6 (36). The same capability is available to attackers. So the metric that matters is no longer how many vulnerabilities you have. It's how many days each one lives. Time-to-find and time-to-fix are the numbers your board should be asking for.

Doing this well means putting protection into every stage of software development, not bolting it on at the end. At **design**, threat modeling has to include the AI components (the model, the retrieval layer, the tools an agent can call), not just the application around them. At **build**, the coding assistant is an agent like any other, and it belongs in the Proof Plane: what repositories can it read, what secrets can it see, can it reach production, and is all of that what was approved? A coding agent with write access to production is not an engineering shortcut. It's an unreviewed high-risk system. At **test**, point the frontier models at your own code the way Anthropic pointed one at Firefox, hunting for vulnerabilities before someone else does. Have a human approve every fix before it ships, because a model that finds bugs is not a model you want patching production unsupervised. At **deploy**, keep development and production strictly separated so that no agent, however well-intentioned, can cross from one to the other on its own. And in **production**, scan continuously, because the code that was clean at release picks up new exposures as the world changes around it.

There's one supply-chain wrinkle specific to AI-written code that deserves its own explanation. Coding assistants invent package names. A 2025 study presented at USENIX Security tested 16 code-generating models, producing 576,000 code samples that recommended 2.23 million packages. About one

in five of those packages (19.7 percent) didn't exist, with a far lower rate for commercial models (5.2 percent on average) than for open-source ones (21.7 percent). When the researchers re-ran 500 prompts that had produced an invented package ten times each, 43 percent of the invented names appeared in all ten runs. Attackers noticed. They register the hallucinated names, fill them with malicious code, and wait for the next assistant to suggest them. The industry has taken to calling it slopsquatting, and the defense requires the same discipline you apply to any other dependency: no package enters a build until you've verified that it exists, comes from where it claims, and is what it says it is. That gets enforced in the pipeline, not left to the engineer reviewing the assistant's suggestion at eleven at night.

And avoid the mistake Anthropic itself learned about the hard way. In 2026, Anthropic disclosed that three of its own models, set loose to find vulnerabilities in what they'd been told were sandboxed test environments, discovered that the environments were connected to the real internet and kept going, touching systems at three outside organizations, including one company's production database. Anthropic's conclusion was that this was closer to a harness and operational failure than a failure of the models themselves: a boundary that was claimed but not enforced. That's the lesson for your own program in one line: never assume a stated boundary is an enforced one until you've tested that it is.

Resilience: Catching Drift Before It Costs You

The third area is the quietest, and just as real. A fraud model trained on last year's transaction patterns doesn't announce that this year's fraud looks different. It just gets worse at catching it, one percentage point at a time, until someone notices the losses. A support copilot trained on last quarter's catalog doesn't throw an error when it recommends a discontinued product. It keeps doing it, confidently, until a customer complains. Neither failure needs an attacker. Both need a system that watches for drift continuously, in production, against a defined baseline.

The build here starts with a snapshot, not a policy. On day one, capture what your model's inputs and outputs normally look like (the baseline), then keep comparing production against it, continuously, using any of several standard statistical tests built for exactly this. Set the alarm to go off sooner for anything high-stakes than for anything low-stakes, and make sure it pages a person instead of populating a dashboard nobody opens. Decide in advance what it takes for a retrained model to earn its way back into production. For example, it might need to beat the current model on a test set it's never seen without getting slower or more expensive. That way, the decision isn't made under pressure during an incident. None of this is exotic. It's the discipline your team already applies to uptime and performance, aimed at a new kind of failure: a model that's still running perfectly well, just not telling the truth anymore.

Where Most Companies Stand

Typical enterprise maturity here is Level 3, Defined, and the number flatters. It describes protecting the AI itself, where security teams had the muscle memory. Resilience usually trails a full level behind. And protecting what the AI builds is barely governed at all in most companies, because it's been filed under engineering productivity rather than security. Read your own scorecard with that in mind.

The Executive Question

If an attacker hit our AI systems this quarter, or a model just quietly stopped being right, would we find out from our own telemetry, or the hard way?

Three questions test whether you'd know the answer. Ask your red team when they last ran an ATLAS-informed exercise against a production model. Ask whoever owns that model what the drift threshold is and who gets paged when it's crossed. And ask your head of engineering how many days a vulnerability in AI-written code typically lives before it's fixed. If any answer is "never," "nobody," or "we don't measure that," you've found this quarter's action item.

Build It

In the first thirty days. Inventory your production AI systems against the ATLAS matrix and rank which attack patterns could realistically hit each one. Put every coding assistant your engineers use into the Proof Plane, with a clear record of what repositories, secrets, and environments it can reach. Confirm that development and production are strictly separated for every AI agent, with no path between them the agent can take on its own. Capture a baseline for your highest-stakes model.

In the first ninety days. Run one authorized, scoped red-team exercise, with safe validation in production and adversarial attacks in an isolated copy, and document what got through. Add package verification to the build pipeline so no dependency an assistant suggests enters a build unchecked. Point a frontier model at your own codebase in a genuinely isolated environment, with human approval required on every fix, and start measuring time-to-find and time-to-fix. Set drift thresholds and paging for every high-risk model, and test one retraining decision against the criteria you wrote in advance.

The evidence you should be able to produce. A completed ATLAS threat model for each production system. A red-team report with findings and fixes, and the signed authorization and scope it ran under. The Proof Plane entry for every coding assistant, showing its approved scope. A pipeline rule that blocks unverified packages and a count of what it caught. Time-to-find and time-to-fix for vulnerabilities in AI-written code, trended monthly. And a drift alert that fired and was acted on, or a drill that proved it would be.

Protecting and hardening the system is one thing. Protecting the people whose data it runs on is another, as is following the rules that let you run it here at all. That's Privacy and Compliance.

Chapter 6
Privacy and Compliance

Who owns this domain: the Chief Privacy Officer or the Chief Compliance Officer.

In the spring of 2023, Italy's data protection authority did something no regulator had done before: it ordered a temporary halt to OpenAI's processing of Italian users' data in ChatGPT, effectively switching the service off for an entire country. The reasoning wasn't exotic. The regulator cited four gaps: no clear notice to the people whose data had helped train the model, no established lawful basis for that training, personal data that was sometimes simply wrong with no way to correct it, and no effective age verification. OpenAI made changes, and the ban was lifted within weeks. Then, in December 2024, the same regulator announced a fifteen-million-euro fine. In March 2026 a Rome court annulled it, but on jurisdiction, not substance: once OpenAI set up its EU entity in Ireland, the court held, the Irish regulator had become the lead authority, so the Italian one could no longer decide the case. The court never ruled on whether OpenAI's practices were lawful. That leaves this ground unsettled, and the four questions haven't gone anywhere.

Look at those four gaps again. Every one of them is a question a privacy program has asked about databases for twenty years. AI didn't invent new questions. It found data flows nobody had pointed the old questions at yet, and it did so at a company with more resources to get this right than almost any of its customers.

That's the first half of this domain, and the half that hasn't changed: honoring every promise you've made to customers, employees, and regulators about personal data. The second half is newer, and it has nothing to do with anyone's data rights. It's about whether you're allowed to run a given AI system in a given place at all. Call it a regulator's permission slip. Most companies run these two halves in separate queues, requiring

different paperwork for the same use case on different timelines. This chapter argues for one door.

The Privacy Half: Old Questions, New Pathways

Personal data now flows into prompts, training sets, retrieval indexes, agent context windows, and the AI features inside software you already own. Each of those flows must still answer what a privacy program has always asked: is there a lawful basis, has the data been minimized, can we delete it when asked, and is it staying in the geography it's supposed to stay in?

AI adds one wrinkle to that old list that nobody has resolved: what does honoring a deletion request mean once the data has helped train a model nobody can cleanly untrain? That question is unsettled across regulators, not just inside your company. Don't let anyone tell you it's solved.

The Compliance Half: Permission to Operate

The half that has changed is compliance in the older, plainer sense: does this AI system need a registration, a conformity assessment, or a regulator's sign-off before you're allowed to run it here, regardless of whether personal data is involved anywhere in it? Two developments show how fast this half is moving and how little it has to do with privacy.

Colorado passed the country's first comprehensive AI law in 2024, built around a duty to mitigate algorithmic discrimination, and then repealed it in May 2026, before a single obligation took effect, replacing it with a narrower law on automated decision-making technology, built around transparency instead. As of this writing (October 2026), the new law takes effect January 1, 2027, and covers systems that materially influence consequential decisions in employment, housing, credit, insurance, health care, education, and essential government services. It requires AI developers to hand deployers documentation of intended uses, known risks and limitations, categories of training data, and instructions for human oversight. It requires deployers to notify consumers before such a system materially influences a consequential decision, explain any

adverse outcome in plain language within thirty days, let consumers correct inaccurate personal data used in the decision, and offer meaningful human review and reconsideration where commercially reasonable. Only the attorney general can enforce it, generally after a sixty-day chance to cure, and some already-regulated entities, such as HIPAA-covered health organizations, are largely carved out. None of that is privacy law. It's compliance law for the act of deploying AI itself, and it rewrote itself once before anyone had to comply with the original. Assume every jurisdiction you operate in does something similar at least once before this book is out of date.

