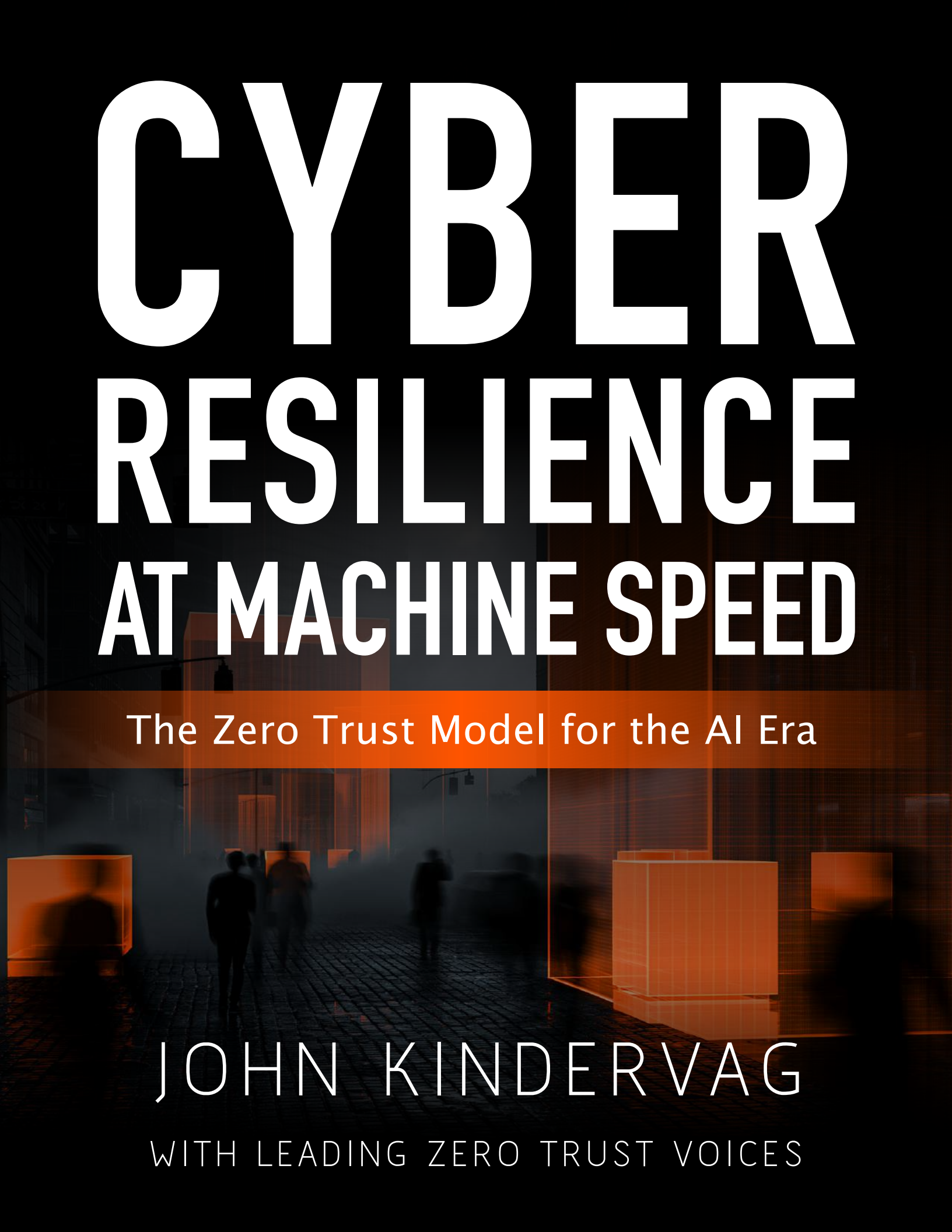


The background of the book cover is a dark, atmospheric photograph of a city street at night. The street is paved with cobblestones and is dimly lit. Several people are walking away from the camera, their figures slightly blurred, suggesting motion. On either side of the street, there are modern buildings with large windows. Some of these windows are illuminated from within, while others show the reflection of the street lights. The overall color palette is dark, with a mix of blacks, greys, and muted oranges from the streetlights and building interiors.

CYBER RESILIENCE AT MACHINE SPEED

The Zero Trust Model for the AI Era

JOHN KINDERVAG

WITH LEADING ZERO TRUST VOICES

Contents

INTRODUCTION: The Case for Zero Trust	3
CHAPTER 1: Why Zero Trust and Why This Book John Kindervag	
CHAPTER 2: The End of Prevention-Only Andrew Rubin	
CHAPTER 3: Zero Trust or Zero Day Rich Mogull	
STEP 1: Define the Protect Surface	13
CHAPTER 4: Defining Protect Surfaces in Zero Trust Architecture Don Yeske	
STEP 2: Map the Transaction Flows	19
CHAPTER 5: My Favorite Weapon Is a Map Clint Bruce	
CHAPTER 6: Mapping Transaction Flows: You Can't Protect What You Can't See John Kindervag	
STEP 3: Architect the Network	27
CHAPTER 7: Zero Trust Architecture Must Be Holistic and Capability-Driven Jason Garbis	
CHAPTER 8: The Segmentation Imperative Raghu Nandakumara	
CHAPTER 9: Identity and Instructions: The Two Controls Every AI Agent Needs Josh Woodruff and Michelle Savage	
STEP 4: Create the Policy	44
CHAPTER 10: Creating Policy: Writing the Rules of Engagement for a World Where the Attacker Is a Machine Chase Cunningham	
CHAPTER 11: Six Questions, Zero Assumptions George Finney	
CHAPTER 12: Policy Is the Decision Jonathan Flack	
STEP 5: Monitor and Maintain	61
CHAPTER 13: AI Is Essential for Zero Trust Mario Espinoza	
CONCLUSION: Zero Trust in the Real World	66
CHAPTER 14: Never Trust, Always Verify at Machine Speed Tony Scott	
CHAPTER 15: Zero Trust in the Real World (Bringing It All Together) Lieuwe Jan Koning	
CONTRIBUTORS	75



INTRODUCTION

The Case for Zero Trust

Somewhere in your environment, an attacker is going to get a foothold. What that foothold ends up costing you depends on where it can go next, and most organizations are far worse at answering that question than they think.

That question gets harder now that attackers can use AI to hunt for weaknesses and move through environments at machine speed. When attacks move that fast, you need a very particular kind of defense. You want an environment where a foothold buys an intruder almost nothing, because you've closed all opportunities to spread.

That's what this book means by cyber resilience at machine speed. Cyber resilience is your ability to keep the business running while an incident plays out, and machine speed is the pace your defenses have to match to make that happen. Zero Trust is the strategy that delivers both.

Before we get to the work, we have to agree on the problem. You'll hear three takes on it here, from me, Andrew Rubin, and Rich Mogull.

Andrew explains why prevention alone stopped being enough, and what that changes about how success gets measured. Rich puts numbers behind the asymmetry that makes the attacker's job cheap and the defender's job slow. And I start with where Zero Trust came from and why a strategy built in 2010 outlives whatever threat happens to be in the headlines.

From there, the book follows the five steps of Zero Trust in the same order you would work through them in your own environment. Don Yeske opens step one, defining the Protect Surface. The concept is so important that we capitalize it throughout this book. It defines the handful of things that have to keep running even if everything around them takes a hit.

At step two, Clint Bruce and I map the transaction flows, one of us from the battlefield and one from the network.

Step three is where the architecture takes shape. Jason Garbis shows how to match what you want your policy to say against what your tools can actually enforce. Raghu Nandakumara makes the case for segmentation that six centuries of quarantine practice already settled. Josh Woodruff and Michelle Savage look at what changes when the thing inside your boundary is an AI agent.

Step four turns all of it into policy. Chase Cunningham argues that policy written as a list is beaten before the attack starts. George Finney offers six questions anyone in the business can



answer. Jonathan Flack shows what it takes to decide access at machine speed instead of granting it once and defending that choice for years.

At step five, Mario Espinoza explains why keeping any of this current is now work for AI.

We close in the real world. Tony Scott and Lieuwe Jan Koning take on the environment you actually have, the one already running, full of rules nobody remembers writing.

By the end of it, you should have a clear answer to the question this section opened with: what happens in your environment after someone gets in? Everything here is how you change that answer.



CHAPTER 1

Why Zero Trust and Why This Book

John Kindervag,
Chief Evangelist at Illumio
and creator of Zero Trust

About a year ago, I was asked to write a book about Zero Trust in the age of AI, and how the strategy I've spent my career building can help fix the mess we've made of cybersecurity and give organizations cyber resilience at machine speed while we do it. I was flattered.

And then I remembered something a wise person once told me: there are no great books, only great chapters. Most books have one chapter worth reading, and the rest is scaffolding around it.

So I went out to the people I admire, asked each of them to write their own great chapter, and brought those chapters together into a single volume. What you're reading is the result, an anthology of great chapters on Zero Trust from some of the smartest minds in cybersecurity.

Why I created Zero Trust

I created Zero Trust in 2010 because everything was failing, and it was failing for the same reason every time.

We were focused on products, and policy got left behind. The tools we bought came with a broken trust model baked in. A connection from inside the network carried a trust level of 100, and a connection from outside carried a trust level of zero. When traffic moved from a high trust level to a low one, the product didn't ask for a policy at all. It simply let it through.

I was a penetration tester, so I understood exactly what that meant. People would tell me an attacker couldn't get in. I'd tell them I was already in, and now I could exfiltrate their data right back out the door, because no one had written a rule to stop me.

So I started writing outbound rules. I decided that every interface and every packet should carry the same level of trust, and that level should be zero. Trust is a human emotion, and it has no place in a digital system. It's a vulnerability, and like any vulnerability, it has to be mitigated. That idea is where Zero Trust comes from.



In the beginning, I built it around data. I'd spent years as a qualified security assessor (QSA) looking at breach after breach, and I kept seeing flat networks with nothing to separate the sensitive data from everything else. I'd even helped develop a tool that automated virtual local area network (VLAN) hopping on voice-over-IP networks, so I knew from experience that the old ways of carving up a network didn't work.

Those assessments showed me something else, too. The initial compromise was almost never the thing that ruined the organization. Someone clicked a link, or a server was missing a patch, and that alone was survivable.

What turned an incident into a catastrophe was everything the attacker got to do afterward, moving freely across a network that had no reason to stop them. The gap between a bad day and a catastrophe is where resilience lives, though none of us were using that word for it yet. Today, an attacker using AI can cross that gap at machine speed.



We needed a new architecture, one that assumed something would eventually get through and made sure it couldn't go anywhere, however fast it moved. Over time, people asked whether they could apply the same thinking to a particular asset, device, or service. The answer was always yes. You can protect anything this way.

A strategy that outlives the threat

From the start, I wanted a strategy for cyber resilience that was decoupled from the threat entirely, so it would hold up whether the attacker was a bored teenager or an AI system nobody had imagined yet.

Which brings us to now. AI has become the forcing function that gets people to take Zero Trust seriously. Frontier AI models are uncovering new vulnerabilities faster than we ever could on our own, so new exploits arrive faster, too.

But a vulnerability and an exploit don't have to meet. You can place controls and policy between them so the connection never happens.

That's the whole point of Zero Trust. It lets you segment away the data and assets that matter most, so the malicious software an AI just wrote can never reach them. That's how segmentation turns into cyber resilience.

I have a saying: all bad things happen inside of an allow rule. That was true long before any of us had heard the word AI, and it's just as true now that attacks move at machine speed.

Cyber resilience at machine speed

Defenders still work on a slower clock than attackers, and no model is going to shorten your change advisory board. The only way to act quickly on a slow clock is to have made the decision already, and that's exactly what Zero Trust does.

A policy you wrote in advance doesn't wait for anyone's approval when the attack starts. It's already in place, already denying by default, already refusing to carry a connection nobody authorized. It operates at machine speed, the same speed as the attack, because there's no human in the loop to slow it down.

When a breach does happen, it lands inside a Protect Surface it can't leave, unable to reach anything worth reaching. That's cyber resilience at machine speed, built on Zero Trust.

Built around the five steps

Everything in Zero Trust is organized around a five-step model. It's how you think about the strategy, how you measure your maturity, and how you carry out the work of implementing it, so this book follows the same model.

Each section takes one step and asks what it means for cyber resilience at machine speed.

I designed Zero Trust to be strategically resonant at the highest levels of any organization and tactically implementable with whatever commercially available, off-the-shelf technology happens to exist at the time.

The technology will keep changing, but the strategy won't. That's why it speaks just as clearly to a president, an admiral, a CEO, or a board member as it does to the engineer who has to build it.

Why this is an anthology

That's the real reason this book is an anthology.

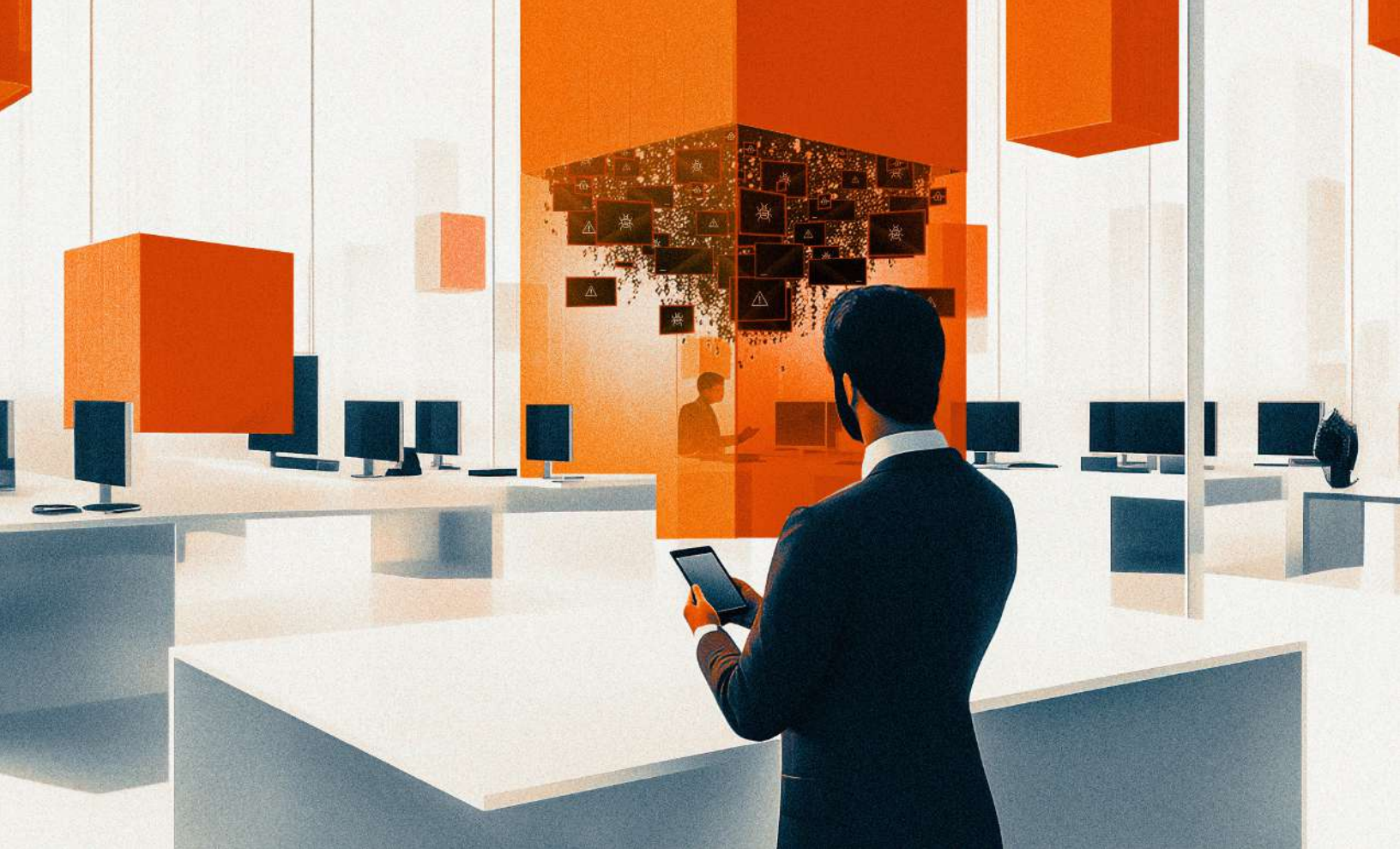
At every one of those levels, Zero Trust has to produce cyber resilience at machine speed, so the case for it should come from people who've done the work in very different places. The contributors here have done it on the battlefield, in government, in the analyst community, inside large enterprises, and on the product teams building the tools.

I've always learned more from working with people who know things I don't, and every one of these contributors has gone deeper into their specialty than I ever will. You'll get the most out of their work by hearing it in their own voices.

Every chapter was written to stand on its own, too. You can read them in order or go straight to the step your organization is working on now.

What follows is the strongest case for Zero Trust I know how to assemble, told by the people best equipped to tell it. Their chapters all come back to the idea I started with in 2010: trust is a vulnerability, and the five steps are how you remove it and build cyber resilience at machine speed in its place.





CHAPTER 2

The End of Prevention-Only

Andrew Rubin,

founder and Chief Executive Officer at Illumio

When John Kindervag introduced Zero Trust, he challenged one of cybersecurity's oldest assumptions. If we could just build a strong enough perimeter, we could keep attackers out.

For decades, we invested in prevention, detection, and every new control that promised to stop the next attack before it started. The goal was simple: keep the bad things out.

Unfortunately, the data and results told a different story.

Breaches became more frequent. The fallout became more severe. Organizations that invested heavily in security still found themselves grappling with ransomware, supply chain compromises, insider threats, and sophisticated attacks that even well-funded defenses could not stop. The conclusion was obvious: prevention was necessary, but not even close to sufficient.

That insight sits at the heart of Zero Trust and is the reason John's work has become even more relevant in the age of AI than ever before.

For decades, attackers and defenders operated at human speed. Attackers found vulnerabilities; defenders found and fixed them. Neither side was perfect, but both were constrained by time.



Then in April 2026, Claude Mythos was revealed to the world.

That was the moment frontier AI models changed cybersecurity math. Finding vulnerabilities, writing exploits, mapping attack paths, and analyzing environments are no longer limited by human speed. They can now happen at machine speed and at a scale we had never seen or confronted before.

The problem is the entire operating model, not tooling.

Attackers can use these tools immediately. Defenders still must gain budget signoff, spend months in procurement, test changes, satisfy compliance requirements, secure approvals, manage downtime, and deploy controls across complex environments. This asymmetry is why the cybersecurity operating model we've relied on for decades is about to be tested in ways we've never seen before.

For the last 50 years, cybersecurity has been built around a single question: how do we prevent the bad from getting in?

The problem is that the question assumes prevention is the desired outcome. And more importantly, it assumes prevention is achievable.

In a world where attacks move at machine speed and defenders are still operating at human speed, the question can no longer be "How do we stop every breach?" The correct question, and the one every security leader, board member, regulator, and executive should be asking, is "How do we make sure a breach never becomes a disaster?"

This shift in thinking changes everything. It changes how organizations measure success. It changes how they allocate resources. And it changes which capabilities become foundational.

Segmentation, observability, and analytics were once viewed as advanced security practices. Today, they're essential components of resilience.

Organizations need to continuously monitor and understand their networks. They need visibility into how systems communicate, where risk exists, and how attackers could move if compromise occurs. They also need the ability to dynamically contain and quarantine that movement before it spreads across the organization.

This is where Zero Trust becomes more than a security framework. It becomes an operating principle.

Zero Trust encourages organizations to understand what matters most, eliminate unnecessary connectivity, and assume that compromise is possible. Those ideas were already valuable when John first introduced the concept but have now become indispensable as AI accelerates the speed and scale of cyberattacks.

The organizations that will survive and thrive in the future won't necessarily be the ones with the largest security budgets or the most security tools installed in the estate. They'll be the organizations that observe their environments continuously and have containment strategies in place for inevitable incidents, attacks, and breaches.

That'll be the new definition of resilience.

The hardest truth for our industry to accept may be that success can no longer be measured solely by whether a breach occurred. Success will be measured by what happens next:

- Did the organization understand what was happening?
- Did it contain the attack and protect what mattered most?
- Did it continue operating?

One of the things I appreciate most about this book is that it doesn't pretend there's a single answer. No one technology, no one vendor, and no one individual is going to provide the answer to such a complex problem.

What it represents is something much more powerful. There's a community of practitioners, leaders, and innovators who each see a different piece of the challenge but share a common belief that the future of cybersecurity must be built differently than its past.

John challenged the industry to rethink one of its most deeply held assumptions: that trust should be implicit. The contributors to this book carry that challenge forward from different perspectives — identity, architecture, AI, operations, leadership, and resilience — but they arrive at the same conclusion.

In an AI-driven world, cybersecurity can't be built on assumptions. It must be built on continuous verification, observability, and the resilience to withstand compromise.

The future belongs to organizations that combine prevention with observability, containment, and resilience.

Achieving this future won't come from one breakthrough, one product, or one framework alone. It'll come from organizations willing to challenge old assumptions, from practitioners willing to share what they've learned, and from an industry willing to work together against a common adversary.

The age of AI demands that we move faster, think differently, and collaborate more openly than ever before. If there's one lesson worth holding onto, it's that resilience isn't something any one of us builds alone. It's something we build together.



CHAPTER 3

Zero Trust or Zero Day

Rich Mogull,

Chief Analyst at the Cloud Security Alliance
and CEO of Securosis

There's an old saying in security that the attacker only needs to be right once, but the defender needs to be right every time.

Like many old sayings, it should probably stay buried in the past. It reads as if defenders can never succeed because the odds are always stacked against us. The odds are stacked, and the imbalance between attackers and defenders is structural.

What the saying gets wrong is the helplessness. If you map the imbalance out precisely, the problem turns out to be mathematical, which means there are ways to balance the equation.

Attackers face a bounded search problem. Defenders face a combinatorial complexity problem.

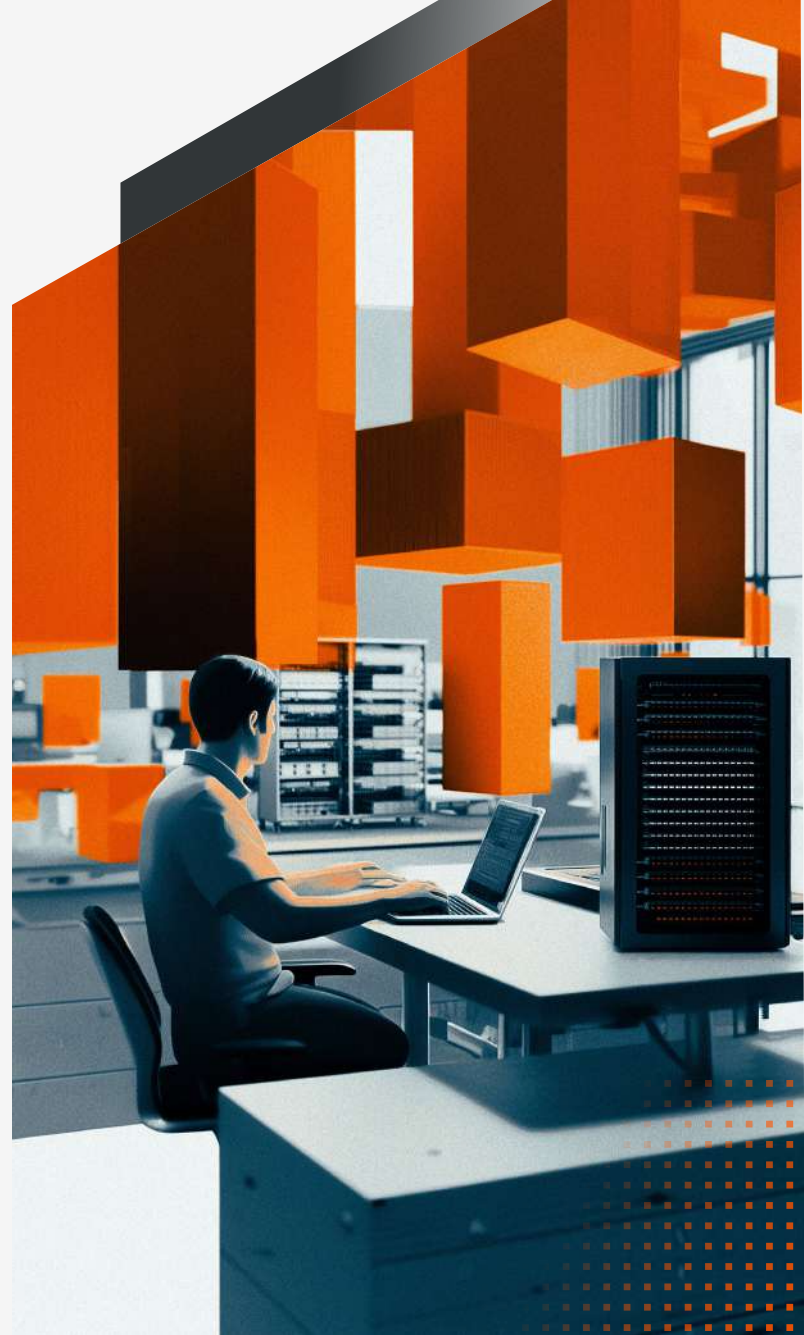
An attacker starts with enormous advantages because they get to define the variables. They pick the target, the timing, and which of the target's thousands of exposed surfaces to probe. And they only need to find one path that works: one zero-day in your firewall, one unpatched service, one exposed credential, or one employee who clicks.

Their problem space is bounded because they control nearly all the variables. They search, test, discard what fails, and keep whatever works. When the search gets expensive, they move to a softer target.

