Chapter 2: Why Traditional TPRM Fails at Incident Response

> *Everybody has a plan until they get punched in the mouth.*
>
> — Mike Tyson (also applicable to vendor risk management)

Third-party risk management as a discipline has matured significantly over the past decade. Organizations maintain vendor inventories, conduct due diligence before signing contracts, assign risk tiers, and perform periodic reassessments. Some deploy continuous monitoring through security rating services. The most advanced programs map fourth-party relationships and assess concentration risk.

And yet, when a critical vendor actually gets breached, most organizations discover that none of this preparation translates into an effective incident response. The TPRM program answers the question "Should we have a relationship with this vendor?" It does not answer "What do we do now that this vendor has been compromised?"

This chapter examines why that gap exists and what it would take to close it.

The TPRM Lifecycle: Built for Peacetime

If you work in third-party risk, you're probably familiar with some version of the standard TPRM lifecycle:

Due diligence. Before signing a vendor contract, assess their security posture. Send a questionnaire (or use a standardized assessment like SIG or CAIQ). Review their SOC 2 report. Check their security rating. Identify risks and negotiate contractual protections.

Risk tiering. Classify vendors by criticality. Tier 1 vendors get annual assessments, Tier 2 gets biennial, Tier 3 gets a security rating check. The tiering is usually based on data access and business criticality, though the criteria vary widely between organizations.

Continuous monitoring. Subscribe to a security rating service—BitSight, SecurityScorecard, UpGuard, or similar—and watch for changes. Set alerts for significant rating drops. Some organizations supplement this with news monitoring and dark web scanning.

Periodic reassessment. At defined intervals, re-evaluate the vendor's security posture. Send another questionnaire. Review the latest SOC 2. Update the risk rating. Check if anything has changed.

Offboarding. When the relationship ends, ensure data is returned or destroyed, access is revoked, and contractual obligations around data handling are fulfilled.

This lifecycle is valuable. It catches obvious problems before they become your problems, and it creates a baseline of documentation about your vendor ecosystem. But notice what it optimizes for: *pre-incident assessment*. Every step is about evaluating risk *before* something bad happens. The lifecycle model treats incidents as exceptions—aberrations in an otherwise orderly process of assessment and reassessment.

Incidents Are Not Exceptions

The data tells a different story. The Ponemon Institute reports that 59% of organizations have experienced a data breach caused by a third party. KPMG's 2024 Third Party Risk Management Outlook found that 73% of organizations experienced significant disruption from third-party cyber incidents within the past three years. These numbers have been climbing steadily.

If three-quarters of your peers are experiencing significant third-party incidents within a three-year window, incidents aren't exceptions—they're expected events. A TPRM program that doesn't include structured incident response is like a fire department that inspects buildings for code compliance but has no plan for what to do when one catches fire.

And yet, few TPRM frameworks include a real incident response methodology. The standard lifecycle has four phases—onboarding, monitoring, reassessment, offboarding—and none of them is "respond to an active incident."

Five Specific Failure Modes

When organizations try to use their existing TPRM capabilities to handle a live incident, they run into five specific problems:

Failure Mode 1: The Dependency Knowledge Gap

Your vendor inventory tells you that you have a contract with Okta. Your risk tier tells you it's a Tier 1 vendor. Your SOC 2 review tells you their control environment met the criteria for the trust services principles as of nine months ago.

None of this tells you the things you actually need to know in an incident: Which 47 SaaS applications route their authentication through Okta? What data flows through those integrations? Which business functions go down if Okta is unavailable? Did your support team submit tickets containing HAR files with session tokens during the exposure window?

This is the dependency knowledge gap. TPRM programs track the existence of vendor relationships but not the structure of vendor dependencies. During an incident, you need the structure—the specific technical integrations, data flows, and business function mappings—and you need it immediately. Most organizations discover they don't have it.

The Shadow Dependency Problem

The dependency gap is often worse than it appears. When organizations conduct thorough dependency mapping, they typically discover 30–40% more vendor relationships than were documented. These "shadow dependencies" arise from departmental SaaS purchases, free-tier tools, and integrations set up by individual engineers. During an incident, an unknown dependency is an unknown exposure.

Failure Mode 2: No Framework for Conditional Probability

The most widely adopted model for quantitative cyber risk assessment—FAIR—is designed to answer the question: "What is the annual probability and expected magnitude of a specific loss scenario?" It does this by estimating Threat Event Frequency (how often an event happens) and Vulnerability (the probability that a threat event becomes a loss event), then multiplying by Loss Magnitude.