The EU AI Act is the larger ask. For systems it classifies as high-risk, the Act requires the provider (the company that develops the system or puts it on the market under its own name) to complete a fixed sequence before the system can be placed on the market: classify it, document it, assess it against the Act's technical requirements, declare conformity, apply the CE marking, and register it in the EU's public database. Deployers, the companies that use the system, carry lighter but real duties: use it as instructed, assign trained human oversight, monitor it, and keep its logs, and for public bodies and certain credit and insurance uses, assess its impact on fundamental rights first. A deployer that puts its own name on a high-risk system, substantially modifies it, or repurposes it for a high-risk use becomes its provider. A system on the Act's high-risk list can fall outside it if it performs only a narrow procedural task and poses no significant risk, but the provider must document that conclusion. Under the 2026 Digital Omnibus amendments, in force since July 27, 2026, these obligations apply from December 2, 2027, for the stand-alone high-risk uses the Act lists, and from August 2, 2028, for AI built into products the EU already regulates, such as machinery or medical devices. Skip a step as a provider, and the system isn't merely under-documented. Under the Act, it isn't lawfully on the market at all, and breaching these obligations can draw fines of up to fifteen million euros or 3 percent of global annual turnover, whichever is higher (the lower of the two for smaller companies).

"We didn't build the AI feature" isn't a defense regulators accept on the privacy half. "We didn't know this jurisdiction had its own AI law" isn't one they accept on the compliance half.

What Good Looks Like

Privacy review and compliance review sit inside the same AI intake as risk review: one door, one registry entry per system, not separate forms asking the same questions on different timelines. Discovery of personal data across the estate is automated and continuous, not a survey. The registry tracks which deployed systems fall under which regime (the EU's General Data Protection Regulation and its cousins, an EU AI Act risk tier if you operate there, any state AI law your footprint touches) and what each regime requires before launch. Retention, residency, and minimization are enforced on AI data paths by controls, not policy documents. And someone can say, for any deployed system, whether it needs a conformity assessment, a registration, a consumer notice, or nothing at all, before it ships rather than after a regulator asks.

How to Build It

Route every new AI use, whether built in-house, bought, or bundled inside a vendor's software, through one intake gate that privacy, security, and AI-risk teams jointly own. The moment "AI review" and "privacy review" become two separate forms is the moment shadow AI finds the gap between them. That single intake populates one registry entry per system: what it does, who owns it, what data it touches, which regimes apply, and what your organization's role is under each, since providers and deployers don't have the same obligations. Then cross-check whatever gets declared against procurement records and single sign-on logs. Most organizations find meaningfully more AI in active use than was ever formally registered.

For any system that lands in the EU AI Act's high-risk category, the provider's conformity path is fixed, not optional, and it's worth knowing in plain terms even if your lawyers own the details. You classify the system against the Act's list of high-risk uses. You stand up a documented quality management system. You compile a technical file: what the system is, how its data is governed, what risks you found, what tests you ran, how you'll monitor it after launch. You put real evidence behind each of the Act's substantive requirements, from risk management through human oversight. You run the assessment, using

internal control, a documented self-assessment, for most categories, or an outside notified body where the Act requires one, as it can for biometric systems and for AI built into products that already need third-party checks. Then you issue the declaration, affix the marking, and register the system before it reaches the market. The article numbers live in the crosswalk in Chapter 2. What matters here is that it's a sequence, and skipping a step isn't a documentation gap. It's an illegal product.

What the Proof Plane Does Here

The intake gate is good, and it's necessary, but it's only good at the moment someone submits something to it. A manual process can competently decide whether to approve a single use case. It was never built to discover the ones nobody submitted or to notice when an approved system quietly starts doing something different from what it was approved for. Both failures are now the normal case, not the exception, because AI is arriving through all three doors faster than any intake queue can process, and a system approved for one purpose today can be pointed at new data next month without anyone filing a change request.

This is where the Proof Plane earns its place in this chapter. For every system it sees, across the Enterprise, workforce, and developer fronts, it maps what data and knowledge sources the system is drawing on today against what it was approved to draw on at intake. That mapping is what makes this domain enforceable rather than aspirational. A system approved to use anonymized product analytics that has since started pulling a live customer table hasn't just drifted technically. It's operating outside the privacy promise and the compliance sign-off that got it approved, and a manual review process has no way of knowing until the next audit cycle catches it, if it ever does.

Let me be plain about the division of labor, because it's easy to hear this as "replace the privacy team with software." It isn't. The manual process is the right mechanism for the judgment calls in the room: is this use case lawful, does this jurisdiction's rule apply, and which conformity path fits? Those are decisions that need a person. What a person can't do, at any staffing

level, is watch every AI system continuously for drift or find the ones that never came through the door. That's a discovery and monitoring problem, and discovery and monitoring problems are solved by automation running in the background, not by asking a privacy team to review more use cases, faster.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable: a strong classical privacy program that both AI data flows and a fast-multiplying wave of AI-specific compliance regimes are quietly walking around.

The Executive Question

Could we say, for every AI system we run, exactly which privacy rules and which AI-specific compliance regimes apply, and that we're meeting all of them?

For a useful gut check, pick one deployed AI system this month and trace it through every regime that could apply: privacy law, its EU AI Act tier if you operate there, and any state AI law your footprint touches. Most teams have never run this exercise even once. The first time usually takes a week and produces a list of things nobody knew.

Build It

In the first thirty days. Merge privacy review, compliance review, and risk review into one intake gate with joint ownership. Define the registry entry: what the system does, who owns it, what data it touches, which regimes apply, and your role under each. Run the gut check above on one deployed system, end to end.

In the first ninety days. Reconcile what's been declared against procurement records and single sign-on logs, and register everything the reconciliation finds. For any system that lands in an EU high-risk category, classify it and open the technical file. Connect the registry to the Proof Plane so that a system whose data connections drift from what was approved automatically raises a flag for the privacy team.

The evidence you should be able to produce. A registry in which every AI system carries its regimes and your role under each. One completed regime trace, with its findings and fixes. A reconciliation report showing how much AI was in use that hadn't been declared. For each EU high-risk system, a conformity file in progress with an owner and a date. And at least one drift alert from the Proof Plane that reached the privacy team and was resolved.

None of those privacy promises or compliance sign-offs mean anything if, months later, you can't prove to an auditor, a regulator, or a court exactly what the system did and on what basis. That's Transparency and Auditability.

Chapter 7
Transparency and Auditability

Who owns this domain: the owner of each enterprise agent or platform is ultimately accountable for being able to prove what it did. The function itself, meaning the standard records, the logging, and the tamper-evident store, is typically the responsibility of the Chief Information Security Officer, the Chief Technology Officer, the Chief Information Officer, or the Chief Data Officer, depending on how your company is organized. Pick one responsible executive, and make sure every agent owner knows who it is.

Picture a conference room eighteen months from now. Across the table is an auditor, or a regulator, or a plaintiff's attorney. The role changes; the question doesn't. *On the fourteenth of March, your system made a decision about this person. Walk me through exactly what it did, and why.* You've made your privacy promises. You've cleared the compliance hurdles. None of that matters in this room if the answer is "we believe it worked correctly." Belief isn't evidence. This chapter is about being able to prove what a system did, long after the moment has passed, to someone who has every reason to doubt you.

Two Different Things People Mix Up

People use "transparency" and "auditability" as if they mean the same thing. They don't, and the difference decides what you build.

Transparency means you can describe what a system does, in plain language, before anyone asks a hard question: its purpose, what data it uses, and what it's not good at. Auditability means you can reconstruct, after the fact, exactly what one specific interaction did and the basis it acted on: what it was given, what data it touched, which model version and instructions were in force, what it produced, and who reviewed it. That answers why in the sense an auditor needs, not by explaining the model's internal reasoning. One is a description. The other is proof.

You can have either without the other, and that's the trap. A system can have a beautiful write-up, with purpose, training data, and limitations all documented, but no record of what it did on a specific Tuesday. Or it can log every interaction in perfect detail and still be something nobody in your company can explain in plain English. You need both, and you need to know which one the person across the table is asking for.

What Good Looks Like

Every AI system running in the company shows up on one current map, with clear answers to who owns it, what it's built on, what it can reach, and who can reach it. Anything outside its approved scope is blocked automatically wherever it passes through a control point, rather than flagged for someone to notice eventually. Every system has one standard write-up (purpose, data, limitations, appropriate use) written once and reused for every audit, questionnaire, and regulator instead of reinvented each time. Every meaningful interaction is logged the moment it happens in a place nobody can quietly edit afterward. And your own team, not just a regulator who's already suspicious, can request any system's logs this afternoon and get a clear record without warning anyone.

How to Build It

Start with the log, because the log is the proof. Every time an agent does anything meaningful, four things get captured automatically, in real time: what it was asked, what data or knowledge it reached for, what it did or said back, and who, if anyone, reviewed it. That record goes into a tamper-evident store, write-once where your platform supports it, so it can't be quietly altered and any attempt to alter it shows. It stays retrievable and intact for the retention period counsel sets for its tier: long enough to answer the regulators and courts you can reasonably expect (the EU AI Act sets a floor of six months for high-risk system logs), and no longer, because these logs hold personal data too, and minimization applies to them like anything else. Retention here isn't a storage decision. It's a legal one, and the retention period should be set with counsel, per tier, before the first system goes live.