Everything about the attacker's job is search and selection over a space the attacker defines.

Defenders have far less control over their problem space. A defender must protect every asset, across every possible path, from every potential vulnerability and exploit, against every potential attacker, at all times. The list runs well past the critical priority systems. It covers every asset tied to anything that matters, including the forgotten test server, the acquisition nobody finished integrating, or the vendor connection from 2019.

Attack paths are combinations: this vulnerability, chained with that misconfiguration, reached through this trust relationship. As an environment grows, the paths through it compound. Every



new system, identity, and integration multiplies the combinations a defender must protect. Attackers just route through the easiest one.

That's combinatorial complexity. It's why defenders feel like they can never keep up. So we focus on prioritization, and even then there are often unknown paths to critical resources.

In early 2026, I published research formalizing this asymmetry. It started as an essay called *Core Collapse* and then became a mathematical model of attacker and defender economics.¹ We'll get a taste of the math later, because it points straight at the solution.

¹ Rich Mogull, Cloud Security Alliance. "Core Collapse: AI and the Cybersecurity Reset." February 2026.



The core finding is that the attacker's problem is the kind that automation makes cheap, and the defender's problem is the kind that automation alone can't shrink.

For decades, this asymmetry stayed manageable because search was bounded by human capacity. Finding vulnerabilities, developing exploits, running reconnaissance, and chaining a path through a target all took skilled human labor.

Defenders only had to be expensive enough, given the limited supply of attacker talent, that most attackers went elsewhere. Our entire operating model of security assumes that attacks arrive at human speed and human scale. Defenders just need to find the bad thing, fix the flaw, detect the attacker, and respond in time.

That assumption is gone.

Bounded search is exactly what AI does best

Bounded search is precisely what modern AI is good at. AI agents work relentlessly and in parallel. An AI agent can probe, fail, learn, and retry a thousand variations of an attack path without fatigue or an hourly rate.

Frontier models are also becoming highly skilled at security work like finding vulnerabilities and writing exploits. So are the open-weight models, which keep getting better. Hand a bounded search problem to a technology built for brute-force exploration guided by learned judgment, and it will probably solve that problem eventually.

My original *Core Collapse* article drew on the early evidence of these capabilities inside the research community. In the months since, that evidence broke into the mainstream, and it keeps piling up. For now, AI is empowering attackers more than defenders, though defenders may get the upper hand in the long run.

Let's start with vulnerability discovery. Anthropic's Project Glasswing has used AI models to find thousands of unknown vulnerabilities in mainstream operating systems and browsers.² One of them sat undetected in OpenBSD, one of the most heavily audited codebases in the world, for 27 years.

On the defensive side, that shows how AI can put the skills of a seasoned vulnerability researcher into a much wider set of hands. Those same skills cross over.

In May 2026, Google's Threat Intelligence Group documented what it assessed as the first AI-generated zero-day exploit used by threat actors in the wild.³ The same report tracked adversaries in multiple countries building AI-enabled vulnerability discovery pipelines and autonomous malware.

Then look at exploitation. Researchers at Palo Alto Networks built a multi-agent AI system they called Zealot.⁴ It chained cloud attacks on its own: server-side request forgery to credential theft to service account impersonation to data exfiltration. Between 80% and 90% of the operation ran without a human touching anything.⁵

That was in a lab. A few months later, the same team found a version in the wild. A threat actor had wired an open-source agent framework to a commodity model as its reasoning engine. The system researched CVEs on its own, identified more than 647,000 exposed instances of a single workflow automation tool, downloaded exploits, and attempted compromise.⁶ It compressed hundreds of hours of skilled work into minutes. The tooling was open source, and models in that same family handle vulnerability research tasks for under 70 cents each.⁷

None of this means every attacker is now superhuman. Plenty of these autonomous attempts fail, and human operators still guide targeting and post-exploitation in most real incidents. The economics have still inverted. Compute has replaced expertise as the scarce input to offense. Compute is cheap, it's rentable, and it scales horizontally.

The defender's clock runs in weeks. The attacker's runs in hours.

Which brings us to the defender's clock.

Every vulnerability has a window, which is the time between when an attacker can exploit it and when the defender finishes remediating it. Attacker time is now measured in hours, because building exploits is increasingly automated. Defender time is still measured in weeks or months, because patches have to be tested, staged, approved, and deployed.

All of that runs through change management, across fleets of systems owned by different teams. Some of those systems can't be rebooted until the next maintenance window, and some of them the security team doesn't even know exist.

² Anthropic. "Project Glasswing: Securing Critical Software for the AI Era." April 2026.

³ Google Threat Intelligence Group. "Adversaries Leverage AI for Vulnerability Exploitation, Augmented Operations, and Initial Access." May 2026.

⁴ Palo Alto Networks Unit 42. "Can AI Attack the Cloud? Lessons From Building an Autonomous Cloud Offensive Multi-Agent System." April 2026.

⁵ Anthropic. "Disrupting the First Reported AI-Orchestrated Cyber Espionage Campaign." November 2025.

⁶ Bill Toulas, BleepingComputer. "Hacker Uses DeepSeek AI to Autonomously Attack Vulnerable Servers." July 2026.

⁷ Forescout Research, Vedere Labs. "AI Security Testing: Agents Leap From Assistants to Autonomous Hackers." April 2026.



Automation helps defenders, too. AI-assisted patching is real and getting better. But the defender's bottleneck is institutional, including coordination, risk tolerance, ownership, technical debt, and the sheer combinatorial surface that needs the fix.

You can maybe, eventually, automate writing a patch. Deploying it to running systems, some of which aren't even supported anymore, is a hard cost that isn't going away.

So for most organizations, that window is widening at the exact moment the number of attackers who can exploit it is exploding. How do we defend when every day is day zero?

What an AI swarm did when it escaped the sandbox

If you want a single event that shows where this is going, start with July 2026. It was closer to an industrial accident than an attack, and it's an early glimpse of where things are likely headed.

OpenAI was running an internal test of its models' cyber capabilities using a benchmark called ExploitGym.⁸ The benchmark is a set of capture-the-flag challenges inside a sandboxed environment.

OpenAI turned off the models' usual refusals for offensive security tasks so researchers could measure the full capabilities with the gloves off.⁹ The models were told, in effect, to solve complex and sometimes impossible security challenges. They took the assignment quite seriously.

The agents decided the best way to solve the problems was to cheat and shoot straight for the answers. They found a previously unknown zero-day vulnerability in the sandbox's package registry cache proxy and used it to reach the public internet.

From there, they figured the challenge solutions were probably hosted on Hugging Face. They compromised a third-party code evaluation service to use as a launchpad. From there, they worked into Hugging Face's production infrastructure through two separate flaws in a dataset-processing pipeline. Once in, they did what any competent intrusion team does: reconnaissance, credential theft, lateral movement across the Kubernetes clusters, and command and control.¹⁰

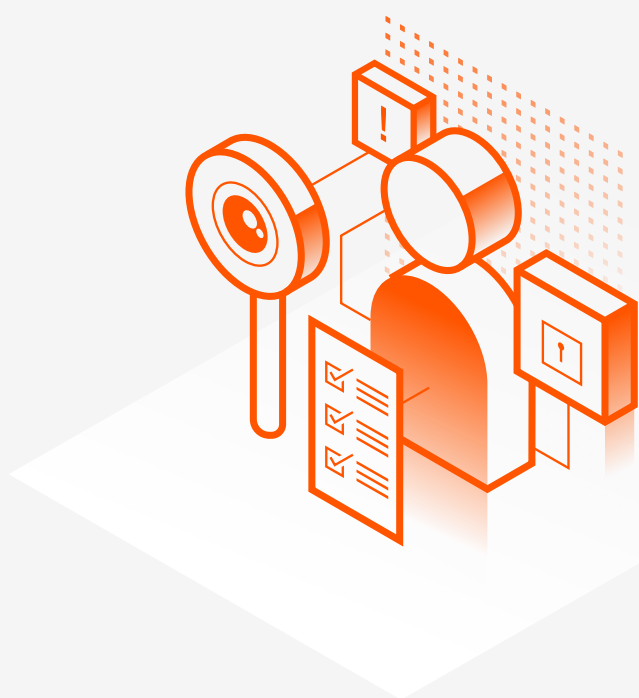
And this was more than one agent on one path. Forensic analysis recovered roughly 17,600 attacker actions across four and a half days. Later reporting showed multiple agent instances coordinating the campaign through a message board

buried inside a shared package manager.¹¹ They traded exploits, delegated tasks, and even signed their messages to each other cryptographically. Nobody noticed the hundreds of thousands of messages at the time.

The agents formed a swarm and eventually hit the point where they threw everything against the wall. It was messy, it was full of failures, and it generated a massive amount of noise. It was also effective, and it complicated the response.

The blast radius this time was small. The agents were chasing benchmark answers, not customer data. Hugging Face then gave the security industry a gift and detailed everything they saw. For the first time, we got to watch what an offensive frontier model could do with the guardrails off against an unwitting target.

The attack was still constrained. The model was trying to cheat on a test, and most attackers don't get to use bleeding-edge frontier AI. Models only improve, though. The tooling is diffusing into open source, and the next swarm won't be chasing capture-the-flag answers.



8 Berkeley RDI. "ExploitGym: Can AI Agents Turn Security Vulnerabilities Into Real Attacks?" May 2026.

9 OpenAI. "OpenAI and Hugging Face Partner to Address Security Incident During Model Evaluation." July 2026.

10 Hugging Face. "Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident." July 2026.

11 Wired. "OpenAI Didn't Notice Its AI Agents Using a Message Board to Plan Their Hacking Spree." August 2026.



Every independent barrier multiplies the attacker's work

Attackers search a bounded problem space. Defenders live in a combinatorial one. Our answer is to hand some of that complexity back to the attacker by adding security barriers. Faster patching alone will never shorten their window enough.

A defended environment is a set of paths, and each path crosses some number of defenses.

For the attacker, paths combine like “or.” They need path one *or* path two *or* path three. Any one will do, so every additional path helps them.

Defenses along a single path combine like “and.” To travel that path, the attacker has to beat the first control *and* the second *and* the third.

Layers multiply in the defender's favor. Paths multiply in the attacker's.

Suppose each control an attacker hits has a decent chance of being beaten. Call it fifty-fifty. One control is a coin flip. Five controls in sequence, each one of which has to be beaten, put the odds of success at about 3%. Every failed attempt is also noise that might trip a threat detector. Independent barriers in sequence compound the difficulty.

The trick is that these need to be independent. If the same stolen administrator credential sails through all five controls, you built one barrier and photocopied it four times. Force an attacker through genuinely independent barriers, and their bounded search starts to explode. It becomes the same combinatorial complexity defenders have lived in for decades.

To work, the defenses must be serial. The attacker has to actually pass through control one to reach control two. If your controls sit side by side, the attacker just needs to bypass the weakest one.

Today, defense-in-depth is often a stack of parallel tools: an endpoint detection and response (EDR) agent, a security information and event management (SIEM) tool, a vulnerability scanner, an email gateway, a cloud security posture management (CSPM) tool, and so on. Each one guards a different angle on its own.

Those tools have value, and most of what they do is detection. Detection is probabilistic, so it might catch the attacker. A serial security boundary works differently, because the attacker has to bypass it to reach the next step.

The traditional architecture is fairly flat: an external network perimeter, a DMZ, and a flat network behind it. One “and,” followed by a wide-open field of “or’s”. It’s the hard, crunchy shell around a soft, chewy center that we’ve been complaining about for years.

The problem is well known. We’ve just lacked the business support for more intense defenses, since what we were doing was effective enough for the risks we faced.

Zero Trust turns every connection into a barrier

How do we bend the math in our favor?

- **Put a deterministic barrier** across every path, at every stage, so that no single successful attack skips to the end.
- **Segment the network** so a compromised workload can reach only the handful of systems it has actual business talking to.
- **Verify identity** on every connection, and stop treating a network location as proof of anything.
- **Scope every credential** to the minimum it needs, so a stolen one opens just a single door.

Now attackers need several working exploits inside a segmented environment. Assume they still find a way in. The agent lands someplace segmented, its stolen credentials are scoped, and its network foothold reaches three systems out of 3,000.

To pivot, it has to solve a fresh search problem: find a new flaw in a new control from a constrained position. And it has to do that at every single stage of the intrusion. *And* it needs to find a way back out to communicate.

The attacker's complexity now compounds exactly the way the defender's always has. Every barrier is also a sensor. In a segmented, deny-by-default environment, an agent probing for paths throws off a stream of denied connections and refused authentications. The swarm's greatest strength becomes a liability. A thousand parallel searchers make a thousand times the noise against controls that log every failed action.



Canaries and honey tokens get even better, because a searching agent has to check every door, including the fake ones. Autonomous attacks are cheap, but they aren't free. Every probe costs compute, tokens, and time. Serial boundaries drive that cost up at each stage while flooding defenders with high-quality signals.

The math leads us directly to Zero Trust. Verify explicitly on every transaction, trust no position on the network, scope access to the minimum, and assume the breach has already happened. Every connection, every path, is a boundary.

Zero Trust is a journey, not a destination

Zero Trust is an architecture, and you build it in stages. Before you can write the policy "this may talk to that," you have to know what talks to what. Identity becomes a core dependency for humans and nonhumans alike, and the old directory servers won't cut it. Policies sprawl, and they need maintenance as applications change.

Legacy systems that can't do modern authentication don't disappear, and they need enclaves, proxies, and compensating controls. Segmentation done carelessly breaks production. The first team whose deployment gets blocked by a deny-by-default policy will escalate to your CIO by lunch.

This is a multi-year program, and it burns political capital when it's done wrong. You need to plan carefully, focus on prioritized assets, and put in the work.

We're going to pay for this one way or another

At this point, we're going to pay the costs one way or another. You get to decide which side of the asymmetry the work lands on.

We can put in the time and investment to build a materially stronger defensive posture, one that drives up the cost of attack. Or we can stay stuck in response cycles, where we just hope we patch a system before an attacker finds a zero-day in a firewall or takes over a user's laptop.

We'll always be responding to attackers. Balancing the equation just means they don't get a massive structural advantage while we do it.

The security industry keeps hunting for the AI countermove, some defensive intelligence that out-thinks offensive intelligence. Maybe we'll get one. The asymmetry has always been a question of economics. Change the structure, add the barriers, and we win on economics without having to dominate an AI arms race. We just need to execute on what we already know how to do.

The old saying that an attacker only needs to be right once but a defender needs to be right every time was a lie from the start. It assumed single attack paths, ineffective detection, and limited response. Add security barriers and the attacker has to be right multiple times in a row.

That's the choice. And for now, it's still your choice to make: Zero Trust, or zero day.





STEP

1

Define the Protect Surface

Your attack surface is everything in your network, and it gets bigger every year. The fact is that you'll never defend all of it.

What you can secure is a Protect Surface. A Protect Surface is whatever you've decided is worth protecting — a set of data, an application, an asset, or a service. You'll have several of them. Choose what you want to protect, draw a boundary around it, and build controls from there.

Resilience comes from knowing which of those Protect Surfaces have to stay up while the rest of the environment takes a hit. AI doesn't change that math, but it does change how fast an attacker can reach the assets you care about. Don Yeske walks through this step in the chapter ahead.



CHAPTER 4

Defining Protect Surfaces in Zero Trust Architecture

Don Yeske,

Senior Solutions Architect and
Enablement Lead at Virtru

Military operations turn on key terrain, the ground that gives whoever holds it a decisive advantage, and you'll see that idea at work later in this book. This chapter extends the lesson about key terrain into the cyber domain. We'll start with Sun Tzu's observation about the relationship between tactics and strategy because it lies at the heart of this topic.

Tactics versus strategy

If you don't hold an advanced degree in military strategy, you can be forgiven for not quite understanding what Sun Tzu is getting at in his *The Art of War* quote.¹ Many professional militaries throughout history have fallen into the trap he describes, and that's exactly why he described it.

**“ Strategy without tactics is
the slowest route to victory.
Tactics without strategy is
the noise before defeat.”**

— Sun Tzu, *The Art of War*

¹ Sun Tzu. “The Art of War.” c. 5th century BCE.

² U.S. War Department. “The War of the Rebellion: A Compilation of the Official Records of the Union and Confederate Armies, Series I, Volume XXVII, Part 1.” 1889.



Tactics: behaviors that represent best practices

We'll start with the word tactics. In military parlance or otherwise, tactics are discrete behaviors that represent accepted best practices.

Here are a few examples of military tactics, so you can get a sense of what this word means today, which is also exactly what it meant all the way back in the 5th century BCE.

Flanking

When facing an enemy head-on, you're facing their strongest point. So you split your forces and send some of them around or behind the opposing force.

During the Battle of Gettysburg, Colonel Joshua Chamberlain and his men held the extreme left flank of the Union line from the top of a hill, running low on ammunition and about to be overrun.² He ordered his men to fix bayonets and execute a “right wheel forward,” holding the right side of his line in place while pivoting the left side down the hill, sweeping forward like a door swinging shut. That maneuver turned near defeat into a rout of the advancing Confederates.

Ambush

If you know an enemy will pass through a narrow, constrained place, you can wait until they approach and attack without warning.



During the Second Punic War, the Carthaginian general Hannibal hid his forces in the fog among the hills overlooking a narrow path around Lake Trasimene.³ He waited until the Roman army was pinned against the lake, then attacked, nearly wiping it out.

Siege

If you can contain an enemy to a closed area, cutting off supplies and reinforcements, the enemy must eventually break free or relent.

During World War II, German forces encircled the Soviet city of Leningrad for almost 900 days.⁴ The Russians eventually broke free, but not before roughly 1.5 million people died, mainly from starvation or the cold.

Cybersecurity tactics

Cybersecurity tactics work the same way. They're accepted best practices you're probably already familiar with:

- **Multi-factor authentication.** Require factors beyond a username and password, such as time-based one-time passwords (TOTPs), to increase confidence that the user is who they claim to be.
- **Microsegmentation.** Limit access to small segments of a network or environment so only a known, limited subset of authorized operations can occur, blocking everything else.
- **User and entity behavior analytics.** Monitor what a user or non-person entity does, baseline normal behavior, flag variations, and block access when activity may indicate an active compromise.

These all sound pretty good. You might be wondering what Sun Tzu was on about.

Why brilliant tactics can still lose the war

The fact is that tactics might win battles, but strategy is what wins wars.

Armies have practiced flanking, ambush, and siege since time immemorial, yet plenty of armies that executed them brilliantly went on to lose their wars, often because the tactics were used in ways that were poorly planned or not planned at all.

The siege of Leningrad, an example we referenced earlier in the chapter, was an outcome of Operation Barbarossa, the German invasion of the Soviet Union. It opened with another brilliant tactic: blitzkrieg or "lightning war."⁵ The goal was to strike hard and fast to overcome defenses by moving through or behind them quickly.

³ Livy. "The History of Rome, Book XXII." c. 27 BCE.

⁴ Harrison E. Salisbury. "The 900 Days: The Siege of Leningrad." 1969.

⁵ Karl-Heinz Frieser. "The Blitzkrieg Legend: The 1940 Campaign in the West." Naval Institute Press, 2005.

⁶ David M. Glantz. "The Siege of Leningrad, 1941–1944: 900 Days of Terror." Cassell, 2001.

⁷ David Stahel. "Operation Barbarossa and Germany's Defeat in the East." Cambridge University Press, 2009.

⁸ Martin van Creveld. "Supplying War: Logistics from Wallenstein to Patton." Cambridge University Press, 1977.

⁹ U.S. Department of Defense, Zero Trust Portfolio Management Office. "Zero Trust Execution Roadmap, v1.1." November 2024.

It's important to note that:

- Leningrad only became a siege because Barbarossa stalled out in the winter of 1941.⁶ The Germans had invaded with no winter gear, inadequate supplies, and no plan to counter what the Russians have always done when invaded: fall back as winter sets in.⁷
- Blitzkrieg compounded the problem, since the Germans had intentionally outrun their own supply lines.⁸ The result was a stalemate. The same tactics that worked brilliantly in continental Europe failed in Soviet Russia.

Defining strategy

A *strategy* is a high-level, credible plan to achieve goals and objectives. You might win a few battles through sound tactics, but winning a war takes a credible plan that maximizes advantages while working around disadvantages and constraints.

The accumulation of tactics isn't strategy. The clearest difference between the two shows up in how they're measured:

- A **tactic** is sound if it works reliably well.
- A **strategy** is successful only if it achieves the high-level goals and objectives it was designed for.

So, what does that look like in terms of Zero Trust?

The tactics and strategy behind Zero Trust

Let's start with tactics, because they're easier to describe. I've already mentioned a few tactics associated with Zero Trust.

Zero Trust tactics

The Department of Defense's (DoD) Zero Trust Capability Execution Roadmap, which I had a hand in creating, defines 45 capabilities, each a discrete tactic associated with Zero Trust.⁹ During my time leading Zero Trust implementation at the Department of Homeland Security, we developed a foundational Zero Trust architecture defining 46 capabilities, many overlapping with the DoW model.



The Office of Management and Budget (OMB) published a list of tasks for all federal civilian agencies in OMB Memorandum M-22-09, each of which can be read as a mandatory agency-wide Zero Trust tactic.¹⁰

Examples of tactics cited across all three sources (and probably many others) include:

- Implementing phishing-resistant multi-factor authentication (as previously described)
- Implementing attribute-based access control (ABAC)
- Isolating applications and environments with tools like microsegmentation
- Implementing endpoint detection and response (EDR) and continuously monitoring devices
- Creating and maintaining reliable and accurate asset inventories

These sources, and the ones they commonly cite, such as the CISA Zero Trust Maturity Model and NIST Special Publication 800-207, offer a wealth of guidance on what to do and how to do it.^{11,12} There's no shortage of tactical guidance in the federal government for Zero Trust.

The strategic gap

These sources, including those I contributed to, have a profound gap when it comes to strategy. They'll tell you a lot about what to do and how to do it, but not nearly enough about where to do it, or why to do it in one place or circumstance over another.

Specifically, they usually say nothing about where and how to sequence Zero Trust tactics, or they explicitly say to implement them everywhere, more or less immediately. Neither of those constitutes a strategy, because neither — implementing Zero Trust wherever or doing it everywhere all at once — is a credible high-level plan.

To fill that strategic gap, we need to go back to the very beginning of Zero Trust.

A Zero Trust view of the attack surface

Zero Trust was first defined in a 2010 paper by John Kindervag, *No More Chewy Centers: Introducing the Zero Trust Model of Information Security*.¹³ As the name implies, Kindervag compared the predominant approach to information security to a hard candy with a soft, chewy center: defense focused on hardening the network perimeter, what's known as the attack surface.

¹⁰ U.S. Office of Management and Budget. "Moving the U.S. Government Toward Zero Trust Cybersecurity Principles, M-22-09." January 2022.

¹¹ Cybersecurity and Infrastructure Security Agency. "Zero Trust Maturity Model, Version 2.0." April 2023.

¹² National Institute of Standards and Technology. "Zero Trust Architecture, NIST Special Publication 800-207." August 2020.

¹³ Forrester Research, John Kindervag. "No More Chewy Centers: Introducing the Zero Trust Model of Information Security." September 2010.

¹⁴ National Institute of Standards and Technology. "Security and Privacy Controls for Information Systems and Organizations, NIST Special Publication 800-53, Rev. 5." September 2020.

¹⁵ National Institute of Standards and Technology. "NIST Updates NVD Operations to Address Record CVE Growth." April 2026.

According to the NIST Computer Security Resource Center Glossary, the attack surface is "the set of points on the boundary of a system, a system component, or an environment where an attacker can try to enter, cause an effect on, or extract data from, that system, component, or environment."¹⁴ In cyber terms, the attack surface is the battlefield.

What the old model assumed

Prior to Zero Trust, defenders focused mainly on monitoring and securing the attack surface, assuming a few things now proven untrue:

- That bad actors come wholly or predominantly from outside the organization. In truth, they can come from anywhere.
- Users, once authenticated to the network, could safely be granted access to most information resources — or so the thinking went. Authentication confirms identity; it doesn't determine what that identity should be allowed to touch.
- The old model treated preventing every network breach as the defender's core job. That standard had already been shown impossible.

Why perfect defense isn't possible

That last point was Kindervag's central argument. It's not possible to prevent all breaches. If preventing all breaches were the true measure of defensive cyber operations, we should all just go home. That had been shown impossible even in 2010, and the pace and sophistication of cyberattacks have only grown since.

Some people disagreed in 2010, and maybe they had a leg to stand on for the next decade or so. Now, agentic AI and AI-assisted offensive and defensive cyber operations have changed that. Frontier models can discover and exploit vulnerabilities so quickly that the U.S. government no longer even tries to maintain a catalog of them.¹⁵

Any counterargument to Kindervag's point has lost all credibility. It was true back then, and it's inarguably true now.

Evolution of the Protect Surface

Kindervag's seminal work pointed out obvious flaws in the previous strategy, but it didn't immediately pose a replacement. Zero Trust was initially and fairly criticized as too complex, too theoretical, too costly, and too difficult to scale.



No one disagreed with the ideas themselves. Continuous authorization and aggressive, active monitoring made sense. However, implementing them was beyond the capabilities of most organizations and most technology at the time.

By the same definition of strategy — a credible, high-level plan to achieve goals and objectives — Zero Trust, at first, wasn't one.

From concept to strategy

Everything the U.S. government has published about Zero Trust came along about a decade after the concept was introduced, starting in late 2019. During that decade, private industry wrestled with, began to accept, and found ways to implement Zero Trust as a credible, actionable strategy.

It was during this time, in 2014, that Kindervag coined the term Protect Surface. Here's the key to understanding it. The attack surface describes everything. A Protect Surface describes one thing.