This is the right model for prospective risk: "What's our annual risk from a vendor data breach?" But it's the wrong model for incident response, because the threat event has already happened. You don't need to estimate its frequency—it's not a question of if but of whether you specifically are affected.

What you need is conditional probability: given that a vendor incident has occurred with certain characteristics, what is the probability that your organization's specific usage falls within the impact zone? This requires a different decomposition than FAIR provides out of the box. It requires thinking about scope probability, affected probability, and exploitability—concepts that map loosely to FAIR's structure but require adaptation for the incident response context.

Without this adaptation, teams default to binary thinking: "We use this vendor, so we're affected" or "The vendor said only a subset of customers were affected, so we're probably fine." Neither of these is a useful risk assessment.

Failure Mode 3: Bias Under Pressure

Incident response is one of the worst possible environments for human judgment. Time pressure, incomplete information, high stakes, and emotional arousal all conspire to amplify cognitive biases that are already problematic in calm, deliberate analysis.

The **availability heuristic** makes the most recent or most vivid incident dominate your thinking. If the last vendor incident your organization experienced was a catastrophe, you'll overweight the probability that this one will be too. If the last one was a false alarm, you'll be inclined to dismiss this one.

The **anchoring effect** means your first estimate—however poorly informed—will disproportionately influence your final assessment. If someone in the meeting says "I think there's a 70% chance we're affected," subsequent discussion will orbit around that number even if it was pulled from thin air.

Groupthink is amplified by the crisis atmosphere. When the CISO says "I think this is serious," junior team members are unlikely to push back, even if they have information suggesting otherwise.

Research from the Good Judgment Project demonstrated that calibration training—teaching people to be better probability estimators through structured practice and feedback—can significantly reduce these biases. Superforecasters who underwent this training outperformed professional intelligence analysts by 30% on accuracy metrics. But almost no TPRM program includes any calibration component.

Failure Mode 4: No Decision Structure

After the assessment—however informal—comes the decision. And this is where things get truly messy.

The fundamental decision in vendor incident response is a four-way choice: accept the risk and continue as-is, mitigate by implementing compensating controls, reduce the dependency scope, or exit the vendor relationship entirely. Each option has different costs, different residual risks, and different time horizons.

In practice, though, this decision is rarely structured as a cost-benefit analysis. It's more often driven by the emotional temperature of the room. If people are scared, the decision tilts toward exit—even when the exit cost vastly exceeds the expected loss from the incident. If people are complacent, the decision tilts toward acceptance—even when the exposure clearly exceeds the organization's stated risk tolerance.

Worse, the decision is rarely anchored to the organization's formal risk appetite. Most organizations have risk tolerance statements, but those statements are expressed in terms ("low/medium/high," or "within acceptable limits") that don't connect to the quantitative reality of a specific incident. When your expected loss from a vendor incident is $400K±15%, is that within tolerance? Nobody knows, because tolerance was never expressed in dollars.

Failure Mode 5: Institutional Amnesia

Perhaps the most insidious failure mode is the one that compounds over time: organizations don't learn from vendor incidents because they don't systematically capture what happened.

When Okta has its next incident—and statistically, it will—the analysis you performed during this incident should be your starting point. The dependency mapping, the exposure analysis, the compensating controls you implemented, the decision you made and why—all of this context should be immediately available to the team handling the new incident.

In most organizations, it won't be. The dependency mapping was done in a spreadsheet that someone saved to their desktop. The exposure analysis was debated verbally in a meeting and partially captured in meeting notes that nobody can find. The compensating controls were assigned as action items that were tracked for a few weeks and then forgotten. The decision rationale exists in a Slack thread that's since been archived.

The result is that every vendor incident is treated as a novel event, even when the organization has faced the exact same scenario before. Analysis work doesn't compound. Institutional knowledge doesn't build. The same questions are relitigated from scratch each time.

We call this accumulated unfinished work *mitigation debt*—the gap between compensating controls you committed to implementing and controls you actually implemented. It's like technical debt, but for risk. And like technical debt, it accumulates silently until the next incident forces a reckoning.

The Missing Layer

To be clear, this chapter is not an argument against existing TPRM practices. Due diligence, risk tiering, continuous monitoring—these are all valuable and necessary. The argument is that they're *insufficient*. They address the steady state but not the crisis. They're the building code inspection, not the fire response plan.

What's missing is a structured methodology specifically designed for the moment when a vendor incident occurs—one that can answer "Are we affected?" with quantitative rigor, support defensible decisions under time pressure, and preserve institutional knowledge for the future.

That's what we'll build in the rest of this book, starting with the foundation: mapping your vendor dependencies in a way that makes incident response possible.