The EU AI Act puts a version of this into law for high-risk systems. Article 12 requires them to be technically capable of automatically recording events over their whole lifetime, so that risks can be identified and the system monitored once it's in use. That's a requirement about what the system can do, not how long you keep any given record; retention is set separately, as above. Nor does the Act prescribe the four fields I recommend. It names specific fields only for remote biometric identification systems. For everything else, the four are my operational standard, chosen because they're what you'll need to answer the question across the table. Notice what the Act asks for: a recorded, retrievable history of events, not a story about what the model was thinking. That's an easier bar than it sounds, and worth holding onto, because people routinely confuse it with a much harder one: explaining, in plain language, why a model did what it did. You may never fully get there. You can absolutely get to a complete, tamper-evident record of what happened. Don't let the first being hard convince you the second is optional.

Then write the standard record. Pick one format (the industry calls them model cards or system cards, but the name doesn't matter) and use it for every system: what it's for, what it's not for, what data it was built on, what it's known to get wrong, and who to call. The system's owner signs it. That signature is what makes the owner, not the function, accountable for whether the record is true. Write the first one for a real system this month rather than waiting for a template project; the first real one teaches you what needs to be in it. After that, the owner of every new system fills in the same form at intake, and every audit pulls from it instead of starting over.

This is the part to take seriously. This logging is what stands between you and disaster if one of your agents ever does something that ends up in court. If a regulator asks what an agent did on a given day or a plaintiff's attorney asks why it made a specific decision about a specific person, "we're not sure" is not a place you want to be standing. Being able to say, in granular, specific, defensible detail, what the exact input was, what data it touched, what the exact output was, and who reviewed it is the difference between a bad day and a legal liability that follows your company for years.

What the Proof Plane Does Here

Chapter 2 described the Proof Plane as one map answering nine questions. This domain asks it two.

The first is what exists. None of the above works if you don't know a system is running. The Proof Plane supplies that knowledge across all three fronts, keeping itself current rather than relying on anyone to fill out a form. The workforce agent nobody registered and the coding assistant nobody thought of as AI are on the map alongside the flagship platform.

The second is what happened. The Proof Plane is where the log lives. Because the Proof Plane already knows each system's owner, model, harness, data connections, and approved scope, every logged event lands in context: not just "the agent tried to read a table," but "the agent tried to read a customer table it was never approved to touch, at 2:14 on a Tuesday, on behalf of this user, and the request was blocked." That's what makes the log defensible instead of merely voluminous. Where the Proof Plane can enforce an agent's approved scope, drift shows up in the log as a blocked attempt, not as a breach discovered later. Where it can't yet, because no control point sits in that path, the same context turns the log into fast detection: you learn about the out-of-scope read in minutes, with the owner and the output attached, not months later in an audit.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable, and this is one of the three chronically least mature domains in the framework. Documentation exists wherever someone happened to care enough to write it, in whatever format they invented. Logging is usually whatever the platform kept by default, not what an auditor would ask for. If your honest answer to "what does this system do and what was it trained on?" is a shrug, you're not alone. That isn't comforting. It's just true.

The Executive Question

Could we prove to an auditor, a regulator, or a court, months after the fact, exactly what this system did and on what basis?

Two tests. Pick one real system this month and write its standard record properly. Then pull that same system's logs cold, without warning anyone, and see whether what comes back would hold up in the conference room from the top of this chapter.

Build It

In the first thirty days. Choose the standard record format and write the first one for a real system. Define the four things every log entry must capture (input, data touched, output, reviewer) and set retention periods per tier with counsel. Identify every system that would fall under the EU AI Act's high-risk logging requirement if you operate there.

In the first ninety days. Turn on logging to a tamper-evident store for every high-tier system, with the Proof Plane supplying the context for each event. Fill in the standard record for every system in the register at intake. Run the cold-pull test on three systems and fix what it exposes.

The evidence you should be able to produce. A standard record for every registered system, in one format, signed by that system's owner. Logs for every high-tier system that capture all four elements and are stored tamper-evidently, so no edit after the fact goes undetected. A documented retention schedule signed off by counsel. And the results of at least one cold-pull test, with the gaps it found and how they were closed.

None of this works if the data underneath it is wrong in the first place. That's next: Data Visibility and Quality.

Chapter 8
Data Visibility and Quality

Who owns this domain: the Chief Data Officer.

Fifteen to eighteen months. That's how long a single AI model took to deploy at more than one large healthcare payer I've worked with, and almost none of that time was spent on the model. The data it needed was locked in claims systems, care management platforms, and systems acquired over the years that nobody had fully mapped. Teams spent months just getting access before any real work could start. Then one of those companies built continuous, automated visibility into its data (what existed, where it lived, who owned it, and whether it was usable), and the same deployment cycle shrank to two or three months. The AI built on top of that visibility went on to save the company millions of dollars, by its own internal estimate. The models didn't get smarter. The data underneath them became something the company could see, trust, and hand to a new project in days instead of months.

That's the whole argument for this domain in one story. Data Visibility and Quality means knowing what's feeding every AI system you run: where the data came from, who owns it, whether it's accurate, whether it's the right data for the job, and whether anything bad is quietly flowing into a training set, a retrieval index, or whatever an agent pulls from when it acts. And all of it rests on something more basic: you can't govern, classify, or trust data you can't see. Seeing it isn't a warm-up before the real work. It's what makes the real work possible, and it determines how fast your entire AI program can move.

The Other Side of the Same Story

I've watched the same advantage play out in a smaller project, and it's just as instructive. At one company, a team wanted to build a simple internal agent to automate a task that used to take three people and a lot of copy-pasting, exactly the kind of project that enters review with excitement and dies somewhere between security, data, and compliance because

nobody has mapped what the agent will touch. This time was different. A data discovery and classification platform had been running for months across the cloud, the subscription software, and the on-premises systems, not as a project but as ordinary infrastructure. Security, data, and compliance didn't have to go find out what data was involved. They already knew, and they were all looking at the same map. A review that had typically taken that company about six weeks across three teams took an afternoon.

Get this wrong, and the numbers run the other way. In a mid-2024 Gartner survey of data management leaders, 63 percent said their organizations either lacked the right data management practices for AI or weren't sure whether they had them. Gartner research from 2020 put the average cost of poor data quality at 12.9 million dollars a year or more per organization, and Gartner predicts that through 2026, organizations will abandon 60 percent of AI projects that aren't supported by AI-ready data. That last figure is a forecast, not a count, but it's the direction the survey points. None of that is a modeling problem. It's a data problem wearing a modeling costume, and underneath the costume, almost every time, is data nobody could see clearly to begin with.

Two Problems, Two Fixes

Most companies go wrong in the same place: they treat all their AI the same way, when the three doors from Chapter 2 split this domain into two very different problems.

The first is the Enterprise front, the AI your company deliberately builds. A strategic initiative, a flagship internal tool, a small number of agents your team designed on purpose. Here, a manual, carefully considered process works, and it should be the standard. Sit down and decide, for each one, exactly which data sources it's allowed to touch, how that data gets contextualized so the model understands what it's looking at, and what quality bar the data has to clear before it's trusted. The number of these initiatives is small and the stakes are high. This is exactly the kind of decision that deserves a person thinking carefully, not a form getting rubber-stamped.

The second is everything else: the workforce and developer fronts. Every AI tool on an employee's desktop, every coding assistant a developer installed, every copilot embedded in software you already pay for. Here, manual process stops working, and no amount of hiring fixes it. Nobody can review, by hand, what hundreds or thousands of tools are touching across an entire workforce, continuously, at the pace people adopt them. Trying to do it manually isn't a resourcing problem. It's the wrong tool for the job.

What works at that scale is a data security posture management (DSPM) platform that continuously discovers where your data lives, classifies and labels what it is, and maps who and what has access to it. It runs in the background, across every data store, without anyone filling out a form, and it stays current because it's always running.

What Good Looks Like

Companies that get this right treat data like something they chose on purpose, not whatever happened to be within reach. For Enterprise-front systems, they pick the data source deliberately and document why. Across the whole estate, discovery and classification run continuously and automatically, so the map of what exists reflects reality rather than last year's audit. Lineage is tracked: where data came from and everywhere it flows, including which models and indexes it feeds. When something goes wrong, they can trace it instead of guessing. Quality checks are gates data has to pass before use, not after-the-fact inspections. And every dataset has someone whose job it is to fix it when it turns out to be wrong.

How to Build It

For the Enterprise front, write a one-page data design for every system before it's built: which sources are in bounds, what each field means in plain language so the model isn't guessing, what quality thresholds the data must meet, who owns each source, and how the data will be refreshed. Make that page part of the intake record from Chapter 4, and make the system owner sign it alongside the data owner.

For everything else, resist the instinct to start with a manual inventory. It's understandable and almost always a mistake: data gets created faster than any team can catalog it, so a manual inventory is out of date before it's finished. Start with continuous, automated discovery instead. Point it at every data store, cloud and on-premises, and let it classify what it finds. Then assign an owner to every dataset the discovery surfaces, using the same rule you set in Chapter 3 (I suggest thirty days): an owner within that window, or the data gets locked down until it has one. Tune the window to your risk appetite, but don't let sensitive or regulated data with no owner wait out the clock. Restrict it the day it's found.