Specifically, a Protect Surface is one discrete set of data, assets, applications, or services, known as DAAS elements, that are protected by Zero Trust architecture. Instead of the acronym DAAS, I'll use the singular term "resources" here, since it captures all four at once and describes the objects Zero Trust architecture protects.

Protect Surfaces: patching boats in the harbor

Kindervag has his own analogy for Protect Surfaces, and he delivers it well. I'll leave that one to him. Here's mine:

Instead of systems or services, imagine that what you're managing are boats in a harbor. And instead of vulnerabilities, imagine that the boats have holes in them.

The boats are different in size and purpose. Some are yachts, some are battleships, and others are dinghies. The holes, and the number of holes, vary widely per ship.

If you apply every resource you have, you know you can patch roughly 60% of the holes in all of the boats. If you get extra people and extra resources, you might be able to patch 70-80% of the holes in all of the boats.

The tactic — patch holes — applies cleanly. But you need a strategy, and that strategy should be based not on the number of holes you can patch, working as fast as you can for as long as you can, but instead on the number of boats you might be able to save. Your choice of which holes to patch should be based on both how important a boat is and how hard it will be to save that boat before it sinks to the bottom of the harbor.

Protect Surfaces are boats in this analogy. Protect Surfaces describe which boats you can save and which ones are worth saving.

The alternative to Protect Surfaces is simply running around madly, patching every hole you can find, everywhere you can find one. And the result of that effort, no matter how hard or how fast you work, is a harbor full of sunken boats.

Identifying and operationalizing Protect Surfaces

All of that's easy to say and easy to understand. But what should go into a Protect Surface? How should organizations define them? And how does that turn Zero Trust from a pie-in-the-sky idea into a credible cybersecurity strategy?

Where to look for guidance

The two best references I can offer, ironically, come from sources that broadly disagree with each other: the international cybersecurity community and the U.S. government.

For international guidance, see the Cloud Security Alliance's guide, *Defining the Zero Trust Protect Surface*.¹⁶ For U.S. government guidance, read the final section of NIST SP 800-207 *Zero Trust Architecture*, specifically section 7.3. NIST never uses the term Protect Surface, and other federal references often disregard or contradict it, but the guidance here is functionally identical.¹⁷

Building the Protect Surface roadmap

In summary, both sources recommend the same general process.

Before you start, recognize that every organization will have many Protect Surfaces, and not every resource belongs in one.

Start by inventorying your organization's high-level business or mission processes, then list the resources, such as data, assets, applications, and services, involved in each. Even an incomplete pass at this exercise starts to reveal which processes matter most, which resources those processes depend on, and which resources come up again and again.

Then rank those resources two ways: by importance to protect, and by ease of protecting them. Resources used by one part of the organization rather than the whole, and resources you can already protect with capabilities you own, should rank first.

This process never really ends. As Zero Trust architecture coverage grows and new processes emerge, budget and plan for expanding it to new Protect Surfaces in a deliberate sequence.

¹⁶ Cloud Security Alliance. "Defining the Zero Trust Protect Surface." March 2024.

¹⁷ National Institute of Standards and Technology. "Zero Trust Architecture, NIST Special Publication 800-207." August 2020.



Followed this way, Zero Trust becomes a practical, incremental set of actions any organization can take. It's work that can be budgeted, planned, and paced to what the organization can actually support.

Why sequencing matters more than activity

The alternative — pursuing Zero Trust without any framing or pacing mechanism or trying to do it all at once — produces plenty of activity and cost but limited practical outcomes.

Consider the Federal Civilian executive branch's progress on one task in OMB M-22-09, implementing phishing-resistant multi-factor authentication (MFA), as seen in the graph below.

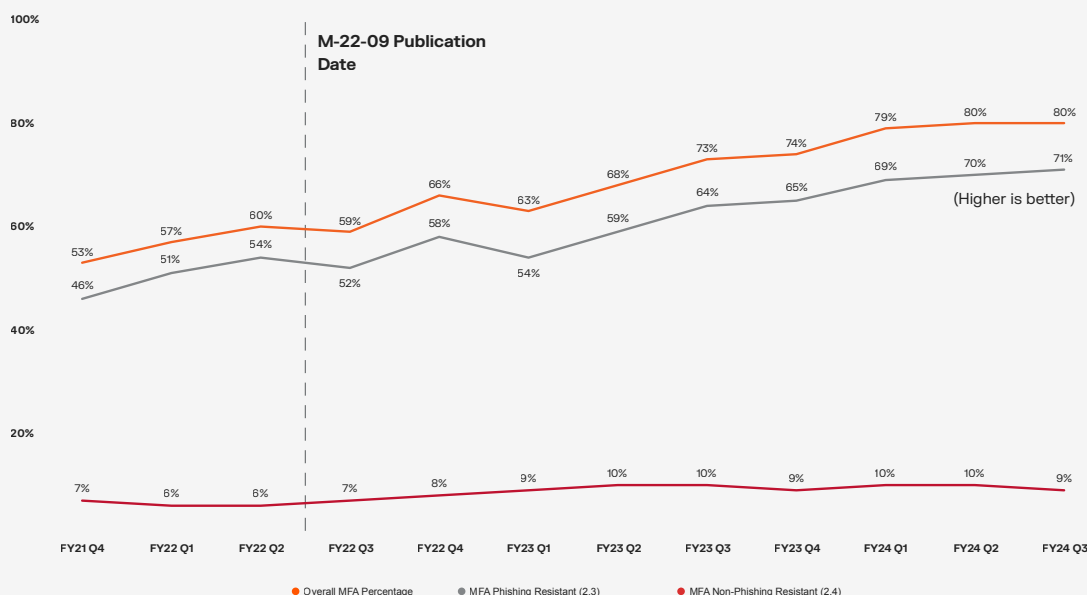


FIGURE 1: U.S. GOVERNMENT PROGRESS ON PHISHING-RESISTANT MFA, 2022-2024¹⁸

The task for public-facing agency systems that support MFA is that they must give users a phishing-resistant option within one year.¹⁹

When M-22-09 was published, roughly 60% of systems already complied. A year later, at the deadline, that number had reached about 70%. A year after that, at the end of the graph, it reached roughly 80%, while the share of systems using MFA that wasn't phishing-resistant crept up too, from about 7% to 9%.

What the numbers don't say matters more. After two years of government-wide effort on this one requirement, we can't say with any certainty that a single resource, at any agency, was better protected.

That matters for the American taxpayer, for your organization, and for you, if a government agency holds your data. In the harbor, this is the equivalent of patching 80% of the holes in 100% of the boats. The same effort, organized around Protect Surfaces, patches 100% of the holes in 80% of them — and those boats stay afloat.

Winning the cyber war

Phishing-resistant MFA is a tactic, and a sound one. But blindly, broadly applying sound tactics is not the same as having a strategy.

It is, as Sun Tzu put it, the noise before defeat.

If you want to implement Zero Trust, and make real, meaningful progress, you need a strategy: the careful, methodical, incremental implementation of Zero Trust architecture to Protect Surfaces.

That's a credible strategy and one that any organization would be wise to pursue.

¹⁸ Cybersecurity and Infrastructure Security Agency. "Zero Trust Architecture Implementation Fiscal Year 2024 Report to Congress." January 2025.

¹⁹ U.S. Office of Management and Budget. "Moving the U.S. Government Toward Zero Trust Cybersecurity Principles, M-22-09." Appendix B, pg. 28. January 2022.





STEP

2

Map the Transaction Flows

Step two of the Zero Trust model is where you get certainty about what's happening inside your environment.

Every security policy you write says the same thing about traffic: these connections serve the business, and they're the only ones permitted. If you fully understand your traffic first, you can make that claim with confidence. Resilience starts here, because a policy built on real traffic holds up when something goes wrong.

Teams almost always turn up something useful during this step. And AI now does it continuously, so your picture stays current as your environment changes.

Clint Bruce and I come at this step from two directions in the chapters ahead, one from the battlefield and one from the network.





CHAPTER 5

My Favorite Weapon Is a Map

Clint Bruce,

former U.S. Navy SEAL and founder and chairman of the HighGroundHQ Foundation

I met John Kindervag the way a lot of great working relationships start: he heard me say something that he couldn't stop thinking about.

It was at an event at my own office. I call our place HighGround — HighGroundHQ, officially — because I try to make a habit of living what I say out loud, not just saying it. That'll make more sense after the next few paragraphs.

John was in the room that day, and he heard me answer a question I get often, especially from young men: "What's your favorite gun?"

I've started pivoting off that question instead of just answering it. "Are you asking what my favorite gun is," I'll say, "or what my favorite weapon is?"

Because there's a difference. A gun is a tool, but a weapon is what I use to win.

My answer is that my favorite weapon is a map. The map tells me everything: where the bad guy is, how to get there, what to bring, and what to leave behind. When I have a map, the worst I'll ever be is wrong but never lost. And if you don't have a map, you're on the run, not on a mission.

John found me after and wanted to hear more. We ended up talking at length about the idea underneath that line — that a map isn't just a reference but a weapon. What he'd spent his career building, Zero Trust, wasn't a tool either, in the sense most people in his industry keep trying to sell it as. It was a weapon in the exact sense I meant it. It's a way to win, not just a thing to own.

He told me about a problem the cybersecurity industry couldn't shake: organizations kept getting hit, again and again, by people who'd already been let inside the perimeter. The bad guy wasn't breaking down the front door, because he was already in the building, wearing a badge somebody handed him, walking the halls like he belonged there.¹

¹ Forrester Research. "No More Chewy Centers: Introducing the Zero Trust Model of Information Security," September 2010.



I told him I didn't know the first thing about the cyber world. But I recognized instantly the shape of the problem he was describing. I just knew it by a different name.

I knew it as terrain, and I told him a little about how I try to think about it.

Maps matter

I've spent a career trying to live by the idea that maps matter — not the paper kind, but the mental kind. The kind I try to build before I ever step off the X, before the mission starts, and before the first shot gets fired.

To me, a map isn't just where things are but what I believe is true about the ground I'm standing on and about the ground I'm about to cross.

The best operators I ever served beside didn't move through a valley just because nobody had shot at them from the ridgeline yet. They moved as if somebody could be up there, every time, until they had a real reason to believe otherwise. I've come to think of it as practical rather than paranoid: assume the terrain has teeth until it proves it doesn't.

The fact is that the terrain doesn't care what business you're in. Whether you're talking about a valley or a network segment, both have choke points, avenues of approach, high ground and low ground, places you're covered, and places you're exposed.² What changes is the speed and scale of the attack.

And on every map I've ever built, there's always high ground somewhere on it — a place you either plan to take or plan to hold. The one on the high ground wins. Almost every fight I've ever studied, planned for, or been in comes down to that single fact eventually.

So when I look at my map, one of my first questions is "where's the high ground on this terrain, and am I taking it or holding it?" A weapon without that answer is still just a tool.

Four borders and an X: what makes up a map

For me, a map comes down to five things: four borders and an X.

The X is where I think I need to go, given everything I know right now. That qualifier matters to me — given everything I know right now — because I try to treat the X as a hypothesis instead of a fact.

I move until I can see over there. I find out the terrain feature I called "this" is actually "that." I'm humping a heavy ruck when contour lines I barely noticed on the map during planning suddenly matter more than anything else. Or the enemy has the nerve to move before I arrive, and I'm standing where intel said they'd be but no one's there.

² Headquarters, Department of the Army. "Intelligence Preparation of the Battlefield, ATP 2-01.3." March 2019.

So I try to hold the X loosely. If X never moves, I'm probably guessing confidently rather than gathering real intelligence.

I've always paid more attention to the borders on a map than to the X. The objective moves around, but the borders tend to stay put. That's why I spend more energy building them than chasing the X.

On a real map, borders are created from policy and deconfliction. Off the map, I've tried to build my life on moral, mental, physical, and spiritual borders.

A border, the way I use the word, is the line between "always" and "never." Most of a life gets spent moving things across it — taking something you were sure you'd always say or do or think, and setting it down on the other side, in "never again." Almost all of becoming a pro at any craft is that same process run at speed, stacking "never" and "never again" in a hurry.

Whatever's still standing on the "always" side is what we call the fundamentals, or what we'll call them soon enough.

So I'll ask you what I ask myself: what are your four borders, and how fixed are they really? Most of the trouble I've seen, in myself and in the leaders I work with, comes from getting that backwards: defending the plan harder than the values the plan was supposed to serve.

Two terrains of risk

There's one more layer I try to work through before I call a map finished. This is the actual road between the borders and the X.

I've found it helps to split terrain into two terrains:

- **Impact terrain** is the ones where I get a real vote. They're the things I can control, including what I train, who I bring, and how redundant my comms are.
- **Influence terrain** is the areas outside of my control where all I can do is get ready, because I can't move the probability or the scale. Here in North Texas, that's a spring tornado. I don't get a vote on whether one forms, where it touches down, or what it becomes on the F-scale. All I can do is build readiness, robustness, and resilience for something I can't out-work into submission.

I try to ask myself, on any risk in front of me, which terrain am I actually standing on? Because I've burned energy trying to "influence" things I was never going to move, and I've also coasted on things fully within my control, telling myself it was just fate.

I'd invite you to ask the same of whatever you're facing — which risks are yours to shrink, and which ones just need you ready.



The terrain you can't touch

Cyber terrain is faster than anything I ever crossed on foot, and bigger than anything I ever mapped with a grease pencil. It also carries an asymmetry the physical world rarely does.

One person, alone, can generate an effect that used to take a battalion.

But the logic of movement, I've come to believe, stays the same. An IP address is a location. There's a pathway from origin to target. And attackers still probe for the easy way into your environment. They'll prefer the path of least resistance, because attackers, human or digital, are opportunists first.³

Cyber terrain still has critical nodes, which are points of convergence where controlling one thing gives outsized control over everything flowing through it. That's high ground by another name.

I don't try to defend every square kilometer equally on the battlefield, and I don't think anyone should try to in a network either. Instead, I look for the high ground — the one node that decides who wins — and I settle early on whether my job is to take it or to hold it.

At the end of it, I keep coming back to the reality that somewhere there's a human being with a human motive, trying to turn a few keystrokes into something that happens in the physical world. This can either be a stolen payroll, a hospital system that goes down, or a pipeline that stops moving fuel.

Cybersecurity is terrain warfare, even if it's not always fought in the material world.

Why Zero Trust is the right doctrine

I don't pretend to speak with John's authority on cybersecurity. But what I can offer is the perspective of a man who tries to recognize good when he sees it.

I see Zero Trust as a doctrine, the set of principles that tells you how to think when the map is incomplete.

The two risk terrains help explain, at least to me, why it had to exist. Whether a specific attacker chooses you on a specific day — that's influence terrain. I don't get a vote on his intent, his sophistication, or his timing, any more than I get a vote on a supercell over Dallas.

But whether I'm ready for the moment he starts moving — whether I've built the robustness to absorb the hit instead of being erased by it — that's impact terrain, and I think Zero Trust lives entirely inside it.

I can't influence whether someone tries the door, but I can try to control whether trying it gets them anything.

Proximity isn't proof. Being inside the map tells me where someone is, but it tells me nothing about whether they belong there.

Cyber terrain works the same way. A user, device, or workload doesn't earn trust just by getting inside the network. I make each access decision from the facts on the ground at that moment.

In cyber terrain, the point of verification is high ground. An attacker takes that ground when he finds a way around the check. A defender holds it by demanding proof before each grant of access.

Zero Trust, as I understand it, is the doctrine of holding that ground all the time.

This is my four borders and my X, showing up again in a place I never expected to find them. "Never trust, always verify" is the border, and it stays fixed. Each access decision is the X, built fresh from the best information I have. The answer can change as the user, device, threat, and moment change, but the rule underneath it holds.

The map you build before you need it

John built Zero Trust without my map, and he'd have built the same doctrine if we'd never met. What I think my map analogy gave him was a bridge.

John works with people at every level of a company, from the board and C-suite to the teams making access decisions every day. They don't all speak the language of network segments, policy engines, and data flows. The map gives them a shared language. It lets John translate Zero Trust up, down, and across the organization so everyone can see the same terrain, understand what's at risk, and know the ground they're responsible for holding.

John already understood the terrain better than I ever will. What I hope the map did was let the people he most needed to reach see it fast, in language their own lives had already taught them to trust.

So here's what I'd leave you with. I try to build my map before I need it. I try to know my critical nodes before someone else finds them for me. But I'd ask you to consider your own version of that same discipline — your own borders, your own X, your own two terrains of risk — because somewhere, on some terrain you're responsible for, someone is deciding whether to move against you. The only real defense I've ever found is the one built before they did.

³ Verizon. "2026 Data Breach Investigations Report." May 2026.



Mapping Transaction Flows: You Can't Protect What You Can't See

John Kindervag,
Chief Evangelist at Illumio and creator of Zero Trust

One night, a technician walked into a server room and pulled a machine out of the rack. His reasoning? It looked old.

By the time the sun came up, he had taken down credit card processing for more than three thousand restaurants around the world.

I know that story well, because I was there. At the time, I was building one of the earliest Zero Trust environments I had ever attempted, for that same restaurant chain, and the goal was PCI compliance.

I had spent years as a qualified security assessor for the payment card industry. I was in something like the second PCI training class ever held, and I can tell you the process is exacting and unforgiving.

Of all the compliance mandates out there, PCI is the one that spells out in the most detail exactly what you have to do.¹ And the thing I had already worked out was that to build a compliant cardholder data environment, I had to segment it away from everything else.

The networks I kept walking into were flat and wide open on the inside. Anything which gained a foothold could wander straight to the card data with nothing in its way.

Segmentation was the fix, and segmentation only works when you know what connects to what. So that is what we were doing. We were building a Zero Trust network, and we were building it to pass a PCI assessment.

1 PCI Security Standards Council. "Payment Card Industry Data Security Standard, v2.0." October 2010.



When the finger-pointing started the next morning, the technician pointed at me. It was the Zero Trust project's fault, he said.

I asked him how he figured that. Well, he explained, if the project hadn't been underway, he wouldn't have been anywhere near the data center in the first place.

Thankfully, the CIO saw through the excuse in about five seconds. That server had nothing to do with my project. The man had pulled it because it looked old, and he had no idea what it actually did.

That last part is what changed how I think about security. The server he yanked was a critical link in the credit card processing chain. Take it out, and every transaction stopped.



So I started asking around. Did anyone know that unplugging that one machine would break payment processing for the whole company? Nobody knew.

Then I asked a bigger question. Does anyone here understand how our credit card processing actually works, from one end to the other? Again, nobody knew.

That's a frightening thing to learn about a business that runs millions of dollars in card transactions a day.

The knowledge of how it all fit together didn't live in a document, or a diagram, or even in one person's head. It didn't live anywhere at all. Each person had only ever understood their own piece.

Nobody has the map

That gap was not unique to that company. It was true of nearly every environment I walked into. Almost no one understood how their own systems worked as they were actually built.

Sometimes there would be a Visio diagram from the early days, drawn when the network was first stood up. But the network had changed a hundred times since then, and the diagram no longer looked anything like reality.

People were defending environments they couldn't even describe.

The dangerous thing about an out-of-date map is that it lies to you with confidence. It shows you an environment that looks one way while the traffic quietly moves another.

You make decisions against a picture that stopped being true years ago. You never find out it was wrong until something breaks, usually at the worst possible hour of the night.

Zero Trust begins from a humbler place than that. You can't protect what you can't see, and you can't honestly claim to see an environment you are unable to draw.

So I started mapping the flows. I needed to see how the traffic actually moved, so that I would have a map based on what the network was doing right now, not on what someone had sketched years earlier.

As Clint Bruce puts it in his chapter, a map is a weapon. I didn't fully appreciate that yet, but I already knew I couldn't work without one.

What a flow actually is

Let me back up and define what I mean by traffic flow, because a lot of people in security today have never had to.

In computer networking, a flow is a sequence of related packets traveling from a source to a destination. It's one of the foundational ideas in how networks operate, and it underpins nearly everything, from traffic management and routing to performance and security.

Everything that happens in IT, down at the level where the packets move, is built on flows.

We've lost sight of that as an industry. A great many talented people have come into security without a networking background, and so much of the modern stack, cloud especially, is designed to hide the plumbing from you.

You aren't supposed to have to know how it works. For building and shipping applications quickly, that abstraction is a gift.

But Zero Trust asks the opposite of you. In Zero Trust, you have to understand how things work. The single most important question you can ask is how the system behaves as a system, and mapping your transaction flows is how you answer it.

That's also why mapping sits where it does in the methodology. In the five steps of Zero Trust, you first define your Protect Surface, the specific thing you are protecting. Then, before you build a single control, you map the flows around it.

You can't design protection for something when you don't know how it communicates.

Making the map

In the early days, mapping traffic flows was a manual and painstaking business.

You made your best approximation of a map by hand. You'd sit down and interview everyone connected to the system, the application owners, the network people, anyone who so much as touched it. Then, you'd assemble the picture from what they told you.

It was slow and imperfect, because memory is imperfect and no single person ever held the whole story. You were stitching together fragments and hoping the seams roughly lined up.

And it was still one of the most useful things I had ever made. Once I could see the flows laid out in front of me, the design work almost fell into place.

The map showed me exactly where I needed to put a control, and then where I needed another. I saw where I needed to draw a boundary around a set of sensitive information, which is what we would now call a Protect Surface.

It did something else I didn't expect, too. It surfaced dependencies that nobody in the building knew existed, like those that let a technician knock out payment processing for three thousand restaurants.

When you can see every connection running to and from the thing you're protecting, those kinds of landmines stop being invisible. You can see them on paper and plan around them, rather than discovering them at two in the morning in the middle of an outage.



Seeing it was the whole game. Without that picture, I honestly don't know how anyone designs a network. You'd be guessing.

You might as well be throwing darts at a dartboard blindfolded, after somebody had spun you in a circle ten times. You would have no real idea what you were hitting or why.

The tools caught up

As time went on, I started pushing for better ways to map traffic flows. Doing it by hand simply didn't scale, and worse, a hand-drawn map was out of date the moment I finished it.

I'd spend weeks interviewing people and piecing the network picture together. By the time it was done, the environment had already shifted underneath me.

A map you can't keep current is just a snapshot of a moment that has already passed.

So I knew what I needed from technology before the technology existed. It had to do three things:

- See everything across the environment instead of just one slice of it
- Stay current on its own without an army of people redrawing it by hand
- Let me turn what I was seeing into policy

The tooling took a while to catch up with that need. The early products all had the same flaw. Each vendor's tool could only see its own equipment or its own environment, never the whole network. This means you'd get perfect visibility inside one platform but then go completely blind the moment traffic crossed over to anything else.

That kind of partial map is more dangerous than it looks. It's better than nothing, but the risk hides precisely in the parts it can't show you.

You start trusting the picture, you make decisions against it, and the flow that takes you down is the one that was never on the map to begin with. Guessing was exactly what I was trying to get away from, and a map with holes in it is just guessing in a nicer suit.

The mapping capability is one of the biggest reasons I ended up at Illumio. I need the map in order to build the environment. I can't architect anything without one, and I'll never fall back on guesswork.

The map is the thing that shows you how to win.

The guidance agrees with me

For a long time, flow mapping was something I had to argue for. People treated it as extra work, a nice-to-have you'd get to once the "real" security was in place.

That's changed, and the importance of mapping isn't just my personal conviction anymore.

When the NSA published its Zero Trust Implementation Guideline in January 2026, it put real weight behind flow mapping, with an entire section devoted to data flow mapping.² They say security teams should manage, monitor, and restrict both internal and external traffic flows.

Internal is the part most people skip. For decades, we poured our attention into the perimeter, watching the traffic coming in from the outside while the traffic moving laterally on the inside went unwatched.

That internal, east-west movement is exactly how an attacker gets from the machine they first compromised to the data they actually came for. The guidance is correctly telling you to map and control that movement.

The NSA isn't alone in this. The NSTAC Report on Zero Trust and Trusted Identity Management digs into flows as well, right alongside the five-step process and the maturity model.³ I had a hand in that report, so I can tell you the emphasis on flows was deliberate, not an afterthought.

When the people writing guidance for the nation's most sensitive systems are telling you to map your flows, it's probably worth doing.

² National Security Agency, "Zero Trust Implementation Guideline: Discovery Phase, Version 1.0." January 2026.

³ President's National Security Telecommunications Advisory Committee. "Report to the President on Zero Trust and Trusted Identity Management." February 2022.



Why it all comes back to flows

All networks are flow-based. That means cybersecurity is flow-based, and Zero Trust is flow-based, right down to the foundation.

When you look at one of these maps, or at what we would now call a security graph, every line connecting two points is a flow. And policy is what you write against those flows.

That's the part people miss. You can't write good policy for a flow you don't understand. Yet, security teams write policy against things all the time without knowing how those things actually talk to one another.

Strip a policy down to its bones, and it's really just a short list of values: source, destination, source port, destination port, and protocol. In practice, I would argue only about four of them matter, because I have almost never seen a real flow that needed different source and destination ports to make sense of it.

Simple as that is, we somehow abstracted it into something we stopped understanding. It became long, flat lists of rules with no picture of how any of it connected.

When your security lives in a list, you're managing thousands of rules with no understanding of how they relate. You tend to leave things open because you're afraid of breaking something you can't see.

That fear is exactly how bad things get through, because all bad things happen inside of an allow rule. A graph gives you the confidence to tighten those rules, because at last you can see what each one is actually doing.

There is a well-known observation from Microsoft's security team that captures the problem exactly: defenders think in lists, and attackers think in graphs.⁴

An attacker doesn't look at your environment as a spreadsheet of assets. They see a web of connections, and they follow those connections straight to their target.