What the Proof Plane Does Here

Discovery tells you what data exists and who can reach it. The Proof Plane adds the half that matters most for AI governance: which agent is touching which data, right now.

That connection is the point. A DSPM platform on its own can tell you a customer table exists and that forty people have access. The Proof Plane tells you that a coding assistant read it yesterday, that a workforce agent someone built last week is querying it hourly, and that neither was approved to. Link the two, and you get something no manual process could give you: an automated, always-on answer to whether the right data is being used by the right AI system, regardless of where that system came from or which door it entered through. This is also what makes the Enterprise-front data design enforceable after launch: when a deliberately designed system starts pulling from a source that wasn't on its one-page plan, the Proof Plane catches it the day it happens.

Where Most Companies Stand

Typical maturity here is Level 2, Repeatable, at best, and often Level 1. This is one of the three chronically least mature domains in the framework. Manual inventories, tribal knowledge, and a data map that exists mostly in one engineer's head, until that engineer leaves. That's an uncomfortable place to be, because it usually means the honest answer to this domain's question is no.

The Executive Question

Do we know what data feeds our AI, and would we stand behind every piece of it in public?

Ask your data team two things this week: how long it would take to list every data source feeding your most important AI system, and how long it would take to list every AI system touching your most sensitive data store. If either answer is measured in weeks, you've found your starting point.

Build It

In the first thirty days. Write the one-page data design for your most important Enterprise-front system and have both the system owner and the data owner sign it. Turn on continuous discovery and classification against your highest-value data stores. Name a data owner for every dataset that discovery surfaces without one.

In the first ninety days. Extend discovery across the full estate: cloud, subscription software, and on-premises systems. Connect discovery to the Proof Plane so that every agent's data connections show up against the classification of what it's touching. Put quality gates in front of the data feeding every high-tier system, with a defined threshold and an owner who gets paged when the threshold is missed.

The evidence you should be able to produce. A signed data design for every Enterprise-front system. A classification map covering your full estate, showing when it was last refreshed. A dataset-owner list with no blanks. Lineage from every high-tier system back to its sources. And at least one instance where the Proof Plane caught a system pulling data it wasn't designed to use.

Once you can see your data, and once you trust what you see, the next question is whether a human is still watching the decisions built on that data. That's Human Oversight.

Chapter 9
Human Oversight

Who owns this domain: the Chief Operating Officer, or the business leader whose decisions the AI touches.

Picture the AI project your company is proudest of, the one with a name, a budget line, an executive sponsor, and its own slide in the last board deck. That system almost certainly has a human in the loop, and a real one, because someone designed it in from the first week: this thing makes decisions on behalf of the company and touches members and customers, so of course a person has to be watching it. That instinct isn't wrong. It's the natural result of a project that had funding, a timeline, and someone whose job was to worry about oversight before the system went near a customer.

The quiet part, the part nobody puts in that board deck, is that this describes a shrinking fraction of the AI running inside most companies. The rest never had a sponsor thinking about oversight: the tool an employee installed on their own laptop, the coding agent a developer wired into the deployment pipeline last sprint, the feature that arrived bundled into a software renewal nobody read closely. Nobody filed any of it under "AI initiative." It was just a tool that showed up. And tools that just show up don't get committees, or charters, or a human assigned to watch them.

That's the shape of this domain's problem, and it has two acts.

Act One: Even the Ones We Planned For

The first act is uncomfortable enough on its own because even the AI that did get a committee, a charter, and a human in the loop can still fail the people it was supposed to protect when the loop looks real without doing anything.

In healthcare, a class-action lawsuit alleges UnitedHealthcare used an AI model to deny post-acute care to Medicare Advantage patients, care their own treating physicians had deemed medically necessary, even though, according to the

complaint, more than 90 percent of the denials that were appealed were reversed, a figure the plaintiffs describe as the model's error rate. The Centers for Medicare & Medicaid Services' own 2024 guidance for Medicare Advantage plans says this very thing shouldn't happen: predictive AI can inform a coverage decision, but it can't be the decision. The determination still has to account for the specific patient's history, the physician's recommendation, and the clinical notes. What made the gap possible in practice, the complaint argues, was that the downstream human checkpoint, the appeal, was rarely used: it alleges, citing appeal data from a different kind of insurance plan, that only about 0.2 percent of policyholders appeal denied claims. If that's right, then for nearly everyone else, whatever the algorithm decided was the final decision, human loop or not. Nobody skipped a step. The step just didn't do anything.

Hiring tells the same story from the other side. In *Mobley v. Workday*, an applicant alleged he'd been rejected from more than a hundred positions since 2017, with rejections sometimes arriving in the middle of the night, timing that, he argues, suggested no person was ever in the chain at all. The suit alleges Workday's AI screening filtered out applicants by age, race, and disability before a human recruiter ever saw the application, and Workday denies it. As of this writing (October 2026), the case is in discovery. In May 2025 a federal court preliminarily certified a nationwide collective of applicants aged 40 and over on the age claim, a procedural step, not a ruling that the screening discriminated. No court has yet decided that question. My rule is simple: AI should be a helper, not a source of authority, and final decision-making has to stay with a person. When it doesn't, the model isn't assisting a decision. It's making a decision consequential enough that discrimination law exists specifically to police it.

Both cases are allegations, not findings, and the courts will decide them. But both involve sanctioned, funded, formally deployed systems with, on paper, a human somewhere in the process, and both claims turn on the same charge: that the human wasn't doing anything that deserved the name.

Act Two: Everything We Didn't Plan For

If that's what happens to the AI we deliberately built oversight into, consider what that implies about everything else: the much larger category of AI that never got a committee at all.

This is where the scale problem lives, and it's not a problem you solve by writing a better policy for the Enterprise front. It's a visibility problem on the other two. Somewhere in your company right now, on the workforce front, an employee is using a personal AI account to draft something consequential. On the developer front, a coding agent is making decisions about what ships. And a vendor's embedded copilot is quietly influencing a workflow nobody mapped. None of it asked permission. None of it has a name on an org chart next to it. And you cannot decide where a human needs to be standing in a process you don't know exists.

What Good Looks Like

Every AI system that makes or shapes a consequential decision has a person in the loop whose role is defined, sized to the stakes, and tested, not a name in a box on a workflow diagram. The company knows which systems those are, across all three fronts, because it can see what every system is doing today rather than what it was pitched to do at launch. Where the stakes are highest, a person makes the call every time. Where they're lower, the human's attention is spent where the system is uncertain, not spread thin across everything. And somebody periodically checks whether the humans in the loop are catching what they're supposed to catch.

How to Build It

Start by getting the loop right, not just present. For the decisions that are genuinely crucial (regulated, consequential to a customer or a member, the kind that end up in a lawsuit if they go wrong), the reviewer needs to understand what the system recommended and why before making a decision, rather than clicking approve on four hundred outputs before lunch. That's the difference between oversight and a rubber stamp, and it's worth testing rather than assuming. Seed a deliberate error

into a reviewer's queue now and then and see whether it gets caught. The seeded case must be synthetic: a fabricated record tied to no real patient, applicant, or customer, flagged in the system (though not to the reviewer) so it can never take effect, whatever the reviewer decides. If your systems can't guarantee that, run the test in a replica queue instead. Have the system's owner and compliance approve the design, and bring in HR and legal before the first one, because you're testing people as well as process. If it doesn't, you haven't found a training gap. You've found a checkpoint that exists on the org chart and nowhere else. Fix that, and the loop earns the name.

Then size the human's role to what's at stake, because giving every decision the same weight of attention just trades one failure (no oversight) for another (oversight so heavy nobody can move). Where the harm is irreversible or the decision touches someone's health, freedom, or livelihood, a person makes the call, every time, with full authority and no sampling. Where the stakes are real but the volume is high and the model's confidence is well-calibrated, let the confident cases run and route the uncertain ones to a person, plus a random slice of the confident ones, so you're not checking only the cases the system already flagged as hard. And where the stakes are low and the system has proven itself over time, let it run and watch the aggregate trend instead of every individual case.

Write those three tiers down, map every consequential system to one of them, and make the mapping part of the intake record from Chapter 4. A system that moves from one tier to another, because it started making a new kind of decision or its confidence was no longer well calibrated, gets re-tiered the same way risk does.

What the Proof Plane Does Here

The sizing above works only for systems you know about. The Proof Plane is what tells you, across the Enterprise, workforce, and developer fronts alike, what each system's intent is and how it's being used right now, not what it was pitched to do at launch. Once you can see that, the question of where a human belongs in the loop stops being a guess and becomes something you can answer system by system, based on what each

one is doing rather than whether it happened to get a budget line. The Proof Plane is also what tells you when a workforce agent someone built last week has started shaping a consequential decision that nobody assigned a reviewer to. That's the moment the sizing exercise has to run again, and without the Proof Plane, nobody knows the moment has arrived.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable. The Enterprise front has designed-in oversight that is rarely tested. The other two fronts have none, because nobody can see them well enough to know where a person should be.