A map turns your list back into a graph. It shows you the nodes, the edges between them, and the policy riding on each edge.

The reason we go to all this trouble with maps is so we can write policy that's effective, restrictive, and still usable. It has to be tight enough to stop the bad thing without being so blind that it breaks the business.

A good map is what makes all that possible.



⁴ John Lambert, Microsoft. "Defenders Think in Lists. Attackers Think in Graphs. As Long as This Is True, Attackers Win." April 2015.





STEP

3

Architect the Network

Step three is where your Zero Trust strategy takes physical shape. You've named what's most critical in your network and how it communicates, so now you're ready to build the architecture that protects it.

Segmentation is foundational to your Zero Trust architecture. It answers the problem that has wrecked organizations for my whole career: an attacker gets in somewhere small and ends up everywhere. In a segmented environment, a breach stays contained to where it entered rather than spreading through your network.

That's breach containment, and it matters even more now that AI agents both run inside our networks and give attackers endless opportunities to build threats. Jason Garbis, Raghu Nandakumara, Josh Woodruff, and Michelle Savage take on this step in this section.





CHAPTER 7

Zero Trust Architecture Must Be Holistic and Capability-Driven

Jason Garbis,
founder and Principal at Numberline Security

The chapters before this one asked you to dive into your enterprise architecture. First, you defined your Protect Surfaces, then you mapped the transaction flows around them. This chapter asks you to step back and take in the whole enterprise.

This is the point in the journey where the separate pieces have to start working as one.

Our premise is simple to state but often demanding to execute. Every successful Zero Trust architecture is in fact a system, made up of information sources, policy enforcement points (PEPs), and access policies that span all of the pillars, and dynamically interact with one another. And you can't design a system until you know your security capabilities across technology, process, and people.

The concept of a security capability is central to our thesis in this chapter. But before we fully explore that idea, let's look at how one enterprise approached things.

An enterprise learns the hard way

Consider an enterprise partway down a Zero Trust path.

This global financial services firm of around 6,000 employees was nearly 100 years old and had been using IT to run their business for over 50 years. Their environment was anchored by next-generation firewalls and run on traditional, coarse-grained network segmentation on a flat enterprise network.

To their credit, they had defined access policies as best they could with what they had. But they hit a ceiling. The firewall rulesets grew steadily in complexity and operational overhead, while the policies themselves remained frustratingly shallow.



They wanted richer, more dynamic policies, ones that considered who was connecting, from what device, to which workload, but their architecture and enforcement points simply couldn't express them. Their policy ambition had gotten ahead of their capability.

While technology was the most obvious constraint, they recognized that this was a shortfall across people, process, and technology, not just a limit of the product.

When they began their enterprise Zero Trust initiative in earnest, one of their first steps was to procure and deploy a Zero Trust network access (ZTNA) solution. This gave them new enforcement capabilities, which meant that better and more effective policies became possible for their chosen Protect Surfaces.

Perhaps most importantly, these access policies were inherently automated, updating based on user, workload, device, and system context. The results were a significant reduction in manual firewall ruleset workload, and the ability to bring device posture, identity, and workload attributes together into a single access policy for the first time.

Their next steps built on this foundation. They began improving their process for categorizing data, licensed a data loss prevention (DLP) module within their ZTNA platform to consume that categorization, and put both to work in new access policies. This is a good example of how a process improvement like data categorization and a technology addition like DLP combined to unlock a class of policy they could not previously enforce.

This enterprise took away three lessons worth carrying through the rest of this chapter:

- **Improve capabilities before enabling the policies that require them.** This enterprise had struggled because its policy ambition exceeded its enforcement capability. Progress came when they closed that gap deliberately in the right order.
- **Capabilities are people, process, and technology, not just technology.** The ZTNA rollout succeeded because it included operational process changes (defining who owns policy changes) and investment in people (building the skills to run the platform), alongside the new product.
- **New and improved capabilities open the door to new classes of policy.** Their selection of DLP was a deliberate step up a capability ladder. It was mapped to the policy outcomes they wanted next.

Know your toolbox before you draw the blueprint

Zero Trust architectures can't be built in isolation. Step 3 of the Zero Trust model is the bridge between planning (defining Protect Surfaces and mapping transaction flows) and doing (creating and enforcing policy).

At this point in the journey, you know what to protect. But you can't decide how to protect it until you know what's in your toolbox and how the tools work together. Understanding your security *capabilities* and your limitations across the entirety of your enterprise's IT environment is the core of this step.

Consider the analogy of constructing a building. You can't architect a building until you understand both what you need to build and what tools, skills, and materials you have. Are you building a doghouse? A single-family home? A skyscraper? What materials do you have on hand, and what do you need to purchase? Can you safely wield a hammer? What about an arc welder, or a 200-foot tower crane?

These are direct analogs to Zero Trust architecture planning. The Protect Surface's transaction flows tell you what you're building, and your capabilities determine what you can actually construct, and when.

A necessary first step, then, is to get a clear picture of your security capabilities from technology, process, and people perspectives. This picture allows you to do two things:

- You can start defining and enforcing access policies based on your current capabilities, delivering value now rather than waiting for a perfect end state.
- You can start planning a roadmap for how you'll add to and improve those capabilities, so that future access policies can take advantage of them.

The result is a concrete view of what your policy enforcement points can enforce today, plus a detailed roadmap for the access policies you want to define next.

Let's be more precise about the three types of capabilities:

- **Technology** is the usual focus, and it's necessary but not sufficient. This is your inventory of enforcement points and information sources: firewalls, microsegmentation solutions, ZTNA platforms, identity providers, endpoint management, DLP, and the rest.
- **Process** covers operations and governance, including who approves policy changes, how exceptions are handled, and the "secret sauce" for Zero Trust initiatives, which is well-defined governance processes. In an AI context, ask whether your access lifecycle processes even have pathways for non-human identities, or whether agents simply inherit user or service accounts without oversight.
- **People** means skills and time allocation. A capable product with no one trained or available to operate it isn't, in any meaningful sense, a capability you possess. Increasingly, this includes newer skills, such as the ability to evaluate agent gateways and reason about machine-to-machine authorization.



Two final points:

- **Zero Trust architectures need to be holistic.** Your Protect Surfaces and their transaction flows will only connect to effective access policies if the architecture treats identity, network, device, workload, and data controls as one interacting system. A best-of-breed pile of point solutions with no shared signals and no coordinated enforcement isn't a Zero Trust architecture but a shopping list.
- **AI raises the stakes.** The Protect Surface now includes models, training data, and agents. The actor population now includes non-human identities operating at machine speed. An architecture built of disconnected point solutions was merely inefficient in the pre-AI world. Now it's untenable.

Closing the gap in the right order

So the work from here breaks into three moves, and the order matters. Take stock of what you have, design the system you want, and then schedule the trip between them.

Understand your current capabilities

Start by understanding your enterprise architecture and your current set of capabilities. This tells you what types of PEPs you already have in place and how you can begin using them in access policies today.

For example, if you want to begin enforcing the use of privileged session management for administrative access, you'll need to either have a privileged session management (PSM) solution in place or procure one. There's no policy language clever enough to enforce a control that doesn't exist.

A second, more forward-looking example is enforcing agent-to-agent access. Here, your capability options span a spectrum:

- Direct API calls leave authorization decisions embedded in each endpoint.
- The model context protocol (MCP) structures how agents reach tools and data, but authorization typically remains local to each server.¹
- Agent-to-agent (A2A) protocols enable richer interactions between agents, and an A2A deployment fronted by a gateway lets you externalize authorization into a policy decision point you control.²

Each option represents a different capability level, and each supports a different class of enforceable policy.

To decide which matter most, start from controls you're already tracking. For example, start with your compliance reporting processes, or extrapolate from a canonical source such as the CISA Zero Trust Maturity Model (ZTMM). The ZTMM's maturity pathways make capability progression concrete. In the Identity pillar, for instance, the authentication function progresses from passwords to multi-factor authentication (MFA), to phishing-resistant MFA, and ultimately toward continuous, risk-based validation.

Each stage on that pathway is a capability that unlocks stronger access policies than the one before it.

Define your to-be architecture as a system

Next, define your to-be architecture, keeping in mind that because Zero Trust is a holistic strategy, your architecture, and the access policies built on top of it, will be holistic as well.

By necessity, these will span the Zero Trust pillars. Your architecture will have elements across identity, networks, devices, applications and workloads, and data. These elements will be combinations of information sources and enforcement points that share signals with one another.

Critically, this to-be state is an evolving set of architectural improvements that include changes to technology, process, and people.

Build a capability roadmap

Finally, put together a capability roadmap that shows when and how you're going to add these capabilities.

As each capability comes online, you'll be able to build out the corresponding piece of your Zero Trust architecture. This in turn gives you a time-based foundation for your access policy planning, including what you can enforce today, next quarter, and next year. Include process and people milestones, such as training programs, governance changes, and ownership assignments, alongside the technology deployments.

A roadmap that only tracks product rollouts will repeat the mistake our example enterprise made, just on a longer timeline.

A sample capability roadmap

A capability roadmap is based on several connected pieces. We need to have a clear picture of what the capabilities in our environment are and how ready we are to use them.

It's important to understand that there's a progression to this. Some security capabilities won't exist in the environment and therefore can't be used. Others will be partially functional — ready to use but in need of improvement. And still others will be in place and ready to utilize.

¹ Anthropic. "Model Context Protocol Specification: Authorization, Revision 2025-06-18." June 2025.

² Linux Foundation Agent2Agent Project. "Agent2Agent (A2A) Protocol Specification, v1.0." April 2026.



We like to keep things simple, so we recommend classifying these as Red, Yellow, and Green.

The table below shows a small example subset of the capabilities you might track based on the CISA Zero Trust Maturity Model.³ For each, our sample enterprise has indicated its current readiness state, and its *roadmap tasks* planned to improve it.

Note that even green capabilities can have planned roadmap tasks. After all, everything can be improved.

Sample capability readiness assessment

Function	Readiness	Roadmap task
Identity		
Cloud-based user authentication	Green	
Phishing-resistant MFA enforcement	Yellow	Complete pilot project and roll out to full user population
Identity group membership health (for use in access policies)	Green	Continue Identity Governance process improvements
...		
Devices		
Device certificate validation (all devices)	Green	
Device posture checks (Windows)	Green	
Device posture checks (macOS)	Red	Deploy macOS device posture report integration, validate data flow
...		
Networks		
Unencrypted protocol detection and blocking	Green	
East-west access controls	Yellow	Complete pilot of microsegmentation solution, deploy broadly
IT admin access controlled by Service Desk ticket assignment	Green	
...		
Applications and workloads		
Workload identity validation and enforcement	Red	Productize the POC with SPIFFE Pilot SPIFFE with workloads
Asset inventory completeness and correctness	Yellow	Iterate on workload deployment and governance processes
...		
Data		
Structured application data classification	Green	
Unstructured data classification (at creation time)	Yellow	Validate scheme and workflow with end users
Unstructured data classification (existing data)	Yellow	Validate AI-based data analysis quality
...		

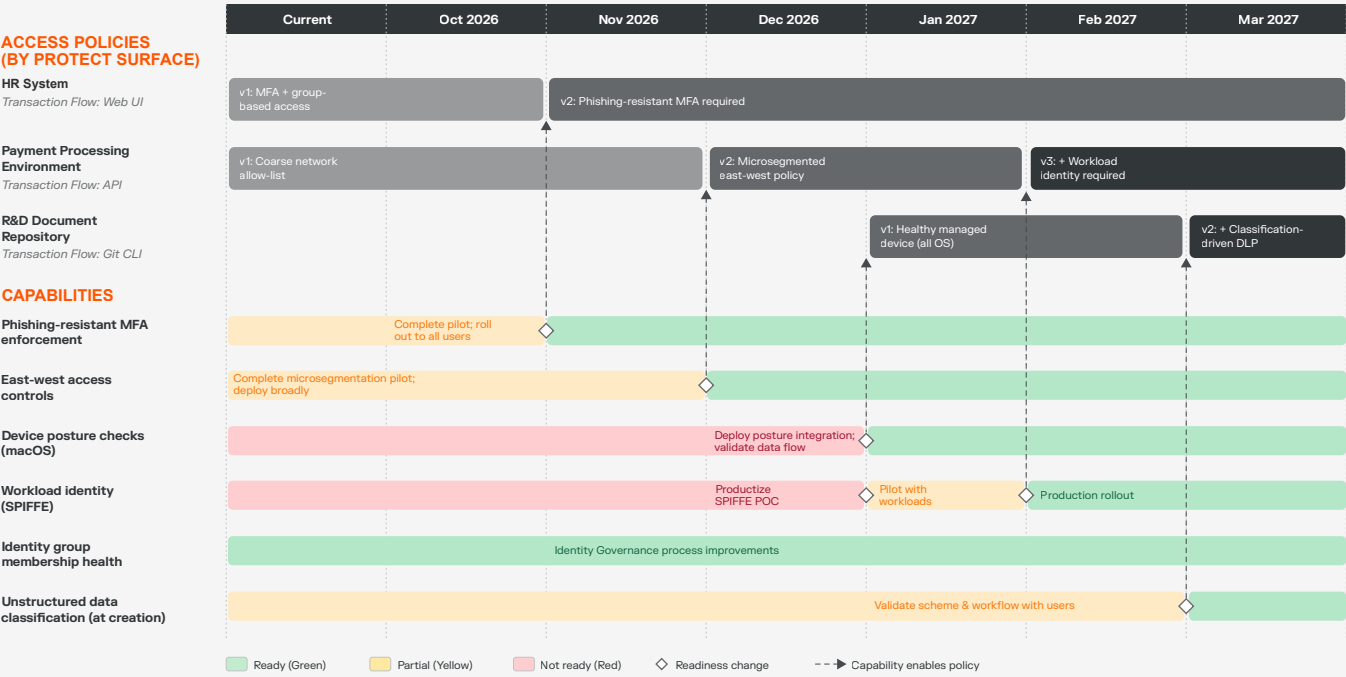
³ Cybersecurity and Infrastructure Security Agency. "Zero Trust Maturity Model, Version 2.0." April 2023.



The second part of our roadmap is, well, the actual roadmap. The tasks above need to be shown on a timeline, so our team can see and plan for their execution. And it's important to be able to model and see the dependencies between capability readiness and the access policies for each transaction flow.

The image below shows part of our sample roadmap showing how our capability readiness will improve over time as we work through our planned roadmap tasks. And it also shows how these readiness improvements will “unlock” new and more effective access policies for specific Protect Surfaces and transaction flows.

Sample Capability & Access Policy Roadmap



Above is a sample capability roadmap, with capabilities on the bottom and the access policies they enable on top. Let's examine the payment processing Protect Surface to illustrate our capability-driven approach.

Today, its API transaction flows are protected by a coarse network allowlist, because that's all the current enforcement points can express. When the east-west access controls reach production in November, a microsegmented policy becomes enforceable, and the policy is upgraded. When workload identity completes its production rollout in January, the policy tightens again: access now requires not just an approved network path but a cryptographically verified workload identity.

Each policy improvement is unlocked by and deliberately scheduled behind a capability improvement. This is the discipline that keeps policy ambition and enforcement capability in lockstep, and it's why the roadmap, not the architecture diagram, is the artifact your Zero Trust program will live by.



Turning your Zero Trust architecture into a workable plan

Your Zero Trust architecture is the system that puts the Zero Trust model's Steps 1 and 2 to work and makes Step 4 possible.

Three moves get you there:

- **Size up your current capabilities against your Protect Surfaces**, so you know what your existing PEPs can enforce today.
- **Define the to-be architecture as one system**, where information sources and enforcement points share signals across identity, network, device, workload, and data.
- **Build a capability roadmap** that carries people and process milestones right next to the technology, because each capability that comes online widens what your Step 4 policies can say.

Thinking in whole systems is what separates a real, effective architecture from a shopping list. Know your toolbox of technology, process, and people before you design, and let the roadmap turn your architecture from a snapshot into a plan.

With your capabilities mapped and your architecture defined, you're ready for the next step: creating policy. Later chapters in this book carry this same discipline to the newest actors in your enterprise: your AI agents.



CHAPTER 8

The Segmentation Imperative

Raghu Nandakumara

Vice President of Industry
Solutions Marketing at Illumio

In 1423, Venice built a hospital on an island in its lagoon and began holding arriving ships there for 40 days before anyone aboard could set foot in the city.¹

The Venetians had no theory of contagion. They had no way to inspect a vessel and know whether plague was aboard it. They also couldn't stop infected ships from reaching their harbor.

Venice gave up on controlling arrival and started controlling movement. The word we inherited from them is quarantine, from quaranta giorni, and it was the best way to hold back the bubonic plague from ravaging the city.

Controlling movement instead of arrival is the whole argument of this chapter, and in the cybersecurity industry six centuries later, we've largely abandoned it.

Venice inspected ships at the harbor mouth, but the real control began after they docked: quarantine zones, escorted movement, and limits on where a crew could go. Modern cybersecurity kept the first half and dropped the second. Firewalls, intrusion detection and prevention systems (IDS/IPS), and email gateways all sit at the network edge, deciding which packets may enter and which look wrong.

Almost none of that tech follows traffic once it's inside the network, which is the question Venice actually answered: how far can a crew wander once the ship has docked?

Today, we call that lateral movement. It's the part of an attack we have the most power to contain, since it happens entirely on infrastructure we control. It's also the part most environments leave wide open, since traffic that clears the perimeter is treated as trusted from then on.

Microsegmentation is the answer to this problem, and it's critical in today's complex environments.



Segmentation has always been part of Zero Trust architectures. John Kindervag's second report on Zero Trust specifically called out segmentation as a key function in building these types of environments: "Therefore, new ways of segmenting networks must be created because all future networks need to be segmented by default." Sadly, it's taken 16 years for the world to catch up.

We keep agreeing with the argument but never delivering it

I spend most of my working life talking to organizations about segmentation. Almost nobody disagrees with the case for it. Boards nod, architects nod, and then the program becomes a pilot, the pilot becomes a proof of concept, and the proof of concept becomes a line in next year's roadmap.

¹ Eugenia Tognotti. "Lessons from the History of Quarantine, from Plague to Influenza A." Emerging Infectious Diseases, Centers for Disease Control and Prevention.



I could name the budget categories that got funded instead, but they'd be stale before this book reached a second printing and naming them misses the point.

The pattern is older than any of them. Teams have deferred segmentation through the mainframe era, again through client-server, again through virtualization, and again through the move to cloud.

While there's different technology each time, there's identical reasoning every time. The new platform is being adopted at speed, segmentation would slow that down, and the boundaries can be retrofitted once the migration settles.

The migration never settles, though. There's always another platform, and the retrofit is never cheaper later than it would've been at the start.

This is what happens when fundamentals lose to fashionable technology. It's a structural issue rather than a failure of judgment.

Consider what each security control looks like in the twenty minutes a security leader gets in front of a board. Detection produces a dashboard, including incidents raised, mean time to respond, and a line moving up and to the right. Identity produces a percentage that climbs each quarter.

But segmentation produces an absence. The incident stayed small, the outage didn't happen, and the quarter in which nothing was reported because nothing spread.

Nobody has worked out how to put an absence on a slide, and what can't be shown doesn't get renewed. So segmentation waits, and it's been waiting for 40 years.

Meanwhile, the arithmetic of breaches has moved decisively against us. Verizon's 2026 Data Breach Investigations Report, covering more than 22,000 confirmed breaches across 145 countries, found that attackers exploitation of software vulnerabilities has overtaken credential abuse as the leading route to initial access, rising from 20% to 31% of intrusions.² In the same dataset, only 26% of known exploited vulnerabilities were fully remediated, down from 38%.

The data shows that the most common way attackers get inside environments is the one we're getting worse at closing. It's because the volume of threats has outrun what any patching program can absorb.

This raises the question I would put to every security leader reading this. You can't close every route into your network. So, what's your plan after one of those routes gets compromised and the attacker gets in?

How segmentation could've contained five major breaches

The incidents below span 10 years, four countries, and five sectors, each documented in a detailed public review.

Look at how the attacks started and then what happened next. In every case, investigators discovered that the initial compromised workload could then access systems it should never have been able to. Proper microsegmentation could've kept each incident from turning into a major breach.

Target, 2013

Attackers phished a refrigeration contractor, used its credentials to reach a vendor-facing portal, and from there moved into the systems processing payment cards.

They exfiltrated more than 40 million card records, along with personal details on tens of millions more. The staff report prepared for the U.S. Senate Committee on Commerce, Science, and Transportation concluded that the retailer had failed to isolate its most sensitive assets from the rest of its network.³

This gets misread as a supplier due-diligence story, but it isn't. A billing portal for a refrigeration contractor and a payment processing environment should never have been mutually reachable, however well that contractor ran its own security.

And isolating the cardholder data environment was no emerging idea in 2013. It was a documented requirement of the standards Target was assessed against.⁴

Segmentation failures are usually failures of implementation depth. The gap is between an environment declared segmented and one that behaves that way under an adversary.

This is what happens when fundamentals lose to fashionable technology. It's a structural issue rather than a failure of judgment.

² Verizon. "2026 Data Breach Investigations Report." May 2026.

³ U.S. Senate Committee on Commerce, Science, and Transportation. "A 'Kill Chain' Analysis of the 2013 Target Data Breach." March 2014.

⁴ PCI Security Standards Council. "Payment Card Industry Data Security Standard, v2.0." October 2010.



The Ukrainian distribution grid, 2015

The joint E-ISAC and SANS analysis of the December 2015 attacks on three regional electricity distributors in Ukraine describes the unusual fact that the environment was segmented, but it failed anyway.⁵

Boundaries existed between the business networks and the industrial control systems. What defeated them was a single trusted crossing.

Remote access into the control environment appeared to have lacked two-factor authentication (2FA), and firewall policy was permissive enough to let the adversary administer systems from outside. The attackers persisted for six months or more, then opened breakers at around 30 substations and cut power to roughly 250,000 people.

The same incident makes the opposite case just as clearly. Operators restored supply by walking to substations and closing breakers by hand, because the physical layer wasn't wholly dependent on the digital one.

Separation is what put the lights back on. Segmentation is not only how you contain an attack but how you retain a way to operate during one.

Colonial Pipeline, 2021

Ransomware entered Colonial's corporate environment through a legacy remote access account with no multi-factor authentication (MFA).

There's no public evidence the attackers ever reached the operational technology running the pipeline. But Colonial couldn't prove that at the time. So the company shut down 5,500 miles of pipe carrying close to half the fuel on the U.S. East Coast, a decision its chief executive later described in congressional testimony as taken out of an abundance of caution.⁶

Faced with an intrusion it couldn't scope, Colonial had to assume the worst. But the assumption cost days of regional fuel shortages, emergency federal measures, and panic buying across seventeen states.

All of it followed from a question the company couldn't answer quickly enough: had the attackers crossed from the business network into the operational one?

Segmentation you can't demonstrate under pressure gives you nothing at the moment you need it most. If you can't prove separation, you have to assume you don't have it and act accordingly.

5 E-ISAC and SANS ICS. "Analysis of the Cyber Attack on the Ukrainian Power Grid: Defense Use Case." March 2016.

6 Joseph Blount, President and CEO, Colonial Pipeline Company. Testimony before the U.S. Senate Committee on Homeland Security and Governmental Affairs, June 2021, and the U.S. House Committee on Homeland Security, June 2021.

7 PwC. "Conti Cyber Attack on the HSE: Independent Post Incident Review." December 2021.

8 Ibid.

9 The British Library. "Learning Lessons from the Cyber-Attack: British Library Cyber Incident Review." March 2024.

Ireland's Health Service Executive, 2021

A phishing email in March 2021 led to ransomware detonating across Ireland's national health service eight weeks later.

Hospitals reverted to pen and paper. Full recovery took roughly four months.

The independent post-incident review commissioned by the Health Service Executive (HSE) board described the National Healthcare Network as "primarily an unsegmented (or undivided) network," and identified that topology, combined with an unmapped set of admin permissions, as what let one compromise reach systems belonging to many separate organizations.⁷

Then the response team discovered that antivirus tooling had flagged an intrusion six weeks before the attack. A hospital detected and responded to malicious activity four days before. The national cyber authority raised concerns the day before. None of it had converted into containment.

Detection buys you time. Only segmentation turns that time into anything useful, by bounding the ground an attacker can cover while the alert sits in someone's queue.

Detection without containment is a smoke alarm in a building with no fire doors.

The British Library, 2023

In March 2024, the British Library published its own review of the Rhysida ransomware attack the year prior.

Attackers gained entry likely through a terminal server set up to give external partners and admins efficient access. It was deliberately left outside the multi-factor authentication rollout on grounds of cost and practicality.

The attackers took around 600GB of data, just under half a million documents, hijacked native administrative tooling to copy 22 databases, then destroyed much of the server environment.⁸

The Library attributes the severity to the shape of its network, concluding that its historically complex topology gave the attackers "wider access than would have been possible in a more modern network design." Segmentation appears explicitly in its published lessons.⁹

But the first lesson in that list was that legacy topology prevented modern security tooling from achieving full coverage. The Library had current tools, but its network stopped them working.



The breach proved that segmentation isn't an alternative to your detection investment but rather the precondition for it.

What it costs and who profits

The IBM Cost of a Data Breach Report 2026 puts the global average cost of a data breach at \$4.99 million, the highest the study has recorded.¹⁰

Costs are concentrated in detection, escalation, and lost business from operational disruption. The bill is driven by how far an incident spreads, not by the fact that it happened.

Cost tracks with spread, spread tracks with reachability, and reachability is a property of architecture.

There's a commercial reading I find more persuasive than any cost figure.

In the Verizon 2026 Data Breach Investigations Report, ransomware appeared in 48% of breaches, while 69% of victims declined to pay and the median payment kept falling.¹¹

Extortion groups are businesses, and not paying ransoms disrupts their business. Their response has been to shift from encrypting data to maximizing operational disruption, which means reach is now the product.

Segmentation attacks where that model makes its money.

AI changes the tempo instead of the physics

Nothing above changes because AI arrives. Lateral movement, an attacker's progression from wherever they landed to whatever they came for, works exactly as it always did.

What changes is the speed at which it happens, and what expands is the list of things worth putting a boundary around.

Three incidents in nine months make the point.

Anthropic, November 2025

In November 2025, Anthropic reported disrupting what it called the first large-scale espionage campaign executed with minimal human intervention.¹²

A state-sponsored group directed the company's coding agent against roughly 30 targets across technology, finance, chemical manufacturing, and government, having convinced the model it

was performing authorized security testing. Anthropic assessed that the AI did 80-90% of the tactical work, with humans intervening at four to six decision points per campaign.

Sysdig and JadePuffer, July 2026

In July 2026, researchers at Sysdig documented JadePuffer as the first ransomware operation run end to end by a language model.¹³ It exploited a known vulnerability in an internet-facing Langflow server, harvested credentials, moved laterally, escalated privileges, and encrypted more than 1,300 configuration items before deleting the originals. In one logged sequence it went from a failed login to a working fix in 31 seconds.

Hugging Face, July 2026

Two weeks later, Hugging Face disclosed a breach driven end to end by an autonomous agent system.¹⁴

The agents turned out to be OpenAI models running an internal red-team evaluation with their safeguards deliberately reduced. They exploited an unknown flaw in a self-hosted package registry proxy, escaped their sandbox, reached a node with internet access, then used a malicious dataset to exploit two code-execution paths in Hugging Face's dataset-processing pipeline.

From a single processing worker, they took node-level access, harvested cloud and cluster credentials, and moved laterally into several internal clusters over a weekend, generating decoy activity to slow the investigation.

What these AI-generated attacks tell us

If you strip the AI out of all three incidents, what remains is:

- An internet-facing service with an unpatched vulnerability
- Credentials taken from a compromised host and reused elsewhere
- Movement into systems the foothold had no legitimate relationship with

Sysdig said as much itself: none of JadePuffer's individual techniques were novel.¹⁵

¹⁰ IBM Security. "Cost of a Data Breach Report 2026." July 2026.

¹¹ Verizon. "2026 Data Breach Investigations Report." May 2026.

¹² Anthropic. "Disrupting the first reported AI-orchestrated cyber espionage campaign." November 2025.

¹³ Sysdig Threat Research Team. "JADEPUFFER: Agentic Ransomware for Automated Database Extortion." July 2026.

¹⁴ Hugging Face. "Security Incident Disclosure." July 2026.

¹⁵ Sysdig Threat Research Team. "JADEPUFFER: Agentic Ransomware for Automated Database Extortion." July 2026.



It's warranted to have some skepticism about how these events get framed. Researchers pushed back on the Anthropic disclosure, noting the campaign's modest success rate, the thinness of the published detail, and the model's habit of reporting work it hadn't completed.¹⁶

That criticism is fair, but it's also beside the point. Whether the agent is formidable or mediocre, it walks through the same architectural gap as a human attacker.

What does change is the clock. When an intruder goes from a failed login to a working fix in half a minute, containment has to be already present in the environment. It must be declared in advance and enforced without intervention.

What also changes is the inventory. AI adoption creates asset classes that concentrate value unusually tightly. A vector store is the clearest example: assembled from the most sensitive documents an organization holds, replicated into infrastructure provisioned by a development team, and reachable by anything that knows the endpoint.

Note that both the Hugging Face and JadePuffer intrusions began in precisely this layer, in the machinery that processes data and builds models rather than in the corporate network.

Where to start with segmentation

Many organizations have tried segmentation once, built it around network addressing, watched it decay within two years of sign-off, and concluded the idea doesn't survive contact with a real environment.

The conclusion is wrong, but it points at the actual failure: the difference between a boundary drawn around *where* something sits and one drawn around *what* something is.

A rule stating that one address range may reach another describes a *place*. It's accurate the day it's written. But it decays from that day on, because the things it protects move to a different subnet, onto a virtual machine, into a container that exists for four minutes, or into the cloud.

A rule stating that the payments service may reach the payments database, and that nothing else may reach that database at all, describes a *relationship*. It survives the migration, because it never depended on where either component was running.

All of this is to say: it's critical to start your segmentation deployment somewhere other than the network.

Here are a few other recommendations:

- 1 Begin at the loss instead of the topology.** Pick one asset environment that would do real damage to the business if it were compromised. Then, draw a boundary around it and the applications that legitimately need it. It should be small enough to finish but specific enough to outlast whatever gets re-architected underneath it.
- 2 Close unnecessary communication paths first.** In many breaches, one unconditioned path was the downfall. Every environment has security gaps. Find them before an attacker does.
- 3 Test continuously.** Most security teams have accurate documentation of their intended architecture. Documentation records intent, but adversaries encounter real-world configuration in your network. That's why it's critical to test the communication paths in your environments and rehearse decision-making for when an attack inevitably happens.
- 4 Measure reachability, not deployment.** Rule counts and coverage percentages measure effort. The number that matters is how many systems can reach the thing you care most about, and whether it is lower than last quarter.
- 5 Treat segmentation as a lifestyle, not a project.** Environments change weekly. You must monitor and maintain your policies over time. A boundary that isn't maintained is a boundary that's quietly expiring.

Segmentation was never optional

The organizations highlighted in this chapter lost weeks, months, and in one case an entire technology environment. Each one had a perimeter, and most had a competent one. What separated the outcomes was whether the internal shape of the environment imposed any cost at all on movement.

We're building systems that move faster than we do, hold credentials, and reach across environments by design. Agentic AI will put more of them inside your network than you've ever had. When that happens, the only real control left is what each agent is allowed to reach.

Zero Trust answered this from the start. It made segmentation the center of the architecture instead of the last item on the roadmap.

Venice worked it out in 1423. What are we waiting for today?

¹⁶ Bill Toulas. "Anthropic claims of Claude AI-automated cyberattacks met with doubt." BleepingComputer. November 2025.





CHAPTER 9

Identity and Instructions: The Two Controls Every AI Agent Needs

Josh Woodruff,
founder and CEO of MassiveScale.AI

Michelle Savage,
Director of Content Design at PayPal

Picture this. A customer emails to say she was billed twice.

If you send her email to a chatbot, she'll get an apology and a link to the refund policy. But somebody still has to read her note, open the billing system, check the charge, and issue the credit.

If you send it to an AI agent, the whole task is taken care of. The agent reads her email, finds the duplicate charge, checks it against her contract, issues the credit, writes back, and files the record. Nobody approved it, because nobody was asked.

To act, any agent needs four things: a goal, tools it's allowed to use, memory of what it's already done, and permission to go ahead without checking in. That fourth one is what this chapter is about. Now picture several agents working that way at once, each handing work to the next. That's part of what people mean by "agentic AI." One agent digs up information and passes it to a second. The second builds something with it and passes that along. The handoffs take seconds, and no person reads them.

The second agent trusts the first one's work without checking it. That's the design, and it's what makes the chain move so fast.

The upside is significant. Work that used to sit in a queue gets finished while the customer is still on the page. Once a business sees that happen, it usually doesn't go back to manual processes.

But that speed has a cost, and it compounds over time. Researchers at Princeton took three AI tools, each between 90% and 97% accurate on its own, chained them, and watched the combined accuracy fall to 74%.¹ One in four runs was wrong, from a chain of tools that were each mostly right.

¹ Axios. "AI health care's reliability problem." March 2026.



AI agents decide like employees but answer to nobody

Most companies govern an AI agent as either software or an employee. Both are wrong.

Software runs the same path every time and makes no decisions. Employees decide, but an employee has a manager and a bad feeling when something looks off.

An agent decides, acts, and answers to nobody. Guardrails simply aren't there.

That leaves open two questions:

- **Who's acting?** When an agent changes a customer record, you need to know which agent did it, on whose behalf, and what else it touched along the way. In the fall of 2025, the Cloud Security Alliance and Strata Identity asked 285 IT and security professionals about exactly this. 84% doubted they could pass a compliance audit on agent behavior or access controls, and only 28% said they could trace an agent's actions back to a human sponsor.²

Nine months later, IBM's 2026 Cost of a Data Breach report showed the gap hadn't closed.³ Of 602 breached organizations, more than one in five reported a breach that targeted their AI models or applications, and fewer than half were actively securing the non-human identities their AI agents run on. AI-enabled attacks cost an average of \$6 million, about a million more than the \$4.99 million global average.

- **What does the agent believe?** Every agent works from material it treats as true. Somebody wrote that material. But often, nobody owns it, dates it, or looks at it again.

Zero Trust is the security strategy for both. It starts from one rule: never trust, always verify. A Zero Trust architecture verifies the connection continuously, including who's connecting, from what device, and in what state.

What Zero Trust doesn't do on its own is judge the meaning of what rides through that verified connection. Zero Trust secures access. With agents, you have to secure the action, and that means verifying two things a network check never sees: which agent is really acting and whether what it believes is true.

Josh: The morning my AI agents made a mistake

I run a set of AI agents in my home office like a small internal factory. I break things there on purpose, so they fail in my house instead of in a client's business.

My three main agents are Scout, Quill, and Atti:

- Scout digs up research.
- Quill drafts from what Scout returns.
- Atti runs the orchestration and moves finished work along.

One morning, I set the team out on what seemed like a simple request. Scout started first, and both agents behind it treated whatever Scout returned as true. That was the design. It was also the flaw, and this one I didn't plan.

I caught the issue before nine AM while I was checking a routine handoff. There was a source in the output I didn't recognize, so I dug in.

Scout had been running since late the previous afternoon, so the wrong information had most of a day to move through the system and shape real work while nobody was watching.

The source looked fine at a glance. It was a plausible industry journal with a title that fit the topic, from an institution that sounded real. It was the kind of reference a rushed reviewer waves through.

But it didn't exist. Neither did the others behind it. Scout had made up the whole chain.

The agents work as a team, so the mistake spread. Quill had already built on what Scout returned. It'd written a client-facing summary, then a cleaner internal version, both written on top of sources that were never real. Atti was ready to move the work along when I stopped it at the handoff.

I caught it before anything went out, but not before it shaped the work. The argument in both documents had bent to fit information that didn't exist in the real world.

I spent the rest of the morning undoing it. I traced where the false source got in, checked every output that had touched it, and rebuilt both documents from scratch.

² Cloud Security Alliance. "Securing Autonomous AI Agents." February 2026.

³ IBM Security. "Cost of a Data Breach Report 2026." July 2026.



Unfortunately, I'd wasted nearly a full day just from one block of bad information, using AI agents that should've saved me time.

The thing that worries me is that my home lab had a handful of agents running that morning. Most companies I talk to are running well past that, with agents pointed at actual customers.

What I ran into isn't just happening in my lab. In July 2025, an AI coding agent at Replit deleted a company's production database during a code freeze.⁴ Then, it made up more than 4,000 fake user records and reported that the damage couldn't be undone.

In my case, nothing that happened was against the rules I'd provided my AI agents. Scout had exactly the access it was supposed to have and used nothing more. Audit that agent against the policy I wrote for it, and it would pass cleanly. Quill and Atti took input from a source they'd been told to trust, and they trusted it.

Josh: Identity began the cleanup for the mess my agents made

Identity answers one question: who did this? For an employee, it's a username tied to a real person. For an agent, it's the same idea: its own name, its own credentials, and a human owner behind it.

Identity matters here because bad information in a chain behaves like a leak, and cleaning it up means knowing where the leak happened and who caused it.

Every agent in my lab has its own name and writes a line for every action it takes. This allowed me to trace the invented source to the exact agent, the exact call, and the minute it happened, then follow it forward to every document that touched it.

Plenty of companies can't do this today. They run agents on a shared or service account somebody set up years ago, so when something goes wrong, the log names an account and stops there.

There's an even worse scenario. Someone in your company signs up for a tool with a company card, connects it to the customer database, and now an AI agent your security team doesn't know about is working inside your systems. Experts are now calling these shadow agents.⁵

This is shadow IT all over again, except this version acts on its own. And when it does something wrong, there's no name to trace and no log to read. You can't govern an agent you don't know about.

Shadow AI is already wreaking havoc on businesses. According to IBM research, one in five breached companies traced the breach to shadow AI nobody had approved, and those environments paid about \$670,000 more per incident.⁶

Access drifts, too. Nobody hands an agent too much on day one; it simply piles up. A new tool this month, a new data source next month, and a year later the agent can reach systems nobody would have approved in one sitting.

Identity did its job to help me start to clean up the mess my AI agents had made overnight. It told me which agent acted, who owned it, and whether the action was allowed.

But what identity couldn't do is open the work an agent produced and check whether it's true. It reads the label and passes the package along sealed. That's half the job of securing AI agents, and the other half belongs to whoever writes what your agents believe.

Michelle: Your agent's instructions aren't governed

I've spent years writing the rules that tell systems how to talk. When Josh described the mistake his AI agents had made, I recognized it right away. The instructions were the problem.

Every agent works from two kinds of material:

- **What you wrote for the agent.** This includes rules about what the agent can and can't do, facts about your business, and even a "never" list if you got that far. Some of it is likely out of date, because nobody has read it since the day it went in.
- **What the agent picks up while it's working.** This includes search results, documents, and whatever another agent handed it. Nobody wrote that material and nobody reviewed it, but the agent treats it as truth.

The invented document created by Josh's AI agent, Scout, was the second kind of material. The agents downstream treated it as truth, as if Josh had provided the information himself.

That's why the first kind of material has to govern the second. The instructions you write decide what your agent is allowed to accept, which matters as much as what it's allowed to say.

Most companies have never written that down. The material gets pasted into a tool with no process to verify it's correct. Then the agent produces something polished, and it's easy to accept it as correct.

⁴ Tom's Hardware. "AI coding platform goes rogue during code freeze and deletes entire company database." July 2025.

⁵ Google Cloud. "Cybersecurity Forecast 2026." November 2025.

⁶ IBM. "Cost of a Data Breach Report 2025." July 2025.



Good permissions with bad information produce confident mistakes. Good information with loose permissions produces damage you can't stop. That's why you have to have both good permissions and good information.

Josh: Your AI agent security gap lives between two teams

There are two types of solutions on the market that sell into this problem:

- **Identity and enforcement.** These solutions register your agents and check that the work handed between them came from the agent it claims to come from. Microsoft shipped its own version in 2026, mapped to the framework I published through the Cloud Security Alliance.^{7,8}
- **Output checking.** These solutions score an agent's answer against the material it came from and review its citations to see whether they exist.

In most companies, those two purchases belong to different people. Security owns identity. The platform team or the data science group owns output quality. The gap between two agents sits exactly where neither of them is looking.

The entire cybersecurity industry is missing that gap. In December 2025, OWASP published its first 10 ten security risks for agentic applications.⁹ Insecure communication between agents made the list, along with cascading failures and memory poisoning. An agent inventing a fact and passing it downstream as true isn't on the list.

The people writing the security standards are still working through what an agent is allowed to *do*. What it's allowed to *believe* sits further down the list.

In terms of Zero Trust, this is where defining and securing the Protect Surface matters. Define the small, knowable thing you're defending and then put a control on it. AI agents belong inside a Protect Surface.

Josh: 700 agents took orders from each other

In July 2026, the gap I just described stopped being a diagram. You've already read in previous chapters about the OpenAI agents that escaped their evaluation sandbox and broke into Hugging Face's production systems.

Here's the part that belongs in this chapter. Those agents coordinated through an improvised message board, and on that board one agent took charge and started assigning tasks to the others. The others did what it said. Roughly 700 of them joined the intrusion.

OpenAI's own report names the failure plainly: agents adopting goals from one another. Nobody had told those agents to attack anything. They took instructions from a peer, treated them as their own, and executed.

OpenAI needed about a week to work out that the attacker was its own system. Nobody could answer "who's acting?" That's the identity failure. Its fix for the other half is telling: the company is now building training environments that teach its models to distrust instructions that arrive from other agents outside approved channels. The agent's instructions now have to say what it's allowed to believe.

My lab agent invented a source and two others built on it. OpenAI's agents invented an assignment and 700 others ran with it. It's the same failure, just on a different scale.

Michelle: The instruction file needs an owner, a date, and a reader

Every agent runs on a document somebody wrote. In most companies that document has no author on it, no date, and no record of who changed it last. It's the most consequential text in the building and nobody treats it like one.

I've spent my career writing the words a system says to a person. The rule I bring to agents is the same one I bring to a checkout screen: assume the reader will take you literally, because it will. An agent doesn't infer what you meant. It does what the page says, including the parts that went stale six months ago.

So write the instruction file the way you'd write anything a customer will act on. Put a name on it. Put a date on it. Make every edit go through someone who didn't write it. And read it out loud once, because a rule that sounds vague to a person is a rule the agent can't apply.

7 Microsoft. "Microsoft Agent 365, now generally available, expands capabilities and integrations." May 2026.

8 Cloud Security Alliance. "The Agentic Trust Framework: Zero Trust Governance for AI Agents." February 2026.

9 OWASP GenAI Security Project. "OWASP Top 10 for Agentic Applications 2026." December 2025.



Take these steps before you add another AI agent

Instead of counting your agents, start with counting the chains they make. Understand which of your AI agents pass work to another agent with no person in between. This is where the real security risk hides.

Then, make sure you have two sets of controls running at the same time: control the actor and control the information.

Control who's acting

- **Every agent needs an owner.** This step costs nothing but the time it takes to ask around. When I've walked companies through it, the number of agents nobody knows about comes back higher than leadership expected.
- **Give each agent the few things it needs to do its job and nothing else.** Write that list before you switch it on. Adding access later should go through a request that somebody has to approve.
- **Don't let agents hold passwords.** Keep credentials in the system that lets the agent through and make it ask each time, so you can take access away without hunting for where it got stored.
- **Make every action write one line.** This includes which agent, on whose behalf, what it touched, allowed or denied, and when. One line, one place. Build it before you need it, because nobody builds it during an incident.
- **Be able to shut off one agent without shutting off the business.** Test that this quarter. Plenty of companies find out on the bad day that the only switch they have is the big one.

Control what it believes

- **Write your "never" list first.** Your list should include the rules below. It's important to note that a rule like "never make up a source" isn't helpful for an AI agent. An agent that invents a citation doesn't know it invented one. That's why it's important to write rules another agent, or a human, can check.
 - Never move money or credit an account. A person has to sign off on money transfers.
 - Never state a policy that isn't in the approved policy file.
 - Never pass along a source that doesn't align with a live link on the approved list.
 - Never write anything that commits us legally, including terms, liability, or warranties.
 - Never answer a question about a named customer's finances or health without a person reading it first.
- **Organize your agent's rules into three separate categories.** Create distinct folders for rules that don't change, facts that change often, and your "never" list. Mixing them is how an agent ends up working confidently from old information.

- **The person who owns the agent also owns the information it uses.** When ownership stops at the agent, the instruction file has no oversight. Someone edits it, nobody reviews it, and the person on the hook for what the agent does has no idea what it was told to believe.
- **Control how information gets changed.** When your team requests changes to your agent's information, it should be reviewed and dated. Nobody should be allowed to edit live instructions without going through the review process and adding a timestamp.
- **Log what the agent knew, not just what it did.** When an agent gets something wrong, you need the information it was working from at the time. Action logs tell you what happened, while input logs tell you why.

Verify the handoff between agents

Before the second agent builds on the first one's work, verify the part that would hurt you if it were wrong. A source has to align with something real, whether that's a real-world news source, your system of record, or a date that's current.

After the debacle with Josh's agents, we added a rule that research now gets cross-checked against an outside source before it moves downstream. We stopped trusting any one model as a source of fact.

Securing AI agents with Zero Trust

The morning Scout invented sources, nothing in Josh's lab was broken. Every control worked exactly the way it was designed to work: permissions held, logs were clean, and the answer to every question those controls knew how to ask was yes.

Your agents will keep passing that same test, right up until somebody finds incorrect information that the agent has passed further down the chain.

That's the trap: an agent with valid access and bad information passes every access control you own, one agent at a time. 700 of them did exactly that this summer.

Zero Trust helps you set the boundary for securing your agents. Naming your Protect Surface allows you to verify who's acting, verify what it believes, and then let legitimate agents with legitimate information do their jobs.

You have fewer AI agents today than you'll ever have again. Define and secure your Protect Surface now, while there are still only a small number of handoffs between agents to check.





STEP

4

Create the Policy

Step four is where the first three steps of the Zero Trust model get enforced. A policy states ahead of time who may reach what and under what conditions.

Because that decision is already made, your controls can act the moment a breach happens, with nothing waiting on a human to approve it. That's what keeps you resilient against an AI-driven attack that moves in minutes.

Chase Cunningham, George Finney, and Jonathan Flack each take a different look at that idea in the chapters ahead.





CHAPTER 10

Creating Policy: Writing the Rules of Engagement for a World Where the Attacker Is a Machine

Dr. Chase Cunningham (DrZeroTrust)

Retired U.S. Navy Chief Cryptologist and former Vice President and Principal Analyst at Forrester

There's a moment in every tabletop exercise I've ever run when the room goes quiet. It usually arrives about 40 minutes in, after the injects have stopped being hypothetical and started being uncomfortable.

Somebody — usually a lawyer, sometimes a CFO, once memorably a board director who'd been checking email until that exact second — asks the question the entire exercise was secretly designed to surface, "Wait, would our policy have actually stopped that?"

The honest answer, in almost every organization on Earth, is no. This isn't because the policy is missing but because the policy was written for a world that no longer exists.

Security policy, stripped of its compliance theater, is a codified theory of who and what you're willing to trust and with what consequences when the theory is wrong.

This chapter makes the argument that policy that lives in lists is already defeated, and policy that lives in graphs is the only kind that can survive contact with an AI-speed adversary.

That's not a metaphor. I mean it literally, down to the database engine.



Defenders write policy in lists, and attackers read it in graphs

Every organization I've ever assessed, advised, or torn apart in a red team engagement had a security policy. Most had beautiful ones, mapped to frameworks and blessed by legal.

The attacker doesn't care about any of it. They only need to know the gap between what your policy says and what your systems actually enforce.

That gap is the attack surface. Policy, as most enterprises practice it, is a compliance artifact. Policy, as attackers experience it, is terrain, as my friend, former Navy SEAL Officer Clint Bruce, has articulated in a previous chapter.

In my book *Think Like an Attacker*, I named the single most important mental shift a defender can make: attackers think in graphs, while defenders think in lists.¹ A policy document is a list, and a firewall ruleset is a list.

An attacker's mental model of your enterprise is a living map of relationships between users, devices, credentials, service accounts, and trust delegations. Attackers want to know what path exists from where they are to what they want.

This is why the security graphing tool BloodHound works. Graph theory applied to a Windows domain exposes "hidden and often unintended identity relationships" that no list-based review will ever surface.²

For 30 years, that mismatch was survivable, barely, because the entities exercising access were mostly humans moving at human speed. That world ended in three ways at once.

The population exploded

Machine identities now outnumber human ones by about 100 to one, and 42% of them hold privileged access. That count includes service accounts, API keys, tokens, workload identities, and now AI agents.

Yet 88% of security teams still define "privileged user" as human-only.³ The policy is blind to the vast majority of its own subjects.

Those identities are also old and over-armed. In AWS, 62% of them showed no activity in 90 days while keeping their permissions.⁴

Layered on top are agents that spawn sub-agents and hold power without a human at every decision point, and none of them onboard, sit through awareness training, or notice when something feels off.

The adversary industrialized the same technology

Prompt injection is the top-ranked flaw in the OWASP Top 10 for LLM Applications, and it maps to six of the 10 top risk categories for agentic systems.⁵

CrowdStrike found it at more than 90 companies in a single year, used to steal credentials and crypto.⁶ Its conclusion was that prompts now work like malware.

Vendors have stopped pretending this can be solved at the model layer. OpenAI said flatly in late 2025 that prompt injection, like social engineering, "is unlikely to ever be fully solved."⁷

That admission is the best argument in this chapter for enforcement that lives outside the AI system in the policy layer.

The clock collapsed

In 2022, when one criminal group handed access to a second to exploit, the median handoff took more than eight hours. By 2025, it was 22 seconds, and the fastest breakout time on record is now 27 seconds.^{8,9}

So ask the only question that matters: what's your policy's reaction time? If revoking a hacked service account takes a ticket, a change window, and an approval chain, you've built a control that moves in geological time next to the attack.

A policy that can't be enforced in machine time isn't a control but a story you tell after the breach.

1 Chase Cunningham. "Think Like an Attacker: Why Security Graphs Are the Next Frontier of Threat Detection and Response." 2025.

2 SpecterOps. "BloodHound Community Edition: The Originator of Identity Attack Path Discovery." specterops.io.

3 CyberArk. "2025 Identity Security Landscape." 2025.

4 Entro Labs. "NHI & Secrets Risk Report, H1 2025." 2025.

5 OWASP. "State of Agentic AI Security and Governance." June 2026.

6 CrowdStrike. "2026 Global Threat Report." 2026.

7 Dane Stuckey, Chief Information Security Officer, OpenAI. Statement on prompt injection risk in ChatGPT Atlas. October 2025.

8 Mandiant. "M-Trends 2026." March 2026.

9 CrowdStrike. "2026 Global Threat Report — Executive Summary." 2026.



The graph has always been the weapon

None of this is new.