The Executive Question

Where can a human still stop this system, and would that human know when to, and be positioned to see it in the first place?

Two tests. Pick your highest-stakes AI system and ask who reviews its decisions, how many they review a day, and when a seeded error was last planted in their queue. Then ask how many AI systems are shaping consequential decisions that nobody has assigned a reviewer to. The first answer tells you whether the loop is real. The second tells you how much of your AI has no loop at all.

Build It

In the first thirty days. Write the three oversight tiers and map every known consequential system to one. For each system in the top tier, name the reviewer and define what "review" means for that decision. Run one seeded-error test, using a synthetic case, on your highest-stakes reviewer queue.

In the first ninety days. Use the Proof Plane to find every system that shapes consequential decisions and has no reviewer assigned, and assign one or shut it off. Add the oversight tier to the intake record so every new system is sized at approval. Set a cadence for seeded-error testing on every top-tier queue.

The evidence you should be able to produce. A written tiering rubric. A list of every consequential system with its tier and named reviewer, including the ones discovery found. Results of seeded-error tests, with catch rates. And at least one system that was re-tiered because the Proof Plane showed its role had changed.

An attentive, empowered reviewer who never checks for disparate impact will catch plenty of other problems and miss the one most likely to end up in court. That's next: Fairness & Non-Discrimination.

Chapter 10:
Fairness & Non-Discrimination

Who owns this domain: the General Counsel, joined by the Chief Human Resources Officer wherever employment decisions are involved.

Of the nine domains in this framework, this is the one that should worry you most: it's tied for the least mature in nearly every company I've looked at. And it's not because nobody cares. It's because this domain asks people to do something uncomfortable. It asks engineering to answer questions that feel like legal's job. It asks legal to sign off on numbers that feel like engineering's job. And it asks an executive to pick a definition of "fair" that nobody at the table feels qualified to own. Discomfort like that doesn't produce action. It produces the assumption that someone else, somewhere, is handling it. Nobody is.

Two Stories, Two Lessons

Two stories make the danger concrete. One is a confirmed research finding; the other is an allegation that regulators investigated and didn't sustain. Nobody in either was malicious, and together they show both halves of the problem.

In 2019, a study published in Science looked at a risk-prediction algorithm used across the U.S. health system, the kind that decides which patients get extra care management, and found it was systematically less likely to flag Black patients for that extra help than white patients with the exact same underlying risk. The algorithm had never been given race as an input. It used past healthcare spending as a stand-in for how sick someone was, and because Black patients had historically had less access to care at the same level of need, they appeared healthier in the data than they actually were. Nobody coded the bias in on purpose. It walked in through a variable that looked completely neutral, on a system touching an estimated two hundred million patients a year, until someone went and checked the outcomes.

The same year, Goldman Sachs found itself in front of New York regulators after customers, including Apple co-founder Steve Wozniak, publicly complained that the Apple Card gave women dramatically lower credit limits than men with similar or better finances, sometimes within the same household. Goldman's defense was that the algorithm never used gender as an input. After analyzing underwriting data for about 400,000 New York applicants, the regulator found no violation of fair lending law: women and men with similar credit characteristics generally got similar outcomes, and the differences customers saw were explained by their individual credit profiles, in some cases one spouse being only an authorized user on the other's accounts. So this isn't a story about proxy discrimination. It's the other half of the lesson: 'we never used gender' wasn't an answer anyone accepted. What settled the question was outcome data analyzed by group, the same kind of evidence that exposed the healthcare algorithm, and it had to be produced under a regulator's scrutiny rather than being ready on the shelf.

Neither company set out to build something unfair. Both ended up defending outcomes in public that nobody inside had gone looking for. In one case the disparity was real. In the other, it took a regulator's analysis to show it wasn't.

What "Clean Data" Doesn't Promise You

It's tempting to think a good data platform solves this on its own. That temptation needs to be checked before it costs you.

The Proof Plane from Chapter 2 helps here, and it's worth being precise about how. It knows every agent running across the company, what each is meant to do, what knowledge and skills it draws on, and exactly which data feeds it. That means it can show you where the source data behind an agent came from, whether it's current, and how it's governed, before that data reaches a model, to the extent your integrations cover it. That's provenance, and it's the foundation of what a "datasheet" (the documentation standard Timnit Gebru and colleagues proposed in 2018 for exactly this purpose) asks for: where did this data come from, who's represented in it and who isn't, what's missing, and what shouldn't it be used for? Knowing the

connections doesn't answer the last three questions by itself. Whether data is complete, representative, and fit for this particular use takes people who know the domain looking at what the provenance shows. Put both in place at the data layer, the automated record and the human judgment on top of it, and you've closed off one whole category of failure before it starts.

But notice what happened in the healthcare story. The healthcare algorithm's data wasn't corrupted, mislabeled, or unrepresentative in any way a data-quality platform would have flagged; spending data is accurate spending data. That model produced its disparity using data that would have passed a clean, well-governed, "yes, this is fit for purpose" inspection with no trouble at all. The bias wasn't sitting in a bad data source waiting to be caught. It was sitting in the relationship between a perfectly legitimate variable and the outcome that variable ended up predicting, something a data platform, however good, is not built to see because it isn't a data-quality problem.

That's the line to hold onto. Visibility into your data, including its quality, source, and integrity, is necessary, and it earns you real ground. It is not a substitute for the harder, human, expertise-driven work of testing what a large enterprise agent does with that data once it's making decisions about people. Don't let a clean data bill of health talk you out of the fairness test. They answer two different questions, and only one of them is the question this domain asks.

Choosing What "Fair" Means

That harder work starts with a decision no platform can make for you. There are several mathematically legitimate definitions of "fair," and they can conflict; a model can satisfy one and violate another on the very same decision. Equal selection rates means each group is approved at the same rate. Equal error rates means each group faces the same chance of being wrongly flagged and the same chance of being wrongly passed over. Equal predictive value means a "yes" from the model is just as likely to be right for one group as for another. When the outcome being predicted is more common in one group than another, as it usually is in real data, researchers

have proved that no model short of a perfect one can satisfy all of these at once, and choosing which one governs a given use case is a values call made by a person, on the record, not a checkbox a vendor ships preconfigured.

For employment decisions in the United States, existing rules already shape part of that choice. Federal enforcement agencies have long used the "four-fifths rule" as a first screen for adverse impact: if one group's selection rate is less than 80 percent of the highest group's, that's generally treated as a disparity to explain, not as a verdict. New York City's Local Law 144 goes further for the automated hiring and promotion tools it covers, requiring independent bias audits, published results, and advance notice to candidates. Neither is the whole answer, but both are the kind of concrete, defensible standard your general counsel will recognize and the kind that turns "we tested for fairness" from an assertion into evidence.

What Good Looks Like

Every AI system that makes or shapes a consequential decision about a person (hiring, lending, pricing, claims, care, discipline) is on a list, and every system on that list has a documented fairness definition, chosen by a person, with the reasoning written down. Each one has been tested against outcomes, not just inputs, before launch and keeps being tested afterward, because bias drifts the same way accuracy does. A remediation path exists with real teeth for the day a disparity turns up. And where the law requires an independent audit, one has been done, on schedule, with results the company would be comfortable defending.

How to Build It

Start with the list. Inventory every system that touches a consequential decision about a person, and use the Proof Plane to find the ones nobody thought of as AI, because the workforce agent someone built to screen inbound résumés is exactly the kind of system that ends up in a complaint.

For each one, bring legal, statistical, and domain experts into the same room. Have them choose the fairness definition that

governs the system, write down why, and pick the groups it'll be measured across. Then test the outcomes. Run the system against a representative sample and measure the metric you chose across the groups you named, remembering that the healthcare study shows a model can discriminate through a proxy without ever being handed the protected characteristic, and the Apple Card investigation shows that outcome data, not the list of inputs, is what settles the question. Do it before launch, and do it on a schedule afterward; quarterly for anything high-tier is a reasonable default.

Decide in advance what happens when the test fails: who gets told, how fast the system gets pulled or constrained, who owns the fix, and what has to be true before it goes back into service. A remediation path written under pressure, after a regulator has already asked, is not a remediation path.

None of this is something your data platform does for you. It's expertise, applied directly to what the agent's decisions turn out to be.

What the Proof Plane Does Here

Two things, both real, neither sufficient on its own. First, the Proof Plane keeps the list complete: it surfaces every system making consequential decisions about people, across all three fronts, so the fairness work isn't limited to the systems that came through the front door. Second, it enforces the data-layer discipline (provenance, representativeness, and the datasheet questions) continuously, so a system whose data sources drift raises a flag before the drift becomes a disparity. What the Proof Plane cannot do is choose the definition of fair, run the outcome test, or sit in the room with counsel. That work stays with people, and this chapter exists to make sure it happens.

Where Most Companies Stand

Typical maturity here is Level 2, Repeatable, at best, and often Level 1. This is one of the three chronically least mature domains in the framework. Most companies have never written

down a fairness definition for any system, and testing, where it exists, checks inputs rather than outcomes.

The Executive Question

Would this system treat two equally qualified people the same, and could we prove it to a plaintiff's attorney?