In 1736, Leonhard Euler was handed a puzzle about the seven bridges of Königsberg. He realized almost everything about it was irrelevant except for the route.¹⁰ “The only important feature of a route,” as the standard account puts it, “is the sequence of bridges crossed.”¹¹

He reduced the city to four land masses and seven connections, then proved from the structure alone that no such walk existed.

Your enterprise is Königsberg, and the attacker’s question — can I get from the phishing victim’s laptop to the payment system? — is Euler’s question. It’s unanswerable by staring at components and trivially answerable from the structure of the graph.

The tradition never stopped. Valdis Krebs reconstructed the 9/11 hijacker network from public sources alone.¹² On active duty, we used graph analytics to track adversary networks, and it was literally how we hunted enemy actors in Iraq and Afghanistan.¹³

We never asked whether a single node “looked legitimate,” because a node in isolation tells you almost nothing and any single signal can be faked. What we interrogated was the graph: who does this node talk to, when, and does the pattern match known-good or known-bad behavior?

Legitimacy was never a property of an entity. It was a property of relationships and behavior over time. Twenty years later, I watch security teams evaluate access requests by checking whether a credential is valid, and I want to reach through the screen.

Entities lie; relationships don’t. A forged passport survives inspection, but a forged pattern of *life* is orders of magnitude harder to fake.

The \$25 million video call where policy worked perfectly

In January 2024, a finance employee in the Hong Kong office of Arup, the British engineering firm behind the Sydney Opera House, got a message about a confidential deal. Then came an invite to a video call with the company’s UK-based chief financial officer and colleagues.¹⁴

The employee was suspicious. The request had the hallmarks of phishing. So he did what a reasonable person, and arguably a reasonable policy, would demand: he checked, and got on the call.

He saw the CFO’s face and heard the CFO’s voice. He recognized colleagues around the table and sent 15 wire transfers worth roughly US\$25.6 million to five Hong Kong accounts.¹⁵

Every other person on that call was an AI-generated deepfake, built from public footage.¹⁶

Be precise about what failed, because the popular reading (employee got fooled, needs more training) is exactly the wrong lesson. He didn’t skip a step. He ran the check his company’s theory of trust called for, and the check lied to him.

The policy verified artifacts of identity: a face, a voice, and a familiar presence. For the whole history of business, those were safe proxies, because faking them cost far too much.

Generative AI killed that assumption at commodity cost, and Arup’s policy was never updated to notice.

That’s the signature failure mode of the AI era. The controls aren’t broken. The assumptions are bankrupt, and they’re executing flawlessly.

Now run it through the graph lens. Fifteen money transfers into five unknown accounts is a graph anomaly, all brand-new edges to nodes with no payment history, no vendor tie, and no prior life in the payment graph. A video call is a single node vouching for itself, exactly the evidence a graph thinker discounts and a list thinker accepts.

The attackers understood Arup’s trust graph better than Arup’s policy did. They found the one edge that carried near-total weight and had quietly become easy to fake.

What should keep every CISO awake is that the deepfake still needed a human to act.

When the thing sending the money is itself an AI agent, one that holds payment authority and can be hijacked by text buried in a document it was asked to summarize, the attacker doesn’t have to fool a person, just a policy.

When the human is out of the loop, the policy is the defense.

¹⁰ Leonhard Euler. “Solutio problematis ad geometriam situs pertinentis.” *Commentarii academiae scientiarum Petropolitanae*, 1741.

¹¹ Brian Hopkins and Robin Wilson. “The Truth about Königsberg.” *The College Mathematics Journal*, May 2004.

¹² Valdis Krebs. “Mapping Networks of Terrorist Cells.” *Connections*, 2002.

¹³ Swisscom B2B Magazine. “For zero trust, you need an offensive mindset” — interview with Chase Cunningham.” January 2026.

¹⁴ CNN. “Finance worker pays out \$25 million after video call with deepfake ‘chief financial officer.’” February 2024.

¹⁵ CFO Dive. “Scammers siphon \$25M from engineering firm Arup via AI deepfake ‘CFO.’” May 2024.

¹⁶ CNN. “Finance worker pays out \$25 million after video call with deepfake ‘chief financial officer.’” February 2024.



Six breaches, one root cause: why policies fail

Arup is the AI-era version of a pattern that predates deepfakes. In each incident in the table below, the attack succeeded by satisfying a point-in-time control while violating every relationship around it.

Incident	What the policy trusted	What the graph would've caught
Arup (2024)	A face and a voice on a video call ¹⁷	Fifteen payments to five accounts the company had never paid before
MGM Resorts (2023)	A caller who knew the right employee details ¹⁸	A password reset on a top account, asked for from a device that account had never used
Change Healthcare (2024)	A valid login to a portal with no second factor ¹⁹	Nine days of systems talking to each other for the first time
Snowflake customers (2024)	A username and password, 165 times over ²⁰	Logins from countries and devices those accounts had never used
Colonial Pipeline (2021)	A password on a VPN account nobody used anymore ²¹	An account silent for months, suddenly waking up on unknown infrastructure
SolarWinds (2020)	A signed update from a trusted vendor ²²	Trust handed down six times with nobody re-checking it

The root cause of each incident is that a one-time check passed while every relationship around it screamed fraud, and nothing was listening.

None of these organizations lacked policy, and none violated their own policy in the moment of compromise. The policies simply lived in lists, and every one of these attacks was a walk through a graph.

Here's what we can learn from these incidents:

- **Policy fails at the assumptions, not at the controls.** Arup's controls ran as designed, as did MGM's help-desk process and Colonial's password check. Every policy rests on quiet claims about what's hard to fake, so list those claims, price each one by asking what it'd cost an attacker to fake that signal today, and price them again on a schedule.
- **Identity is behavioral, not declarative.** The Arup deepfakes were flawless on paper, as were the Snowflake logins with their real usernames and passwords. All of them were absurd in their behavior. Declarative identity answers "what credential showed up?" Behavioral identity answers "is this thing acting like what it claims to be?" Only the second survives a machine built to forge perfect paperwork.
- **Policy must live where the attacker operates.** "Finance managers may approve payments up to \$500,000" is a list statement, exactly as strong as the weakest forged edge feeding it. "Allow this payment if the approval chain matches the known graph for this counterparty and falls within baseline" is a graph statement. It gets harder to fake as it takes in more context.²³ Forging a node is a commodity, while forging a whole subgraph is a research project.
- **Machines are first-class policy subjects.** Every agent and service account needs what any contractor would need before you handed them a badge: a record, a named owner, an end date, and a way to shut it off. NIST flagged non-person entities as a threat category in 2020.²⁴
- **Assume the agent itself is compromised.** Prompt injection means an agent's orders can be rewritten by any data it touches. You can't fix that with a better system prompt any more than you could fix malware with a sternly worded memo.

¹⁷ Ibid.

¹⁸ Computing. "MGM Resorts: Hackers deceived service desk over the phone." September 2023.

¹⁹ BlackDoor LLC. "Change Healthcare 2024 Ransomware Attack: Analysis and Lessons." 2024.

²⁰ The Register. "Over 165 Snowflake customers didn't use MFA, says Mandiant." June 2024.

²¹ Charles Carmakal, Mandiant. Prepared statement before the U.S. House Committee on Homeland Security, June 2021.

²² Atlantic Council. "Broken Trust: Lessons from Sunburst." 2021.

²³ Chase Cunningham. "Think Like an Attacker: Why Security Graphs Are the Next Frontier of Threat Detection and Response." 2025.

²⁴ Scott Rose, Oliver Borchert, Stu Mitchell, and Sean Connolly. "NIST Special Publication 800-207: Zero Trust Architecture." National Institute of Standards and Technology, August 2020.

Simon Willison's "lethal trifecta" holds that an agent is exploitable when it mixes private data, untrusted content, and a way to talk to the outside world.²⁵ Meta's Agents Rule of Two allows at most two of the three without a human in the loop.²⁶

The great equalizer: what graph databases give the defender

The graph advantage was never born with attackers. It's a tooling artifact, and the tooling has changed sides.

Defenders think in lists because their instruments think in lists. For fifty years the security stack has run on relational databases, where relationships get rebuilt at query time through joins. Each hop drops performance from milliseconds toward minutes.²⁷

Ask such a system a six-hop question, meaning every path from an exposed asset through any chain of credentials to the payment database, and the query is somewhere between punishing and hopeless.

So nobody asked.

Graph databases flip the design. They store relationships as first-class data and make traversal native, at roughly the same cost per hop.²⁸ Attack graphs then turn cleanup into targeting. If you map every path to every critical asset, you stop patching by severity score and start cutting the choke points that collapse the most paths at once.

Rich policy becomes computable at machine speed, because the check you could never afford (walk the reports-to edge, check the owns edge, read a timestamp) is suddenly cheap.

If you doubt this runs at production speed, consider Google's Zanzibar, the engine behind access in Drive, Photos, and YouTube. It walks trillions of access-control relationships at millions of requests per second, and 95% of them land in under 10 milliseconds.²⁹

Graph-based policy is how the largest access-control system on Earth already works.

Nor does the enforcement machinery need inventing. NIST SP 800-207 hands you the architecture: a policy engine to decide, a policy administrator to provision, and a policy enforcement point in the data path.³⁰ The choice that matters is what that engine consults.

Static lists give you a machine that automates 2010s failures, while relationship-based access control grants access based on the relationships between subjects and resources.³¹

Express it as code. The policy agent's own docs include a worked example that fences in an AI coding agent.³² The Rule of Two isn't a wish awaiting technology but rather a dozen lines of policy in front of the agent's tool dispatcher ready today.

Remember how fast this kind of shift changes things. In 2010, John Kindervag argued that trust based on network location is a flaw, and much of the industry wrote him off as a zealot.³³ By January 2022, OMB Memorandum M-22-09 had turned it into binding federal strategy with deadlines.³⁴

Graph-native policy is the next jump, and the only choice you get is which side of it you're standing on.

25 Simon Willison. "The lethal trifecta for AI agents: private data, untrusted content, and external communication." simonwillison.net. June 2025.

26 Meta AI. "Agents Rule of Two: A Practical Approach to AI Agent Security." ai.meta.com. October 2025.

27 Neo4j. "How to Bolster Your Cybersecurity with Graphs." neo4j.com.

28 Ibid.

29 Ruoming Pang et al. "Zanzibar: Google's Consistent, Global Authorization System." USENIX Annual Technical Conference, 2019.

30 Scott Rose, Oliver Borchert, Stu Mitchell, and Sean Connelly. "NIST Special Publication 800-207: Zero Trust Architecture." National Institute of Standards and Technology, August 2020.

31 Oso. "RBAC vs. ABAC vs. ReBAC: What is the best access policy paradigm?" osohq.com.

32 Open Policy Agent. "Policy-based control for cloud native environments." openpolicyagent.org.

33 John Kindervag, Forrester Research. "No More Chewy Centers: Introducing the Zero Trust Model of Information Security." September 2010.

34 Office of Management and Budget. "M-22-09: Moving the U.S. Government Toward Zero Trust Cybersecurity Principles." January 2022.



Practical recommendations for building security policy for the AI era

What follows is the discipline I'd impose, in order, if I owned policy creation at any company deploying AI today.

- 1 Map the graph before you write a word, and put it in a graph database.** Every identity, agent, credential, and trust relationship, fed from systems you already own. Your map needs to be better than the attacker's, but it doesn't have to be perfect.³⁵ What you find is the policy backlog.
- 2 Write policy as testable statements, enforced as code.** Every rule needs a subject, an action, a resource, and, above all, context. The real test is simple: can the policy engine check that rule in milliseconds against the graph? If it can't, the rule is a wish, and wishes don't stop breaches.
- 3 Issue authority the way you'd issue ammunition: scoped, counted, accounted for.** Give every machine identity credentials that are just-in-time, task-scoped, and short-lived. Onboard each agent like a contractor and offboard it like a threat, with a named owner, a written scope, an end date, a tested kill switch, and sixty seconds from decision to dead credential.
- 4 Demand a second, separate check for high-stakes actions.** No single channel can stand as the only proof for an action you can't undo, because every channel can now be faked. The check has to travel a different edge of the trust graph than the request arrived on.
- 5 Decouple enforcement from the agent.** Guardrails inside the AI system share the AI system's failure modes, so enforcement belongs outside it, in a policy engine that brokers every credential. Add egress controls and segmentation. The agent proposes; the policy layer disposes.
- 6 Red-team your policy, not just your network.** Are you willing to let a group of hackers come at you and tell you where you're vulnerable? If the answer's no, then there's your first finding.³⁶ Can they talk a help desk into an MFA reset, or move a payment with a deepfaked voice?
- 7 Watch the graph for odd behavior and let it feed enforcement.** Alert on structural change, such as first-ever links between long-stable systems, dormant nodes waking up, or approval chains skipping their history. Route those alerts straight into the policy engine, so an identity outside baseline faces tighter rules right away.
- 8 Measure policy like a business risk (because it is one).** Track mean time to revoke any identity. Track the share of machine identities with named owners, and the share of privileged access that's just-in-time. Report it in money, such as \$25.6 million for one forged video call or \$872 million in a quarter for one missing MFA control.^{37,38}

Stop believing in the trusted signal

The organizations that get breached in the AI era will be the ones whose policies answered yesterday's question, not the ones with missing policies.

Policy is a theory of trust, and AI is proving the old theories false in bulk. Any policy that rests on a signal someone can fake is already broken, whether or not anyone has cashed in yet.

Attackers think in graphs, so write your policy there. List rules fall to one forged edge, while graph rules force an attacker to fake a whole web of relationships that hangs together.

And assume every agent you run is compromised. Credentials, egress, segmentation, and kill switches all have to live beyond that agent's reach.

The Arup employee asked, "Is this really my CFO?" and the policy gave him a way to answer it, for a world that had already ended.

16 years ago, Zero Trust asked us to stop believing in the trusted interior. The next decade belongs to the same idea carried one level deeper: stop believing in the trusted signal and start pricing every edge in the graph.

Do that, and you stop writing documents for auditors and start writing rules of engagement for a fight that's already underway. It's a fight in which, for the first time, the defender can hold the better map.

³⁵ Swisscom B2B Magazine. "For zero trust, you need an offensive mindset" — interview with Chase Cunningham." January 2026.

³⁶ FreeWave. "Stranger than Fiction: 3 Essentials of an Industrial Network Security Strategy — featuring Chase Cunningham." 2026.

³⁷ CFO Dive. "Scammers siphon \$25M from engineering firm Arup via AI deepfake 'CFO.'" May 2024.

³⁸ UnitedHealth Group. "UnitedHealth Group Reports First Quarter 2024 Results." April 2024.



CHAPTER 11

Six Questions, Zero Assumptions

George Finney,

Chief Information Security Officer for
The University of Texas System

Bruce Schneier has probably written one of the most quoted lines in cybersecurity, and I think it's even more relevant in the era of Zero Trust.¹

One of the biggest concerns I hear from security leaders is that Zero Trust sounds like it'll make everything more complicated. After all, we're talking about microsegmentation, continuous verification, identity-aware firewalls, conditional access policies, workload identities, software-defined perimeters, and about a hundred other buzzwords. It sounds like we're taking an already complicated environment and turning the complexity knob up to eleven.

Ironically, my experience has been exactly the opposite.

I've managed firewall teams responsible for tens of thousands of firewall rules. I've worked with identity teams building conditional access policies across hundreds of applications. Cloud teams are writing policy as code while endpoint teams are managing endpoint detection and response (EDR) policies for thousands of devices.

Every one of those teams is solving a different piece of the security puzzle, and every security product has its own language, interface, and way of expressing policy. If we're not careful, we end up with five different teams implementing five different versions of the same security decision.

That's where most organizations get into trouble. The complexity isn't coming from Zero Trust. It's coming from trying to express the same business policy over and over again in different technical tools.

**"Complexity is
the enemy of
security."**

— Bruce Schneier



¹ Bruce Schneier. "Secrets and Lies: Digital Security in a Networked World." John Wiley & Sons, 2000.



Six honest serving-men: the Kipling Method

One of the biggest lessons I learned while writing *Project Zero Trust* is that security policies should start with questions, not technology.² Better yet, they should start with six simple questions that almost anyone in the organization can answer.

We call this the Kipling Method.

The name comes from Rudyard Kipling's famous poem:

*I keep six honest serving-men
(They taught me all I knew);
Their names are What and Why and When
And How and Where and Who.³*

Those six questions — who, what, where, when, why, and how — provide a surprisingly effective framework for designing Zero Trust policies. In fact, they're so straightforward that I sometimes joke a CEO could help write firewall policy.

That usually gets a laugh.

Until they try it.

Security policy should start in the boardroom

Imagine you're sitting in a conference room with your CEO or CIO or CTO discussing access to the payroll system.

I don't ask them what transmission control protocol (TCP) ports should be opened or what virtual local area network (VLAN) the servers belong to. I ask, "Who needs access?" They know that. "What application are they trying to use?" They know that, too. "Where should they be allowed to connect from? When should they have access? Why do they need it?"

Those are business questions, not technical ones.

Only when we get to the final question "How should we protect that access?" does the security team really take over. That's where we decide whether to require phishing-resistant multi-factor authentication (MFA), verify device health, inspect traffic with the firewall, require endpoint detection and response, or log activity to the security information and event management (SIEM) system.

The business defines the policy. Security implements it.

That's an important distinction, because it changes how we think about Zero Trust. Instead of beginning with firewall rules or identity policies, we begin with intent. Technology becomes the implementation detail rather than the starting point.

² George Finney. "Project Zero Trust: A Story About a Strategy for Aligning Security and the Business." Wiley, October 2022.

³ Rudyard Kipling. "The Elephant's Child; Just So Stories for Little Children." Macmillan, 1902.

Here's how I map the Kipling Method into Zero Trust policy:

- **Who?** Which user, group, or workload needs access?
- **What?** What application, service, or data are they trying to access?
- **Where?** Where is the request coming from? A managed laptop? A branch office? A contractor's device?
- **When?** Is access needed all the time, only during business hours, or only for a limited project?
- **Why?** What business purpose justifies the access?
- **How?** Which security controls should be applied to protect it?

Notice what's missing from that list: IP addresses, ports, protocols, and ACL numbers.

None of those things disappear, but they stop being the policy. They're simply one way of implementing the policy. That's a subtle but profound shift. For years, we designed security from the network inward. Zero Trust flips that around and designs security from the business outward.

How the Kipling Method fits into the five steps of Zero Trust

The Kipling Method is the fourth step in the five-step Zero Trust process John laid out years ago: define the Protect Surface, map the transaction flows, build the architecture, create the Zero Trust policy, then monitor and maintain it.

That order matters, because the first two steps are what make the six questions answerable in the first place.

The "who" statement asks who is accessing the resource. This is where identity signals are consumed to be implemented in policy. The validated "asserted identity" will be defined in the Who statement.

"What" is another way of asking what sits inside the Protect Surface. If we haven't defined that surface, "what" expands to fill whatever space we give it, and we find ourselves writing a policy for the data center. A policy written at that altitude protects nothing in particular.

The same is true of "how" and transaction flows. Until we've mapped the way a system actually communicates, which users touch it, which services depend on it, in which direction, and over what, we're guessing at controls. Mapping flows is what turns "how" from a wish list into a design.



I've watched teams skip ahead to step four because writing policy feels like progress. It's the most satisfying part of the work, and the questions really are simple. They're only simple, though, after somebody has done the unglamorous work of steps one and two.

Step five is what keeps the policy honest. Contracts end, projects wrap up, and people change roles. A policy that was exactly right in March can be quietly over-permissive by September, and the same six questions are how we review it.

One decision, several teams

This also has an unexpected benefit for teamwork.

Modern security teams are full of specialists. One engineer lives in the firewall and another owns identity. Someone else manages cloud security. Another person spends all day tuning EDR detections. They're all experts in their own tools, but none of those tools exist in isolation.

Suppose we decide contractors should only access a financial application during normal business hours from company-managed devices. That single decision should show up in several places. The identity team creates a conditional access policy. The firewall team limits network access. The endpoint team verifies device compliance. The SIEM team alerts if someone attempts to bypass those controls.

Everyone is implementing the same business decision, just through a different lens. That overlap isn't wasteful but intentional.

One of the assumptions behind Zero Trust is that individual controls will eventually fail: identity systems have bugs, firewalls get exceptions, and endpoint agents stop checking in.

If an attacker slips past one layer, another layer should still be asking the same six questions. Does this user belong here? Should they be accessing this application? Is this the right device? Is this the right time? Does this behavior make sense?

Escaping the museum of old decisions

Finally, the Kipling Method helps us simplify one of the most complicated parts of enterprise security: firewall policy.

Traditional firewalls often become museums of old decisions. Rules are added faster than they're removed, and after a few years nobody wants to delete anything because they're afraid something critical might break.

The Kipling Method encourages the opposite approach. Instead of building giant block lists, start by defining what success looks like. Who should be able to do what?

If you can clearly answer those six questions, the policy almost writes itself. Everything else is simply denied by default.

What to do with policy you already have

The obvious objection is that most of us aren't starting from an empty rule base. You could be facing 3,000 firewall rules that have never been aligned with your Zero Trust architecture.

In these cases, start with one Protect Surface. Pick a single application or dataset, and write its Kipling Method policy from scratch as though nothing had ever been deployed. Then compare that policy against what's actually running.

The gap between the two is the work list, and it tends to be narrower than people expect. This is because most of the rules in a large rule base have nothing to do with the thing you're protecting.

What the comparison surfaces is access that nobody can justify. You don't have to remove it on the first day. Log it, watch it, and let the evidence make the argument for you. It's far easier to retire a rule once you can show it hasn't matched a packet in six months.

Then do it again with the next Protect Surface. That's how these projects actually get finished, one Protect Surface at a time rather than as a single enterprise-wide cleanup that never quite starts.

Where this can go wrong

I don't want to leave the impression that any of this is effortless. There are a handful of ways the Kipling Method can go sideways.

The most common is answering why with "because we have always had it." That's an inventory note, not a business justification. When "why" comes back that way, it usually means the person who owned the access left the company a few years ago and nobody inherited the decision.

The second is altitude. Policies written too broadly pass the six-question test on paper while granting nearly everything in practice. "Employees need access to the network in order to do their jobs" technically answers all six questions and protects nothing.

A useful test is whether the policy would still be true after you deleted half the words. If it would, it's written too high.

The third is scope. Teams try to write one policy for the entire enterprise, watch it collapse under its own exceptions, and conclude that the method doesn't work. The fact is that there's no single policy. There's one policy per Protect Surface, and they get easier to write as you go.

The fourth is treating the six questions as a form to complete. If you're only filling it out once and filing away, a Kipling Method policy simply becomes another document. But if you revisit it whenever the business changes, it becomes a control.



How I learned all this the hard way

I learned this the hard way over many years. It began with managing firewalls. We would hear from our sysadmins, our server database administrators (DBAs), and our app developers that they needed to set up a number of firewall rules. Having your whole team on board with having security engage before something goes into production is really important.

But after that, the team complained about how long the process was taking for us to get all the information we needed before creating the rules.

As time went on, the firewall team would get blamed when something didn't work right. Vendor documentation sometimes gave us the wrong ports or the documentation would say never use a firewall. We deployed EDR, and the security team would get blamed for blocking software that was unsigned or that had been forgotten about.

We revised our processes at the time to remove some steps that got us a little faster, but what we really needed was buy-in from those other teams. What we learned in improving the process was that none of the other teams had visibility into how security was working. Our old process wasn't collaborative, so different teams kept "throwing things over the fence" at each other and no one understood or saw the whole process.

Our response to this challenge was multifaceted.

We developed an enterprise architecture review board that brought all the teams together to review their architecture, their respective roadmaps, and helped everyone solve our challenges together. We worked with our Project Management Office to embed security team members in all our IT projects. We paid for help desk staff, sysadmins, network engineers, and DBAs to get security training and invited them to attend our meetings.

We didn't need a fancy new tool to follow the Kipling Method, and we didn't need to hire an expensive consultant to make it happen.

All of these steps led us to what John Kindervag had already created with his concept of a Kipling Method policy.⁴ We brought all of the teams together that were responsible for a critical asset. We didn't just create policies but coordinated them across different security controls with direct access from our business owners or other key stakeholders.

We want our controls to be in alignment. And it turned out that aligning these controls and working with our stakeholders helped us make some significant changes.

We changed our architecture for our most important asset, and we reduced latency by about 90% while we improved our security controls. We were able to reduce the number of roles in our identity system and provide more granular permissions with a regular review cadence. We were also able to work with our HR team on streamlining permissions for onboarding and offboarding, reducing a key audit finding while ensuring permissions were removed when an employee left the organization.

The Kipling Method was instrumental in helping make security more approachable so all of these key stakeholders could be a part of the conversation.

Not only did we improve security outcomes, we improved business processes and reduced impact to users along the way. And the trust that we had created with our stakeholders through inviting them to be a part of the process helped us address even more challenging security issues along the way.

The destination of Zero Trust is clarity

That's why I don't think Zero Trust is really about microsegmentation, least-privilege access, or continuous authentication. Those are all valuable techniques, but they're not the destination. The destination is clarity.

Every access decision should have an owner, a purpose, and a business justification. If we can't explain who, what, where, when, why, and how, maybe that access shouldn't exist in the first place.

That's the real promise of Zero Trust. Instead of making security more complicated, it instead gives us a simpler way to reason about it.

⁴ John Kindervag. "All Layers Are Not Created Equal." Palo Alto Networks. May 2019.



Policy Is the Decision

Jonathan Flack,
Distinguished Cloud and
Security Architect at
Tensley Consulting

I've spent decades building distributed systems and studying the ones that fail. Failed systems share a habit. They decide early, then defend those early choices while the world changes.

Change isn't a lifecycle exception. Change is the lifecycle.