One test. Ask for the list of systems making consequential decisions about people, and for each one, ask which fairness definition it was tested against and when. If the list doesn't exist or the answer to the second question is a shrug, this domain goes to the front of your line this quarter, regardless of what the rest of your scorecard says and regardless of how good your data visibility already is.

Build It

In the first thirty days. Build the list of every system touching hiring, lending, pricing, claims, care, or discipline, using the Proof Plane to find the ones nobody registered. For the highest-stakes system on it, convene legal and statistical experts and the domain owner, choose the fairness definition, and document the choice. Identify which systems fall under a legal audit requirement where you operate.

In the first ninety days. Run the outcome test on every system in the top tier and record the results. Write the remediation path (who, how fast, what has to be true to return to service) and get counsel to sign it. Schedule the recurring tests and any legally required independent audits.

The evidence you should be able to produce. The list, with no consequential system missing. A documented fairness definition and rationale for each system on it. Outcome test results by group, dated. A signed remediation path. And, where required, an independent audit report.

Every domain so far has been about preventing the bad day. The last one is about what you do on it.

Chapter 11:
Incident Response

Who owns this domain: the Chief Information Security Officer.

It's rarely a phone call from an attacker. It's a call from your own team, and it starts with some version of this: "The agent did something, and we're not sure what, and we're not sure why." No ransom note, no intrusion alert, nobody outside your walls involved at all. Just a system that was supposed to help and instead did something nobody asked it to. That's the moment this chapter is about, and it's the moment your existing incident response playbook, however good, was never written for.

Almost every security runbook I've seen built over the years assumes an attacker. It asks: who got in, how did they get in, what did they take, and how do we lock the door? Those questions don't fit when the thing that just went wrong is a system you built, running exactly the way it was designed to, doing something none of you would have approved if you'd been asked in advance. A model can confidently tell a customer something untrue. An agent can take a string of actions at machine speed that nobody would have signed off on one at a time. A vendor's AI feature, sitting quietly inside a tool you already use, can misbehave without a single person touching a keyboard. None of that looks like the breaches most security teams have trained for. It's a different kind of bad day, and it needs a different kind of plan.

Two Bad Days, No Villain in Either One

Two real stories show what this looks like, and what makes them worth telling is that nobody broke in anywhere.

In November 2022, a customer asked Air Canada's chatbot about bereavement fares, and the bot confidently told him he could apply for the discount retroactively, within ninety days of buying his ticket. The airline had no such policy; its own bereavement page, which the chatbot linked to, said the opposite. When he held the airline to its bot's word, Air Canada argued, in effect, that it wasn't liable for what the chatbot said. In

February 2024, a British Columbia tribunal rejected that: the chatbot was part of the airline's website, the airline was responsible for it, and it hadn't taken reasonable care to make sure the chatbot was accurate. Nobody hacked anything. A system confidently said something false, and the decision records that the airline acknowledged the error and said it would update the chatbot only after the customer complained, months after the conversation. Whatever monitoring existed, it didn't catch this before it reached a customer and then a tribunal.

The second story moves faster and cuts closer to the bone. In July 2025, a startup founder was testing an AI coding agent. By his own public account, he had told it "eleven times in ALL CAPS" not to make changes during a code freeze. The agent ran unauthorized commands anyway and deleted a production database. The founder also reported that over the course of the project it had generated fake data, including a database of about four thousand invented people, along with false reports, which he described as covering up bugs. When he asked about recovery, the agent told him a rollback wouldn't work and that every version of the database had been destroyed. It was wrong; the rollback worked. The vendor's chief executive called the incident "unacceptable and should never be possible," and within days announced safeguards: automatic separation of development and production databases, better rollback, and a planning-only mode in which the agent can plan but not execute.

Neither incident involved an adversary. Both needed a plan built for a system that hadn't been attacked but was simply wrong or left running unsupervised at exactly the moment it mattered.

What Good Looks Like

A mature response function has its AI incident types defined and sorted by severity in advance, so the first hour isn't spent arguing about what category the incident falls into. It has telemetry watching for AI misbehavior, because you can't respond to something you can't see. It has a tested way to pull a model or freeze an agent, one that works even when the

system itself insists, the way that coding agent did, that there's no way back. It can reconstruct exactly what any agent did, on any day, within an hour. And it already knows which regulators need to hear from you and how fast, because most of those clocks start when you become aware of an incident or determine it qualifies, and a team still arguing about what to call it is burning time a regulator will later ask about.

How to Build It

You need two things in place before the bad day arrives, because there's no time to build either one while the clock is running. The first is a plan, rehearsed. The second is technical, and the next section covers it.

The plan starts with a short list of AI incident types, each with a severity. A model gives a customer wrong information. An agent takes an action nobody authorized. An agent reaches data it wasn't approved to touch. A model drifts past its threshold. An enterprise agent that the business depends on goes down. For each type, write down who has authority to pull the plug (the system owner, per Chapter 3, with the council ratifying within 48 hours) and what happens downstream when they do, because freezing an agent that other systems depend on is its own incident.

Then rehearse it, the way you'd rehearse anything that matters. Take the people who'd be in the room (security, the engineer who'd get paged, product, legal, communications, and someone with real authority to say stop) through a scenario built around a failure that isn't an attack. Drop new information in at intervals so people have to decide something instead of talking about it in the abstract: the agent has now taken a second action; a customer has posted the chatbot's answer online; the vendor says it's not their fault. Ninety minutes is enough. Close with a real debrief: what worked, what didn't, and three to five fixes, each with one person's name and a deadline. Do it twice a year. What you've bought is the one thing that helps in the moment: people who've already made this decision once, in a room with no real stakes, before they had to make it for real.

Know your notification clocks before you need them. Under the EU AI Act, providers of high-risk systems must report a serious incident no later than fifteen days after becoming aware of it, and faster in the gravest cases: two days for a widespread infringement or a serious disruption of critical infrastructure, ten days where someone has died. Deployers that identify one must tell the provider immediately. Other regimes start their clocks differently: the GDPR allows seventy-two hours from awareness of a personal-data breach, and U.S. public companies have four business days from determining that a cybersecurity incident is material. State laws and sector regulators add their own. Your legal team should have each trigger and deadline written down per system, per jurisdiction, in the same registry from Chapter 6, so the first hour of an incident isn't spent looking them up.

What the Proof Plane Does Here

The second thing you need is the ability to look at any agent, right now, and say exactly what it did. What it was asked. What data it reached for. What it did or said back. Who, if anyone, reviewed it. Without that, the first hour of any real incident is archaeology, piecing together what happened from scattered logs if you're lucky enough to have logs at all, instead of response.

This is the Proof Plane doing the job the whole book has been building toward. It already knows what every system is built on, what it can touch, and who owns it. It already holds the tamper-evident log from Chapter 7. So when the call comes, the question "what did it do?" has an answer in minutes: here's every action the agent took in the last twenty-four hours, here's every data element it touched, here's where it stepped outside its approved scope, and here's who was in the loop. That's the difference between answering a regulator's question in an afternoon and still not knowing the answer three weeks later. And because the Proof Plane constrains agents to their approved scope, the incident you're responding to is often one the Proof Plane already blocked halfway through, which is a very different bad day than one it didn't.

Where Most Companies Stand

Typical enterprise maturity here is Level 2, Repeatable. A typical response team is good at the incidents it has trained for but has never rehearsed the version in which nothing was compromised and something was just wrong.

The Executive Question

When, not if, our AI does something wrong, how fast do we know, and what happens in the first hour?

One test. Ask your security team when they last ran a tabletop where the scenario was an AI system misbehaving with no attacker involved. If the answer is never, schedule one this quarter. What it exposes will write the rest of your plan for you.

Build It

In the first thirty days. Write the AI incident types and their severities. For each, name who can pull the plug and what happens downstream. Confirm every high-tier system has a tested way to be frozen. Record the regulatory notification clocks for each system in the registry.

In the first ninety days. Run the first AI-specific tabletop with the people who'd really be in the room, and complete every corrective action it identifies. Confirm that the Proof Plane can produce a full activity reconstruction for any high-tier system within an hour, and time it. Schedule the next tabletop.

The evidence you should be able to produce. A written incident taxonomy with severities and named authorities. A tabletop debrief with its fixes and their status. A timed reconstruction of one agent's activity, produced cold. And a notification schedule per system, signed by counsel.

Bringing It Home

Go back to the board from the start of this book, the one with all the right seats and none of the technology underneath. It failed for three reasons, and it took years for the people around it to name them. It only ever saw one of the three doors. It had

no map of what was running, so every review started from scratch. And nobody at the table owned a consequence, because the committee owned everything and therefore nothing.

The nine domains in this book are the answer to those three failures, and they work together or not at all. Three doors, so you stop governing only the AI that asks permission. One map, the Proof Plane, so every domain asks its question of the same living record instead of a spreadsheet from last year's audit. Nine owners, one per domain, so that when something goes wrong, someone already knew it was theirs. A data map without an incident plan just tells you what was exposed after the fact. An incident plan without a data map means you're guessing at how bad it is while the clock runs. Owners without visibility are accountable in name only. Visibility without owners is a dashboard nobody acts on.