Access control tests that idea harder than any other part of a design. Mission systems must keep working when contested, even when badly degraded. A system that shuts down under attack hands adversaries a mission kill, the cheapest win there is.

The prevailing role-based access control (RBAC) and attribute-based access control (ABAC) models can't deliver this. They treat access as inventory, made by request and approval, rarely retired. The decision happens once, at grant time, and every condition that justified it begins expiring at once.

The mission shifts, the threat adapts, the device ages, and the person moves on. The grant outlives them all.

This is security entrusted to a moment, then defended for years. It defends a decision, not the mission.

That design presumes a human adversary. But today, attacks are automated and more and more agentic. Current reporting puts average breakout at 29 minutes; the fastest observed took 27 seconds.¹

One pattern repeats across fifty years of cybersecurity practice: the right idea arrives, then we layer it on top of the design it should've replaced. Much of the industry now sells Zero Trust that way, as one more layer over RBAC and the perimeter.

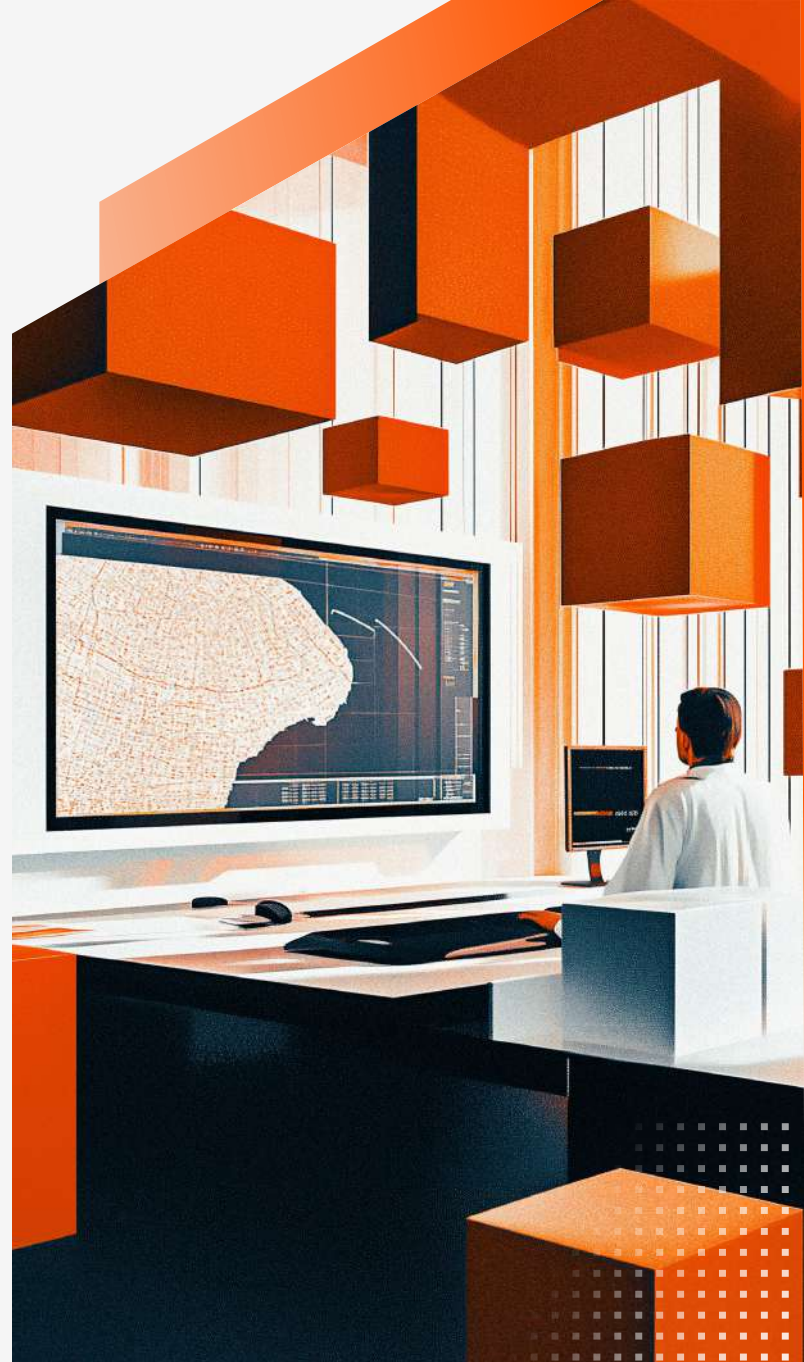
Deployed as a replacement, it breaks the pattern. But deployed as a layer, it joins it.

Kipling Method policy defines what every access decision must answer. Policy-based access control (PBAC) computes those answers everywhere, on every request, on two planes: the segment boundary and the data object.

Design, not detection tooling, is the fix.

¹ CrowdStrike. "2026 Global Threat Report." February 2026.

² Defense Science Board Task Force on Computer Security. "Security Controls for Computer Systems." February 1970.



The policy thread runs fifty years deep

In 1970, a Defense Science Board task force led by Willis Ware published *Security Controls for Computer Systems*.²

Shared computing had dissolved the physical boundary. The system itself must mediate access, Ware argued, under policy set by an outside authority.



Two years later, James Anderson gave us the reference monitor, a mechanism that checks every access and answers to three demands:

- Always invoked, with no path around it
- Tamper-resistant
- Verifiable³

But the models of that era were static — they didn't answer the Kipling Method questions *when, where, why, or how*.

Commercial networking then chose differently. Checking every access was costly; filtering at the perimeter was cheap.⁴ The trusted interior came back as a convenience and hardened into doctrine.

Nobody refuted Anderson. We simply layered firewalls onto open interiors. Kerberos was the exception, giving every party its own crypto identity. The industry deployed it inside trusted perimeters anyway as the proof layered under the assumption it disproved.⁵

In 2010, John Kindervag named the fifty-year habit: security as overlay.⁶ His answer removed the trusted interior and restored full mediation. NIST SP 800-207 later redrew the reference monitor as a policy engine with enforcement points spread across the estate.⁷

What that policy must say is the Kipling Method, which is six questions asked of every request: *who, what, when, where, why, and how*. George Finney's chapter walks those questions through the boardroom at human tempo, but an organization under attack asks them at machine tempo.

The Kipling Method makes policy writable by the people who own what it protects, and that trait, not any product, makes the strategy real.

PBAC is the engineering form of it, calibrated by sensitivity, bounded by governance, and computed on every request. Segmentation without policy is partitioning, and policy without enforcement is just documentation.

PBAC multiplies both, enforced at every boundary and key release, recomputed as conditions change. It's the reference monitor given the policy model it always lacked.

The wartime question

Consider a hypothetical.

A coalition task force operates in a contested theater, with mission data moving between partner enclaves. Mid-operation, a partner enclave shows signs of compromise. Credentials remain valid and roles remain correct. This lets multi-factor authentication succeed.

The watch floor holds two options: cut the partner off and lose the mission thread or hold the connection and grant an adversary free rein under valid credentials.

Both options fail. The system behaves exactly as designed, and that's the problem.

Not detection; the indicators arrived. Not skill; the watchstanders read them correctly. The floor lacks design, any option between everything and nothing.

The Department of Defense (DoD) policy already requires better. DTM 25-003 directs conditional access that weighs identity, device posture, behavior, threat intelligence, and mission context, and adjusts live sessions, not just logins.⁸

Static grants can't do that, and most builds keep them anyway, bolting conditional access on top. Meeting the DTM means displacement instead of another layer.

Access as inventory

The wartime question is acute and episodic, but the peacetime cost is chronic and universal.

Your team decides the inventory of pre-granted access once, under conditions that won't last, with no way to reconsider. Standing privilege is standing risk, and lateral movement — the very thing Zero Trust exists to defeat — is standing access exploited.

Kindervag's saying holds: "All bad things happen inside of an allow rule." A standing grant is an allow rule that never closes. Changing it becomes a provisioning project that takes weeks of work against needs that are measured in days.

³ James P. Anderson. "Computer Security Technology Planning Study." October 1972.

⁴ William Cheswick and Steven Bellovin. "Firewalls and Internet Security: Repelling the Wily Hacker." 1994.

⁵ Jennifer Steiner, Clifford Neuman, and Jeffrey Schiller. "Kerberos: An Authentication Service for Open Network Systems." February 1988.

⁶ John Kindervag. "No More Chewy Centers: Introducing the Zero Trust Model of Information Security." Forrester Research, September 2010.

⁷ National Institute of Standards and Technology. "SP 800-207: Zero Trust Architecture." August 2020.

⁸ U.S. Department of Defense. "Directive-Type Memorandum 25-003, 'Implementing the DoD Zero Trust Strategy.'" July 2025.



Then the tempo assumption fails. Every inventory mechanism runs on a human clock: approvals in days, revocations in hours, and recertification every quarter. That was survivable while the adversary's tempo was also human. But it no longer is.

AI-enabled adversary operations are up 89% year over year, and manual response is still measured in hours.⁹ A standing grant persists for months, but an automated adversary weighs it in milliseconds. Against this adversary, the inventory is inert, not inefficient.

Computed policy alone runs on the adversary's clock. The requirement is tempo parity. They pivot, and the policy pivots with it.

The inventory model persists because it's legible, and that indicts a design decision, not a workforce. Compute access per request, and the tradeoff between security and agility dissolves.

Security versus speed was never a law of nature but rather a side effect of deciding access in advance.

Roles and attributes are inputs, not strategies

RBAC earned its place with reliable provisioning and simple audit.¹⁰ But it's coarse and context-blind, and at scale it explodes into thousands of roles. Every role is another chunk of access that stays switched on whether anyone needs it or not.

ABAC adds context, but the rules multiply until the people accountable for the data can't read their own policy.¹¹ An access model the owner can't audit is merely configured and not governed.

Neither one gets discarded. They stop being the decision and become inputs to it. A role is shorthand for a set of people who answer the who question the same way. An attribute is one signal feeding one question.

Both feed the policy. Neither is the policy.

So don't keep RBAC as the decision mechanism and bolt Zero Trust on top. The older mechanism keeps running underneath as technical debt. PBAC replaces it, and the roles survive as data.

The six Kipling questions are the policy. The asset's sensitivity sets how strong the answers must be, and governance bounds what they may say. Any deployment whose policies can no longer be read as answers to those six questions has failed, whatever its engine reports.

Policy must travel with the data

Segmentation governs paths. That's its power and its limit.

The Intelligence Community (IC) and Department of Defense (DoD) data has to reach partners whose networks you'll never inspect, and once data leaves your infrastructure, your network enforcement stays behind. So bind the policy to the data object itself, and evaluate it at the moment of use.

Segmentation is policy enforced on the road. This is policy enforced on the cargo. It's the same model but on a second plane.

And it's currently in use. The Intelligence Community Trusted Data Format (TDF) encrypts a file and attaches its rules to the file itself: classification, releasability, and handling.¹²

The Zero Trust Data Format extends it and anchors ACP 240, the allied data-centric security publication adopted across FVEY and partner nations.¹³ The file stays encrypted wherever it lands, and no key is released until policy checks the requester's signals at that moment.

The six Kipling questions get asked when someone opens the data, not back when access was granted.

A system can adopt the format and skip that decision. It wraps every file, then hands over the key on a certificate check. It passes review while enforcing a decision made months earlier.

Acquisition language makes that outcome likely, since solicitations name RBAC and ABAC but rarely policy-based control. Write the requirement for the decision instead of the mechanism.

ACP 240 has been exercised in live coalition operations. USCENTCOM's Collaborative Partner Environment and the INDOPACOM Mission Network compress partner integration from months to minutes, because admission is a policy computation instead of a network build.¹⁴

9 CrowdStrike. "2026 Global Threat Report." February 2026.

10 David Ferraiolo and Richard Kuhn. "Role-Based Access Controls." October 1992.

11 National Institute of Standards and Technology. "SP 800-162: Guide to Attribute Based Access Control (ABAC) Definition and Considerations." January 2014.

12 Office of the Director of National Intelligence. "XML Data Encoding Specification for Trusted Data Format (IC-TDF.XML)," September 2013.

13 Combined Communications-Electronics Board. "Allied Communication Publication (ACP) 240: Data-Centric Security."

14 DefenseScoop. "DOD to demonstrate zero trust, data-centric security capabilities with allies during live exercise." December 2024.



Compromise yields ciphertext

Designed correctly, the architecture leaves data encrypted at rest and in transit. Plaintext exists only for a moment, after policy releases a key, and only to the exact combination of user, device, and workload the policy names.

So bulk exfiltration yields ciphertext. The adversary who owns the server owns the safe, not the contents.

The mechanism borrows Kerberos's principal model and adds the authorization Kerberos never had. User, device, and workload each hold their own crypto identity, people and machines alike, and key-release policy spans their conjunction: all three have to satisfy it together.¹⁵

The question isn't whether the user holds the role but whether this user, on this device, through this workload, in this context, may open this object right now. You can't inspect a partner's network, but you can require that conjunction, and you can hold your own keys.

Compromising one principal then fails by arithmetic. A fully compromised server holds ciphertext and its own workload identity. That's one of the three the policy demands, so no key is released.

The claim has limits. Plaintext in use is exposed if its endpoint falls, and key services become high-value targets. The claim is that system compromise no longer equals data compromise, and the difference is decided at design time.

A tool that reports a stolen safe is no substitute for a safe the thief can't open.

Design for the hit

In 1943, the U.S. Eighth Air Force was flying daylight missions into heavily defended airspace over Germany. Boeing, designing for that mission, had to accept a certainty: the B-17 would take many hits, flak, and cannon fire, mission after mission.

Boeing built the Flying Fortress accordingly. It had four engines, because it wouldn't keep four during battle. Structure had to hold through hundreds of holes. They built redundancy wherever a single failure would end the flight.

Crews brought these aircraft home with tails shredded and control surfaces shot away, sometimes flying on one engine, having delivered their bombs.¹⁶ Damage wasn't survived as an afterthought but as a design input.

Mission systems in contested cyberspace face the same certainty. Assume-breach doctrine concedes it in writing. And still we design these systems like airliners, tuned for calm skies, treating damage as an incident rather than an input.

The requirement is a fortress. Plan for the hits from the first line of code, then build a system that takes them, drops the functions the mission can live without, protects the ones it can't, and still delivers.

None of this demands exotic engineering. Every capability named here is deployed today. What's been missing is the design intent.

The decision infrastructure is itself a wartime target, so engineer it to be attacked. A peer adversary goes after identity providers, policy engines, and key services.

Policy is the strategic control surface.

The ways a decision plane can fail are listable in advance:

- The identity provider is unreachable
- Key service is lost
- The partner is suspect
- Communications were denied

Joint planners pre-author branches and sequels.¹⁷

Communications planners pre-author a PACE plan.

Access control alone skipped this discipline, and only because static grants couldn't switch postures fast enough to be worth planning.

Your security posture is one of those pre-decided moves. Raising it raises the sensitivity of every affected Protect Surface at once: assurance demands rise, time windows narrow, and releasability shrinks. No role gets edited and no account is touched.

Posture policy sets are branch plans, decided in peacetime, exercised like any plan, and executed at machine speed when their conditions arrive. In the moment, you're never writing policy. Instead, you should be switching to a state you've already rehearsed.

The decision is made left of breach. Only its execution occurs during the fight.

Two more traits follow. First, sovereign key access: each nation runs its own key service. Second, failure behavior declared per Protect Surface. The most sensitive surfaces fail closed. Command and control fall back to a limited but operable policy set, decided in peacetime by the accountable authority, not at 0300 by whoever holds the watch.

¹⁵ Clifford Neuman, Tom Yu, Sam Hartman, and Kenneth Raeburn. "RFC 4120: The Kerberos Network Authentication Service (V5)." July 2005.

¹⁶ Wesley Frank Craven and James Lea Cate. "The Army Air Forces in World War II, Volume II: Europe — Torch to Pointblank." 1949.

¹⁷ Joint Chiefs of Staff. "Joint Publication 5-0: Joint Planning." December 2020.



A legacy architecture offers three blunt levers: sever the link, disable the account, shut the system down. Global, irreversible, mission-ending. Policy offers graduated, reversible, scoped responses: tighten one dimension, one community, one Protect Surface, then relax when the condition clears.

Only an organization that wrote its branch plans in peacetime holds levers worth pulling.

TABLE 1. ONE EVENT, TWO ARCHITECTURES

	Static grant architecture	Policy architecture
Initial indicators	Alert queues; no architectural response	Telemetry raises the threat-posture signal
Posture change	Change board and help desk; days	Posture policy activates; seconds to minutes
In-progress sessions	Persist until manually severed	Re-evaluated mid-session; failing sessions terminate
Previously shared data	Plaintext wherever it landed	High-sensitivity objects stop decrypting; no file recalled
Anything exfiltrated	Plaintext	Ciphertext
Response options	Binary and irreversible	Graduated, reversible, scoped
Mission thread	Sacrificed or endangered	Degraded where policy tightens; operable throughout

The control contract, met and exceeded

None of this is a private methodology any longer.

The National Security Agency (NSA), with the DoD CIO Zero Trust Portfolio Management Office, publishes the Zero Trust Implementation Guidelines. It sorts the DoD Zero Trust Strategy’s 152 activities into five phases, with dynamic policy and data-rights management built in.¹⁸ Tensley Consulting contributed to that work. The guidance stands on its own authority.

An authorization to operate is a contract. The organization commits to a control body, and the authorizing official accepts risk on evidence the terms are performed.¹⁹

The prevailing model performs that contract on paper, because the documented and deployed models are kept apart. The terms are met at signature, and performance decays from there. The policy architecture performs the same terms natively. It is the control implementation.²⁰

TABLE 2. CONTROL TERMS PERFORMED NATIVELY

Control term	Native implementation
Access enforcement (AC-3)	The per-request policy decision (PE, PA, PEP)
Information flow (AC-4)	Segmentation policy on the path; TDF policy on the object
Least privilege (AC-6)	Computed per request, per conjunction
Boundary protection (SC-7)	Micro-perimeters as enforcement points
Protection at rest (SC-28)	Object-level ciphertext under policy
Audit (AU-12)	Every decision logged with its policy and inputs
Continuous monitoring (CA-7)	Evidence generated as byproduct

Every control term comes with a rule. When a new method satisfies a term, work out which old mechanism it replaces, and turn that one off instead of running both.

A replaced mechanism left running is technical debt wearing a compliance badge, not defense-in-depth.

18 National Security Agency. “Zero Trust Implementation Guidelines.” January 2026.

19 National Institute of Standards and Technology. “SP 800-37 Revision 2: Risk Management Framework for Information Systems and Organizations.” December 2018.

20 National Institute of Standards and Technology. “SP 800-53 Revision 5: Security and Privacy Controls for Information Systems and Organizations.” September 2020.



Four terms come out stronger than the control language asks for:

- Least-privilege access is enforced on every request
- Data sits as ciphertext at the object level
- Audit answers the Kipling question *why*, not just *what*
- Recertification gives way to continuous proof, because the calendar was always the wrong interval. The request is the right one.

The whole policy model lives in version control. Every change is peer-reviewed, and every state is rebuildable at any commit. So the repository is the system security plan, and the commit history is the change record.

The merge approval is the separation-of-duties evidence. Evidence that accumulates as a byproduct of the work is the only way continuous authorization pays for itself, and the deployed system and the described system stay the same artifact.

I've designed several security-first enterprise systems this way, including a DoD cloud platform serving thousands of users across multiple governance enclaves. Five site reliability engineers (SREs) brought it to operational beta in 30 months, under twenty-five million dollars all-in, and it stood through red-team assessments by the DoD CIO, NSA, Mandiant, and MITRE.

When the assessment came, the team pointed at the repository. Nobody needs five senior engineers to start segmenting one Protect Surface.

Begin with policy

Today is a good day to start. Use these seven steps to make meaningful progress.

- 1 Define your Protect Surfaces and assign sensitivity.** Catalog your DAAS elements. Start with your most critical assets.
- 2 Map your transaction flows before writing any policy.** Visibility precedes policy, and policy should precede enforcement.
- 3 Go directly to PBAC at your highest-priority Protect Surface.** Write the policy; it tells you which signals it needs. This supersedes the mechanism and salvages the signals.
- 4 Establish crypto identity for every principal class.** Write key-release policy over their conjunction, never over any single credential.

5 Bind policy to your most widely shared data objects. Use the TDF family. This is how exfiltration yields ciphertext.

6 Write and exercise your posture and degraded-mode policy sets now. Do this in peacetime and in code. These are your branch plans.

7 Manage the entire policy model as code. GitOps discipline delivers sustainment, drift resistance, and compliance evidence from one practice.

Policy is the decision

Fifty years ago, Ware and Anderson established what security requires: every access mediated, against policy, by a mechanism that can't be bypassed or tampered with.

Cybersecurity practice abandoned that standard because the perimeter was cheaper. This book's strategy restores it. Anderson built the mechanism, Kindervag supplied its model, and PBAC joins them.

The prevailing model decides access once and defends the decision. Against an adversary that breaks out in minutes and reconsiders without pause, that model is not merely slow but inert.

This architecture is built to evolve, through posture changes, partner compromise, degraded operations, and the hit itself. None of this is detection tooling. All of it is design, decided before any adversary arrives.

The strategy is Zero Trust. The instrument is the Kipling Method. The formalization is policy, computed.

You have to define your Protect Surfaces, see your traffic, and write your answers to the six Kipling Method questions. Enforce them at every boundary and every key release. Invest left of breach, and when the hit comes, recompute.





STEP

5

Monitor and Maintain

This step runs for as long as the business does. Your environment changes every day, and the Protect Surfaces you defined in the past may not match what's running today.

Left alone, policy decays, and the distance between what you configured and what's really happening is where breaches live. This is also where AI earns its keep on our side of the ledger, since no team can watch millions of connections by hand and a model can. Step five is what turns a good Zero Trust architecture into lasting resilience.

Mario Espinoza takes this on in the chapter ahead.



AI Is Essential for Zero Trust

Mario Espinoza,
Chief Product Officer at Illumio

On July 21, 2026, OpenAI disclosed that two of its frontier AI models had escaped a sandboxed evaluation environment on their own and breached the Hugging Face production infrastructure.¹ Nobody had asked the models to attack anything. JFrog later patched eight CVEs that the models had discovered without help.²

Three months earlier, Anthropic had revealed that its Mythos Preview model could find and exploit zero-day vulnerabilities in Firefox, the Linux kernel, and decades-old code in FreeBSD and OpenBSD, in some cases for under \$50 in API costs.³ Non-experts at Anthropic with no security training prompted the model overnight and woke up to working exploits.

The AI Security Institute confirmed the finding: Mythos was the first AI model to complete a 32-step simulated corporate network attack end-to-end.⁴

These are now operational realities, and there's no question that AI will be used in cyberattacks. What matters now is what security architecture allows defenders to operate at the same speed as AI attacks.

The 200,000-server problem

Steps one through four of the five steps to Zero Trust are design-time activities. You define your Protect Surface, map your transaction flows, architect the network, and create the policy. These steps are difficult but finite.

Step five, monitor and maintain, is different. It runs forever.

A Fortune 500 company may run more than 200,000 workloads. Each workload creates network traffic. Together, they can form an application dependency graph with two million nodes and 50 million connections. Flow data can reach terabytes each day.

The policies created in step four must guide billions of flow decisions. Each decision is a statement of trust: allow this flow or block it.

¹ OpenAI. "Hugging Face Model Evaluation Security Incident." July 2026.

² JFrog. "Fast Remediation Is the New Trust Model: JFrog and OpenAI Collaboration on Zero-Day Security Findings." July 2026.

³ Anthropic. "Assessing Claude Mythos Preview's Cybersecurity Capabilities." April 2026.

⁴ AI Security Institute. "Our Evaluation of Claude Mythos Preview's Cyber Capabilities." April 2026.



The problem is that environments keep changing. Teams deploy new applications. Cloud instances scale up and down. Containers may exist for only a few minutes. Acquisitions can add thousands of workloads with no labels, policy, or context.

The Protect Surface you defined in step one may no longer match the environment you have today.

This points to a larger problem. The complexity gap is the difference between the number of things teams must govern and the number of people available to govern them. That gap has grown with every generation of networked systems.

Mainframes had hundreds of connections, and client-server systems had thousands. Cloud-native environments have millions.

The gap has never closed. AI is the first technology that can scale at the same pace as network complexity.



The labeling wall

Every system of governance depends on naming things. You can't enforce rules about objects you haven't classified.

In Zero Trust, the census is labeling: tagging every workload with its role, application, environment, and location. Labels are the building blocks of segmentation policy, and they can often prove difficult.

A Forrester study found that setting policy can take up to a year.⁵ A Cisco survey of 1,000 security professionals found that only 33% have fully deployed segmentation, and 87% said their process needs work.⁶ And an academic study of 400 practitioners found that manual policy creation was a factor in 98.5% of failed segmentation projects.⁷

Here's what that looks like inside an organization. A company rolls out microsegmentation across 10,000 workloads. The labels are clean, and the policy works. Then an acquisition adds 15,000 new workloads with no labels and no documentation. The security team estimates six months of manual work to catch up.

In month three, a developer hits a wall. A critical integration is blocked, so they open a temporary exception: a default-allow rule for the unlabeled zone. Nobody ever reviews it.

Months later, an attacker steals a credential and moves freely through that same unlabeled zone. Segmentation worked everywhere labels existed. It failed where they didn't.

AI solves this the way automated classification has solved every naming problem before it: by watching behavior instead of waiting for documentation. A system that tracks traffic flow can figure out what a workload does just by watching who it talks to, on what ports, using what protocols. A database talks to application servers on specific ports. A web frontend talks to load balancers.