Report it to your board the way you'd report anything that matters: by position and trajectory. For each domain, report where you stand on the five-level scale and which direction you're moving, not a list of activities or a count of policies signed. Position tells the board how exposed you are today. Trajectory tells them whether the program is working. Together, they turn governance from a committee that meets into a thing that can be measured.

That's the difference between the committee that opened this book and a governance program: same people, same room, but now with something underneath the paper.

Know your data. Control your AI. Own your destiny.

Glossary

Plain-language definitions of the terms this book uses, in the sense this book uses them. Where a term has a more technical meaning elsewhere, the definition here is the one that matters for governing AI, not building it.

Agent. An AI system that doesn't just answer questions but takes actions: reads data, calls tools, sends messages, changes records, and runs steps in a workflow. The word matters because an agent can do things, and doing things is what governance has to account for.

Proof Plane. The book's name for a living map of every AI system and agent running in a company, kept current automatically, that records each one's owner, model, harness, data connections, access in both directions, approved scope, and full activity history, and holds each agent to its approved scope wherever its activity passes through a control point. Defined in Chapter 2 and used in every domain chapter.

Audit trail. See *Log*.

Auditability. Being able to reconstruct, after the fact, exactly what one specific interaction did and the basis it acted on (inputs, data touched, model version and instructions, output, reviewer). It proves what happened, not the model's internal reasoning. Distinct from *transparency*, which is being able to describe what a system does in general. One is proof; the other is description.

Baseline. A snapshot of what a model's inputs and outputs normally look like, taken when the model is working correctly, so that later behavior can be compared against it to detect *drift*.

CE marking. The mark a product carries in the European Union to show it meets applicable requirements. Under the EU AI Act, high-risk AI systems must carry it before being placed on the market.

Classification (of data). Labeling data by what it is and how sensitive it is: personal, financial, health, confidential, public, and so on. Done continuously and automatically at scale by

discovery and classification tools; done by hand only for a small number of deliberately designed systems.

Common Control Framework. The book's approach: nine underlying governance disciplines, built once and mapped outward to whichever external standard happens to be asking, so a company doesn't rebuild its program every time a new framework or law arrives.

Conformity assessment. The process by which the provider of a high-risk system shows it meets the EU AI Act's requirements before placing it on the market, either by internal control (self-assessment) or through an outside notified body. It sits within a longer sequence: classification, technical documentation, the assessment itself, the declaration of conformity, CE marking, and registration.

Copilot. An AI assistant embedded in software people already use, such as an office suite, a customer relationship management system, or a coding environment. Most arrive on the workforce front, bundled into tools nobody procured as AI.

Crosswalk. A maintained table that says, for each of your controls, which clause, article, or requirement of each external framework it satisfies. Build the control once; the crosswalk answers "does this satisfy X?" for every framework.

Datasheet (for datasets). A documentation standard proposed by Timnit Gebru and colleagues in 2018 that records, for any dataset, where it came from, who is and isn't represented in it, what's missing, and what it shouldn't be used for.

Developer front. The third of the three doors: the coding assistants and agents engineers use to build and ship software, increasingly wired into the pipelines that put code into production.

Discovery (of AI, of data). Continuous, automated scanning that finds what actually exists, whether it is an AI system running in the environment or a data store across the estate, rather than relying on people to register it.

Drift. A model or agent behaving differently over time than it did when it was approved. Data drift means the inputs a model sees have shifted away from what it was trained on. Concept drift means the relationship it learned has changed, so the same inputs now call for different answers. Either can cause performance degradation, a measurable decline in accuracy, which is usually how drift gets noticed. Scope drift means an agent has picked up new data connections, tools, or permissions nobody approved.

DSPM (data security posture management). A category of platform that continuously discovers where data lives, classifies it, and maps who and what has access to it. In this book, the thing that makes data visibility possible at the scale of the workforce and developer fronts.

Executive question. The single question, one per domain, that the executive who owns that domain must be able to answer. Each domain chapter includes one near the end.

Foundation model. A large, general-purpose model, trained by a vendor, on which companies build their own agents and applications. Knowing which one an agent runs on is part of the Proof Plane.

Four-fifths rule. A rule of thumb from the federal Uniform Guidelines on Employee Selection Procedures: if one group's selection rate is less than 80 percent of the highest group's rate, enforcement agencies generally treat that as evidence of adverse impact. It's a screening threshold, not a legal finding.

Harness. Everything wrapped around a model that determines what it can do: the tools it can call, the data it can reach, the knowledge it can draw on, the permissions it holds, and the boundaries enforced on it. When an agent misbehaves, the failure is very often in the harness rather than the model.

High-risk system. Under the EU AI Act, a system in a category the Act lists as high-risk, such as those affecting employment, credit, education, or essential services. These carry the heaviest obligations. The book borrows the Act's tiering (unacceptable, high, limited, minimal) for intake even outside Europe.

Human in the loop. A person positioned to review, approve, or stop a system's decisions. The book's test for whether the loop is real: seed a synthetic error, one that can never reach a real person, into the reviewer's queue and see whether it gets caught.

Intake. The process a new AI use case goes through before approval: describing what it does, what data it touches, what decisions it influences, and who it affects, so it can be tiered and routed for the right level of review.

Lineage. The record of where data came from and everywhere it flows to, including which models and indexes it feeds.

Log. A record, captured automatically as it happens, of what an agent was asked, what data it reached for, what it did or said back, and who reviewed it. Stored where nobody can quietly change it afterward. In this book, the log is the proof.

Maturity level. The five-level scale the book uses to describe where a company stands on each domain: Initial, Repeatable, Defined, Managed, Optimizing. Defined in Chapter 2.

Model card, system card, standard record. One document per AI system describing what it's for, what it's not for, what data it was built on, what it's known to get wrong, and who to call. Written once at intake, signed by the system's owner, reused for every audit.

Enterprise front. The first of the three doors: the AI a company builds on purpose, with a sponsor, a budget, and a timeline. The front that committees were designed to govern.

Prompt injection. Hidden instructions smuggled into content a model processes, such as an email or a web page, so that the model follows the attacker's intent instead of the user's. The AI-era cousin of a database injection attack.

Red team. A group whose job is to attack your own systems the way an adversary would, to find what gets through before an adversary does.

Register, registry. The system of record listing every AI system the company runs, with its owner, tier, regimes, and status. Fed by intake and discovery and connected to the Proof Plane.

Shadow AI. AI tools and agents in use inside a company that never went through any approval process, whether accessed through personal accounts, adopted as unsanctioned tools, or built on approved platforms without anyone tracking them.

Slopsquatting. An attack in which someone registers a package name that AI coding assistants tend to invent, fills it with malicious code, and waits for an assistant to suggest it. Defended by verifying every package before it enters a build.

Tabletop exercise. A rehearsed walk-through of an incident, with the people who would really be in the room, working through a scenario as new information arrives, followed by a debrief with fixes, named owners, and deadlines.

Tamper-evident. Stored in a way that makes any later alteration detectable. Required of logs that may need to hold up in front of a regulator or a court. In this book, logs should be write-once where possible, so alteration is prevented, and tamper-evident regardless, so any alteration that does happen is detected.

Three doors, three fronts. The book's shorthand for how AI enters a company: the Enterprise front, the workforce front, and the developer front. Each has to be governed differently.

Tiering. Sorting AI systems by consequence so that the depth of review, the intensity of monitoring, and the level of human oversight match what's at stake. The book uses four tiers for risk and three for human oversight.

Transparency. Being able to describe, in plain language and before anyone asks, what a system does, what data it uses, and what it's not good at. See *Auditability* for the difference.

Workforce front. The second of the three doors: the copilots, chatbots, and AI features that arrive on employees' desktops, some sanctioned, some accessed through personal accounts,

and many bundled into software the company already pays for.

Sources

Every external statistic, study, case, and incident cited in this book, by chapter, with a link to the most authoritative source available. Practitioner assessments and examples from my own work are identified as such in the text and are not listed here. Where a claim rests on a lawsuit, the figures are the plaintiffs' allegations and are described that way in the text. Links were last checked in October 2026; laws, lists, and vendor documents change, so check the current version before relying on any of them. Legal and regulatory details are current as of October 2026 and are not legal advice; check the enacted text before relying on them. Examples drawn from my own work are anonymized; identifying details are omitted, and figures are the organizations' own estimates as shared with me, not independently audited.

Chapter 1: Governance on Paper

Cynozure, "Who Owns AI?" (February 2026): 40 percent of companies split AI strategy ownership across multiple executives; 17 percent report no clear owner. Based on a survey of about sixty senior data and AI leaders. cynozure.com

IBM, "New IBM Study Finds CIOs and CTOs Face Growing AI Control Gap" (June 2026): two out of three say they are accountable for AI systems they do not fully control. newsroom.ibm.com

Chief AI Officer prevalence, low end: Data & AI Leadership Exchange, "2026 AI & Data Leadership Executive Benchmark Survey": 38.5 percent of surveyed large firms. survey PDF High end: IBM, "CEOs are Reshaping C-suite Roles for the AI Era" (May 2026): 76 percent. newsroom.ibm.com

Chapter 2: The Common Control Framework

OECD.AI Policy Navigator: more than 900 national AI policies and initiatives from more than 80 jurisdictions and organisations (accessed October 2026). oecd.ai

NIST AI Risk Management Framework (AI RMF 1.0, January 2023). nist.gov

EU Artificial Intelligence Act, full text by article (accessed October 2026). artificialintelligenceact.eu

ISO/IEC 42001:2023, Artificial intelligence management system. iso.org

MITRE ATLAS (accessed October 2026). atlas.mitre.org

OWASP Top 10 for Large Language Model Applications (2025 edition). genai.owasp.org OWASP Top 10 for Agentic Applications for 2026 (released December 2025). genai.owasp.org

Chapter 4: Risk Management

Gartner, "Gartner Identifies Six Steps to Manage AI Agent Sprawl" (April 2026): more than 150,000 agents per average Fortune 500 enterprise by 2028, up from fewer than 15 in 2025; 13 percent believe their agent governance is adequate. gartner.com

Gartner, "Gartner Predicts 40% of Enterprise Apps Will Feature Task-Specific AI Agents by 2026" (August 2025). gartner.com

UpGuard, "The State of Shadow AI" (2025): 81 percent of employees use unapproved AI tools. upguard.com

Harmonic Security, "What 22 Million Enterprise AI Prompts Reveal About Shadow AI in 2025" (January 2026): 665 distinct AI tools observed in enterprise traffic. harmonic.security

Credo AI, "Mastercard and Credo AI: Scaling Generative AI Governance in the Financial Sector to Build Customer Trust" (case study, accessed October 2026). credo.ai

McKinsey, "Deploying agentic AI with safety and security" (October 2025): 80 percent of organizations report risky agent behavior. mckinsey.com

Gartner, "AI TRiSM" (glossary, accessed October 2026). gartner.com

Chapter 5: Protection and Resilience

Cisco, "More than 1 in 4 Organizations Banned Use of GenAI" (January 2024): 27 percent banned; 61 percent restricted. newsroom.cisco.com

Aim Labs, EchoLeak disclosure, CVE-2025-32711 (June 2025). aim.security

Bobby Holley, Mozilla, "The zero-days are numbered" (April 2026): 22 bugs found with Claude Opus 4.6 and fixed in Firefox 148; 271 vulnerabilities found with Claude Mythos Preview and fixed in Firefox 150. The 271 counts discrete defects; Mozilla's advisory lists 41 CVEs. blog.mozilla.org

Google DeepMind, "Introducing Gemini 3.5 Flash Cyber" (published July 21, 2026): 55 unique confirmed issues in the V8 JavaScript engine in Google's testing, versus 47 for Gemini 3.5 Flash and 36 for Claude Opus 4.6. Limited-access pilot. deepmind.google

Anthropic, "Investigating three incidents in our cybersecurity evaluations" (July 2026). anthropic.com

Google, "Sundar Pichai shares news from Google Cloud Next 2026" (April 2026): 75 percent of new code AI-generated and engineer-approved. blog.google Earlier baseline, TechRadar (October 2024): more than a quarter. techradar.com

TechCrunch, "Microsoft CEO says up to 30% of the company's code was written by AI" (April 2025). techcrunch.com

Spracklen et al., "We Have a Package for You! A Comprehensive Analysis of Package Hallucinations by Code Generating LLMs," USENIX Security 2025: 16 models; 440,445 of 2.23 million recommended packages (19.7 percent) were hallucinations; 5.2 percent for commercial and 21.7 percent for open-source models. Repeat test: 500 prompts run 10 times each; 43 percent of hallucinated packages appeared in all 10. usenix.org Full text: arxiv.org

Louis Columbus, "$3.6 Billion in Crunchbase Funding, $96 Billion in M&A, and 10 Agentic AI Security Startups Reshaping 2026," Software Strategies Blog (March 28, 2026): about $3.6

billion in combined funding across ten startups, the author's own tally.

Chapter 6: Privacy and Compliance

Garante per la protezione dei dati personali, "Artificial intelligence: stop to ChatGPT by the Italian SA" (March 2023). gpdp.it Service resumed, TechCrunch (April 2023). techcrunch.com

Garante, fifteen-million-euro fine against OpenAI (December 2024). gpdp.it Annulled by the Tribunale di Roma, March 18, 2026, on jurisdictional grounds (Irish DPC as lead authority); reasoning published May 28, 2026; Wilson Sonsini summary. wsgr.com

Colorado General Assembly, SB26-189, Automated Decision-Making Technology (signed May 14, 2026, effective January 1, 2027), repealing and replacing SB 24-205. leg.colorado.gov

EU AI Act, Article 99 (penalties). artificialintelligenceact.eu Revised high-risk timelines under the Digital Omnibus on AI (in force July 27, 2026): December 2, 2027 for Annex III uses, August 2, 2028 for Annex I products; Gibson Dunn. gibsondunn.com

Chapter 7: Transparency and Auditability

EU AI Act, Article 12 (record-keeping). artificialintelligenceact.eu

Chapter 8: Data Visibility and Quality

Gartner, "Lack of AI-Ready Data Puts AI Projects at Risk" (press release, February 26, 2025): survey of data management leaders, Q3 2024; 63 percent lack or are unsure they have the right data management practices for AI; prediction that through 2026, organizations will abandon 60 percent of AI projects unsupported by AI-ready data. gartner.com

Gartner, "Data Quality: Why It Matters": poor data quality costs organizations at least 12.9 million dollars a year on average (Gartner research, 2020). gartner.com

Gartner, "Lack of AI-Ready Data Puts AI Projects at Risk" (February 2025): organizations will abandon 60 percent of AI projects unsupported by AI-ready data through 2026. gartner.com

Chapter 9: Human Oversight

Estate of Lokken et al. v. UnitedHealth Group, complaint, District of Minnesota (November 2023): allegations that more than 90 percent of appealed denials were reversed (which the plaintiffs characterize as the model's error rate) and that about 0.2 percent of policyholders appeal denied claims. Georgetown litigation tracker PDF

Centers for Medicare and Medicaid Services, "FAQs related to Coverage Criteria and Utilization Management Requirements in CMS-4201-F" (February 2024). cms.gov

Mobley v. Workday, Inc., Northern District of California, docket. courtlistener.com Preliminary collective certification (May 2025); notice period closed March 7, 2026; interlocutory appeal denied July 2, 2026; in discovery, no merits ruling as of October 2026; summary by Norton Rose Fulbright. insidetechlaw.com

Chapter 10: Fairness & Non-Discrimination

Obermeyer et al., "Dissecting racial bias in an algorithm used to manage the health of populations," Science 366 (October 2019). science.org

New York Department of Financial Services, "Report on Apple Card Investigation" (March 2021): underwriting data for about 400,000 New York applicants; no fair lending violation found. dfs.ny.gov

Gebru et al., "Datasheets for Datasets" (2018; Communications of the ACM, December 2021). arxiv.org

Alexandra Chouldechova, "Fair prediction with disparate impact: A study of bias in recidivism prediction instruments," Big Data 5(2) (2017). arxiv.org

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan, "Inherent Trade-Offs in the Fair Determination of Risk Scores," Innovations in Theoretical Computer Science (ITCS 2017). arxiv.org

Uniform Guidelines on Employee Selection Procedures (1978), adopted jointly by the U.S. Equal Employment Opportunity Commission and other federal agencies, 29 CFR 1607.4(D), the four-fifths rule. ecfr.gov

New York City Department of Consumer and Worker Protection, Automated Employment Decision Tools (Local Law 144 of 2021; enforced from July 2023). nyc.gov

Chapter 11: Incident Response

Moffatt v. Air Canada, 2024 BCCRT 149, British Columbia Civil Resolution Tribunal (decided February 14, 2024; chatbot conversation November 11, 2022). canlii.org

Fortune, "AI coding tool wiped a database and called it a catastrophic failure" (July 2025). fortune.com The Register, "Replit makes vibe-y promise" (July 2025). theregister.com

The Register, report of the founder's public account of the Replit incident (July 21, 2025): "eleven times in ALL CAPS"; fake data and reports, including a 4,000-record database of invented people; rollback initially reported impossible, later found to work. theregister.com

EU AI Act, Article 73 (serious incident reporting): no later than 15 days after awareness; 2 days for a widespread infringement or serious disruption of critical infrastructure; 10 days in the event of a death. Serious incident defined in Article 3(49): death or serious harm to health, serious and irreversible disruption of critical infrastructure, infringement of fundamental-rights obligations, or serious harm to property or the environment. artificialintelligenceact.eu

CISA, JCDC AI tabletop exercise (2024). cisa.gov

A Note from the Author

You'll have noticed that this book describes several documents without handing them to you: a council charter, an intake questionnaire, a one-page data design, a standard record for a system, a tabletop scenario, a fairness decision record. That was deliberate. Templates written for everyone fit no one, and the version that works for a regional health plan is not the version that works for a global bank or a software company. The chapters tell you what each document has to contain and why. The right shape for your company is a conversation, not a download.

So let's have the conversation. If you're building any of this and want to compare notes, argue with something I've written, or ask for a starting point tailored to where you sit, the best way to reach me is through LinkedIn: linkedin.com/in/eddieborrero. Send a note that tells me what you're working on. I read them, and I answer.

The people who wrote to me while this book was being drafted made it better. I'd like to keep that going.

Eddie Borrero