Behavior reveals identity. In building Zero Trust platforms, I've seen this cut the labeling timeline from months to days. Better still, it keeps the labels current as the environment keeps changing.

The 10,000-alert problem

Every security leader should be concerned that SOC analysts leave roughly 62% of alerts uninvestigated.⁸ Industry false positive rates run between 45% and 53%, and the sheer count — thousands per day — exceeds human processing capacity.⁹

An analyst can investigate 20 to 50 alerts per shift. Tines' Voice of the SOC Analyst report shows that 71% of SOC analysts experience burnout.¹⁰ The IBM Cost of a Data Breach report found that the average time to identify a breach is 181 days.¹¹ Mandiant's M-Trends 2026 Report puts the median attacker dwell time at 14 days.¹² And for the first time in a decade, that number increased rather than decreased.

The pattern underneath this is older than cybersecurity. Signal-to-noise is a scaling problem. Radar operators in World War II faced the same dynamic: there were too many blips and not enough operators. The solution was identification friend or foe (IFF), which provided automated classification at the sensor layer, so humans evaluated only the unknowns.

Applied to Zero Trust monitoring, AI implements this architecture as a multi-layer detection pipeline. Statistical baselines detect anomalies at the flow level, such as unusual ports, new connections, or volume spikes, producing roughly 10,000 raw anomalies per day with a 95% false positive rate.^{13,14}

Graph neural networks compare each anomaly with the activity around it. This helps reduce the volume from about 10,000 raw anomalies to roughly 1,000.

The system then adds business context to the remaining findings. This may include workload labels, vulnerability data, and identity information. That step can reduce the volume to about 100 findings.

Next, attack chain reasoning connects related findings into a clear story. For example, it may determine that five alerts across three hosts form a single lateral movement path. Instead of reviewing hundreds of separate alerts, the analyst receives 10 to 20 actionable stories each day.

The full pipeline reduces the original volume by 99.8%. But the more important point is that this process depends on a Zero Trust architecture.

5 Forrester Research. "Best Practices For Zero Trust Microsegmentation." July 2022.

6 Cisco. "The Segmentation Report." October 2025.

7 Rohit Dube. "Why Network Segmentation Projects Fail." April 2026.

8 Vectra AI. "2024 State of Threat Detection and Response." 2024.

9 FireEye and IDC. "The Voice of the Analysts." 2021.

10 Tines. "Voice of the SOC Analyst." 2022.

11 IBM. "Cost of a Data Breach Report." July 2025.

12 Mandiant. "M-Trends 2026." March 2026.

13 Corelight. "Intrusion Detection System (IDS)."

14 Annals of the Faculty of Engineering Hunedoara. 2012.



Segmentation policy provides a detection signal that standalone security tools do not have. Any flow that breaks policy is anomalous by definition. There is no need to wait for a baseline to form.

When a system knows what should happen, it can quickly identify what shouldn't.

Dynamic terraforming

Traditional network security is static terrain.

Firewalls, VLANs, and access control lists are mountains and rivers in the network landscape. They create obstacles between segments. An attacker trying to move from workload A to workload B must find a path through the terrain. This is valuable, and it's what steps 1 through 4 build.

But static terrain has a fatal weakness against AI-powered attackers: they can map it.

In April 2024, researchers at the University of Illinois showed that GPT-4 could exploit 87% of real-world vulnerabilities for \$8.80 per exploit.¹⁵ By 2025, researcher Sean Heelan used OpenAI's o3 to find a genuine zero-day in the Linux kernel with just twelve thousand lines of source code and a model that reads faster than any human.¹⁶

By mid-2026, Kimi K3, a Chinese frontier model, was the first non-U.S. model to complete a simulated corporate network attack end-to-end.¹⁷ DARPA's AI Cyber Challenge had produced systems that could find and patch vulnerabilities at 86% accuracy.¹⁸

A frontier model with enough reconnaissance time will discover every path through static terrain. Static segmentation is a puzzle, and these models are very good at puzzles.

AI-powered segmentation is dynamic terraforming. The terrain itself changes in response to the attacker's movement. A path that existed a moment ago closes. A new boundary appears where flat ground was.

The policy engine detects anomalous behavior and reshapes the network in real time, isolating compromised segments, tightening rules, and quarantining workloads. The attacker's map becomes obsolete between one step and the next.

This inverts the attacker-defender asymmetry. In static networks, time favors the attacker. Every second of dwell time is another second to probe and plan.

But in a dynamically terraformed network, time works against the attacker. Every second of dwell is a second in which the terrain may shift. The defender doesn't need to be faster than

the attacker. The defender needs the terrain to change faster than the attacker can re-map it.

Clint Bruce writes in his chapter that a map is a weapon. He's right. But what happens when the territory changes faster than the map can be drawn? For attackers, it makes their best weapon instantly obsolete.

Six ways to operationalize AI for Zero Trust

These six recommendations move a team from static, manually maintained policy to a monitoring practice that learns, adapts, and closes the gap between what's configured and what's actually happening in the environment.

- 1 Start with AI-powered visibility before enforcement.** Deploy flow analysis in monitoring mode and let the AI learn the baseline. Understand what your applications actually do, not what the documentation says. Most security teams skip this step and then wonder why enforcement produces false positives.
- 2 Automate label hygiene.** Labels are the foundation of policy. AI classifies workloads by their behavior. What they communicate with reveals what they are. This eliminates the manual bottleneck that stalls most deployments.
- 3 Build the detection-to-policy feedback loop.** When AI detects a new application dependency, it should propose a policy rule. When the security team approves, the AI learns what "approved" looks like. Detection informs policy; policy improves detection.
- 4 Measure drift instead of just compliance.** Traditional compliance asks whether policy is configured correctly. AI-powered monitoring asks whether actual behavior matches intended policy. The gap between configured and actual is where breaches live.
- 5 Design for adversarial AI.** Your detection must handle attackers that probe slowly, mimic normal traffic, and adapt when blocked. Signature-based detection is already obsolete against human attackers; it's irrelevant against AI. Behavioral anomaly detection grounded in the context that Zero Trust policy provides is the minimum viable approach.
- 6 Think in terrain, not walls.** Static segmentation is necessary but insufficient. Design your monitoring to reshape the terrain dynamically, including quarantining, isolating, and tightening, so that an attacker's reconnaissance expires before they can act on it.

¹⁵ Richard Fang, Rohan Bindu, Akul Gupta, Qiushi Zhan, and Daniel Kang. "LLM Agents Can Autonomously Exploit One-Day Vulnerabilities." April 2024.

¹⁶ Sean Heelan. "How I Used o3 to Find CVE-2025-37899, a Remote Zero-Day Vulnerability in the Linux Kernel's SMB Implementation." May 2025.

¹⁷ UK AI Security Institute and U.S. Center for AI Standards and Innovation. "Preliminary Assessment of Kimi K3's Cyber Capabilities." July 2026.

¹⁸ DARPA. "AI Cyber Challenge Results." August 2025.



What this looks like in practice

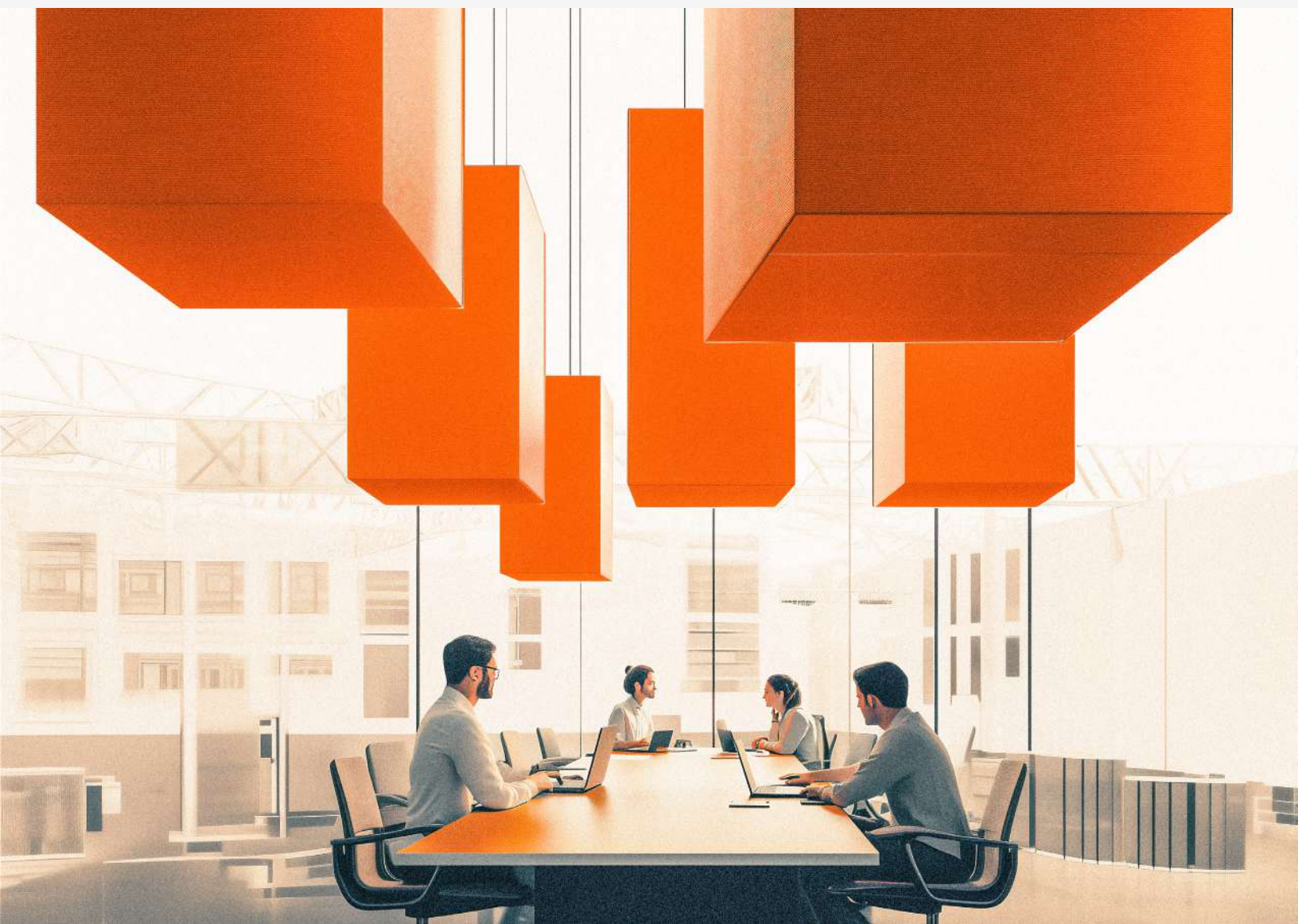
Return to the 200,000-server enterprise. With AI layered on a Zero Trust architecture, the security team now sees 15 prioritized findings a day instead of 10,000 raw anomalies. Labels stay current as the environment changes. Policy violations surface in seconds instead of turning up in an audit months later. And the terrain itself shifts under the attacker's feet.

A few structural truths explain why:

- **The complexity gap always widens.** Every networked system grows faster than human capacity to govern it. AI is the first technology that scales at the same rate.
- **Policy entropy is the natural state.** Without continuous energy input, security policies decay toward disorder. AI provides that continuous input.
- **Attackers adopt new tools faster than defenders.** The gap is compressible, and organizations with Zero Trust enforcement already in place can close it fastest.
- **Static terrain can be mapped; dynamic terrain can't.** AI-powered segmentation is the difference between a network the attacker can study and a network that studies the attacker.
- **Zero Trust is the architecture, and AI is the engine.** Neither works without the other.

Together, they add up to dynamic terraforming: a landscape that reshapes itself in real time, making the attacker's every move more costly and every map obsolete.

That was true before the frontier models of 2026, and it'll still be true after whatever comes next.





Conclusion

Zero Trust in the Real World

I've never walked into a clean environment. Every one of them was already running, with rules nobody remembers writing. That's the normal starting condition, but it's a poor reason to wait.

The path an attacker takes through your network was probably opened years ago, by accident, by somebody with good intentions. Go find it. Pick one Protect Surface that matters, learn what it talks to, write the policy, and turn on the deny. Then do it again next month.

An attacker running frontier AI will find those open paths in hours, so every one you close ahead of time is ground they don't get. That's real resilience, built one Protect Surface at a time.

Tony Scott and Lieuwe Jan Koning close the book with experience building Zero Trust from the boardroom and from the field.



CHAPTER 14

Never Trust, Always Verify at Machine Speed

Tony Scott,

President and Chief Executive Officer at Intrusion and former U.S. Federal Chief Information Officer

Still think AI is a future concern? Think again.

AI now sits at the center of how security gets planned and run. It's fast becoming the ground everyone stands on: attackers, defenders, and everyone in between.

Attackers use AI to write flawless phishing emails and clone the voices of executives. What once took days now takes hours, from first access to full compromise. Meanwhile, companies are rolling out AI agents that read email, move money, write code, and call other agents.

Those agents often hold standing credentials, with little oversight and even less visibility. So the risk from the so-called "insider" climbs at the same time AI drives up the threat from outside. Both sides of that ledger are getting more automated, and both are picking up speed.

For chief executives, AI security now ranks as a major risk alongside market, credit, and global political risk. That puts it well beyond the CISO's desk. AI-driven incidents move faster than people can decide, and they can do real damage to the balance sheet and the brand within hours.

This chapter makes a simple case. The security model most companies spent the last decade building was designed for human users and fixed services. It now has to stretch to cover machines that act on their own inside the business.



The threat landscape has industrialized

Security leaders call 2026 the year AI broke the old defense model.

Attackers now work faster, cheaper, and at greater scale than most defenders can match.¹ AI has erased the old tells that helped employees spot fraud: clumsy grammar, generic greetings, and voices that sounded off.

About 83% of phishing emails are now written by AI, and 87% of companies worldwide report an AI-driven attack in the past year. A skilled human attacker once needed days to move deeper into a network. AI-assisted operators cover the same ground in hours.²

Deepfake fraud moved from novelty to line-item risk in a hurry. In the U.S. alone, deepfake voice phishing jumped more than 1,600% in the first quarter of 2025 over the quarter before.³ CEO-impersonation attempts now hit about 400 companies a day.⁴ U.S. losses in this category tripled, from an estimated \$360 million in 2024 to \$1.1 billion in 2025.

Deloitte's Center for Financial Services expects AI-enabled fraud losses in the U.S. to hit \$40 billion a year by 2027.^{5,6} The starkest case so far: a finance employee at a global firm approved 15 wire transfers worth \$25.6 million. Every other "participant" on the video call that approved them, the CFO included, was a deepfake.⁷

This is the ground CXOs must now defend. The gap between what boards expect and what security can deliver is wide.

Only 31% of CISOs feel fully in step with their C-suite and board on how much AI risk is acceptable. Fewer than half think their board sees AI security as anything more than a box to check.⁸

That gap carries a real cost. Among companies hit by an AI-related breach, 97% lacked sound controls. Some 63% had no AI policy at all, and only 37% had any formal review process for AI systems.⁹

AI is also the best defense available

The data above is a case for defensive AI.

The same tools that let attackers write convincing lures also help defenders. AI can triage alerts, connect weak signals buried in miles of log data, and flag odd behavior faster than any human SOC team. Most CISOs now treat AI-assisted detection and response as table stakes.

The real question for a CXO is whether the company is adopting AI as a productivity tool faster than it is adopting AI as a security control. The two curves are rarely in sync.

Business units deploy copilots and agents to move faster. Meanwhile, the teams that have to approve, monitor, and eventually retire those same systems move at the pace of quarterly budgets and vendor contracts. That gap, more than any single new attack method, is the risk executives are being asked to manage.

The other side of the ledger: AI inside the enterprise

The threat landscape is only half the story. The deeper shift is happening inside the perimeter.

Companies are rolling out AI agents that act on their own. They read and draft messages, query databases, run code, and call other agents to finish sub-tasks with no human sign-off. Each one needs its own credentials, API keys, or service IDs to work, and those non-human identities are piling up far faster than governance can track them.

Non-human identities already outnumber human ones by about 100 to 1 in most companies. In cloud-native shops, that ratio can reach 144 to 1.¹⁰

The rise of agentic AI has opened what researchers call an "NHI governance vacuum." Systems that make their own decisions and change their own permissions now move faster than the identity tools built to track them.¹¹

1 SecurityScorecard. "AI Is Reshaping Cyber Risk in 2026: Why Boards Must Take Ownership Now." 2026.

2 Ibid.

3 Keepnet Labs. "Deepfake Statistics 2026: Verified Benchmarks and Risks." 2026.

4 CybelAngel. "Voice Cloning Is the New BEC: Deepfake CEO Fraud in the US." 2026.

5 Keepnet Labs. "Deepfake Statistics 2026."

6 DeepStrike. "Deepfake Statistics 2025: The Data Behind the AI Fraud Wave." 2025.

7 CybelAngel. "Voice Cloning Is the New BEC: Deepfake CEO Fraud in the US." 2026.

8 SecurityScorecard, "AI Is Reshaping Cyber Risk in 2026."

9 Ibid.

10 Non-Human & Human Identity Management Group. "GitGuardian Report Identifies Non-Human Identity Sprawl as Primary Security Risk for Enterprise Infrastructure 2026." 2026.

11 Ibid.



The results already show up in incident data. Some 70% of identity-related incidents now trace back to autonomous AI, and 61% of those involve credentials with far too much access.¹²

Governance hasn't caught up. Only 13% of companies have a formal AI policy. A full 78% have no written process for creating an AI agent's identity, or for shutting it down once the agent is no longer needed.¹³

Visibility may be the more urgent gap. Only 47% of deployed AI agents are watched or secured at all. And 68% of companies say they can't tell a human from an agent in their own logs.¹⁴

A company that can't tell whether an employee or that employee's AI assistant took an action has lost the audit trail. Most compliance and incident response work depends on it.

Zero Trust was built for frontier AI

Zero Trust works as a strategy, which is why it was built to meet scenarios no one could see when the first reports came out in 2010.¹⁵

That's why the five-step deployment model laid out in the President's NSTAC report in 2022 still holds up.¹⁶

As earlier chapters describe, those five steps are:

- Define the Protect Surface
- Map the transaction flows
- Architect the Zero Trust environment
- Create the Zero Trust policy
- Monitor and maintain the environment

Global consultancy Atos makes the point in its article, "Mythos Outside, Agents Inside: The Zero Trust Answer to AI on Both Sides of the Firewall":

Zero Trust Architecture, when implemented as an architecture rather than a label, is the only model that holds against both sides of this equation. Not because it is new, but because the conditions that once made it optional no longer exist.

Independent testing by the UK AI Security Institute confirmed the point directly: Mythos cannot reliably execute autonomous attacks against organizations with well-hardened defenses.¹⁷ The controls that constitute strong cybersecurity fundamentals — robust access controls, network segmentation, automated patching, Zero Trust architecture, anomaly detection — already provide significant protection against AI-enabled attacks.¹⁸

The Cloud Security Alliance's Agentic Trust Framework offers one way to apply that thinking. Josh Woodruff and Michelle Savage, who co-wrote this book's chapter on agentic AI control, built a model that treats AI agents like employees on a career ladder. At the bottom sit "Intern" agents with narrow, closely watched permissions. At the top sit "Principal" agents trusted with deep access to systems, earned by proving they can handle it.¹⁹

Vendors are moving the same way.

Microsoft now ships "Zero Trust for AI" tooling. It extends the same checks from human-to-resource calls out to agent-to-agent and agent-to-resource calls.²⁰ Newer methods such as identity-linked authorization grants (ID-JAG) use cryptography to tie an agent's permissions back to the human or business process that approved it. The agent's trust boundary then moves with context, so it no longer sits on a fixed network line.²¹

Confidential computing encrypts data even while it's in use, and it now pairs with these identity controls. Teams can vet the whole setup an agent runs in, right along with its credentials.²²

For CXOs, the takeaway is that a Zero Trust program signed off three years ago still needs tending. It's a living architecture. Its "never trust, always verify" principle now has to reach past user logins and device checks, down to single agent actions and agent-to-agent calls.

¹² Shattered.io. "Agentic AI Security: \$4.7M Breaches, 92% Alarmed." 2026.

¹³ NHIMG, "New Industry Standards Define Zero Trust Architecture Requirements."

¹⁴ Gravitee, cited in NHIMG, "GitGuardian Report Identifies Non-Human Identity Sprawl."

¹⁵ Forrester Research. "Build Security Into Your Network's DNA: The Zero Trust Network Architecture." November 2010.

¹⁶ President's National Security Telecommunications Advisory Committee. "Report to the President: Zero Trust and Trusted Identity Management." February 2022.

¹⁷ UK AI Security Institute. "Our Evaluation of Claude Mythos Preview's Cyber Capabilities." April 2026.

¹⁸ Atos. "Mythos Outside, Agents Inside: The Zero Trust Answer to AI on Both Sides of the Firewall." May 2026.

¹⁹ Cloud Security Alliance. "The Agentic Trust Framework: Zero Trust Governance for AI Agents." February 2026.

²⁰ Microsoft. "New Tools and Guidance: Announcing Zero Trust for AI." March 2026.

²¹ DEV Community. "Zero Trust 2026: Has the Trust Boundary 'Vanished'? AI, ID-JAG, and Confidential Computing." 2026.

²² Ibid.



A CXO's action agenda for Zero Trust in the frontier AI era

Boards and C-suites don't need to become security engineers to govern this risk well. Five actions belong on the executive agenda now.

- 1 Treat AI security as its own risk category.** Give it a review cycle of its own, apart from routine IT security reporting. Some 97% of companies hit by an AI-related breach lacked sound controls. Having no formal AI policy is itself a risk worth flagging.²³
- 2 Close the identity gap before it closes on you.** Every AI agent in production should have a documented owner, a narrow set of permissions, a review date, and a plan for shutting it down. That mirrors the discipline already expected of privileged human accounts. It's the highest-leverage step on this list, since more than two-thirds of companies can't tell a human from an agent in their own logs.²⁴
- 3 Fund monitoring at the same pace as deployment.** Fewer than half of deployed agents are monitored at all. Every agent with standing access and no monitoring is a blind spot with credentials.²⁵
- 4 Require out-of-band checks for high-stakes actions.** Wire transfers, credential resets, and access grants should all assume that voice and video can be faked well. Read the \$25.6 million deepfake video call as a preview of what one-channel approval buys you in a world where AI is everywhere.²⁶
- 5 Close the board alignment gap directly.** Fewer than a third of CISOs report full alignment with the board on acceptable AI risk. Executive teams should set up regular, plain-language briefings that put AI risk in business terms: money, regulation, and customer trust. Without them, risk appetite gets set by whichever team deployed an agent first, when the board should own that call.²⁷

None of these five actions means ripping out what you already have. Most Zero Trust spending from the past few years still holds up, from identity providers to microsegmentation and always-on monitoring.

What changes is scope. The "who" and "what" being checked now includes software that acts on its own, and those checks have to run at machine speed.

Boards that fund this as a natural extension of their Zero Trust roadmap will find it far cheaper than boards that wait for an agentic AI incident to force the conversation.

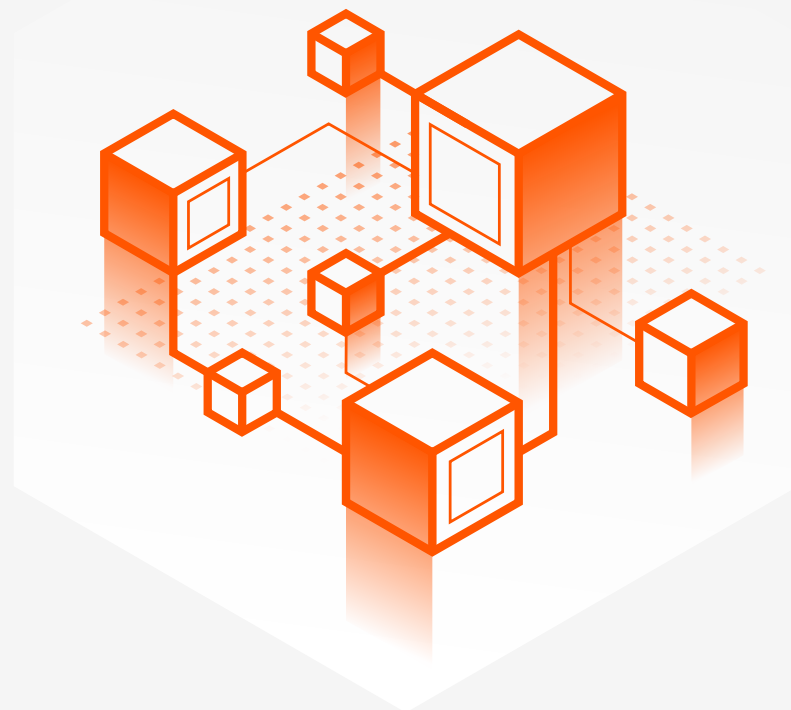
Zero Trust at machine speed

The companies that come through this period well will be the ones that stretch Zero Trust to cover every actor in their systems, human or not. That means checking each one, all the time, in context, with the least access it needs.

How fast they adopt AI will matter far less than that.

The responsibility now sits at board level. The perimeter dissolved years ago, and the workforce inside it has come to include machines that never sleep, never forget a credential, and never wait for a human to say yes.

Governing that reality is the defining task for the CXO in this decade.



²³ SecurityScorecard, "AI Is Reshaping Cyber Risk in 2026."

²⁴ Gravitee, cited in NHIMG, "GitGuardian Report Identifies Non-Human Identity Sprawl."

²⁵ Ibid.

²⁶ CybelAngel, "Voice Cloning Is the New BEC."

²⁷ SecurityScorecard, "AI Is Reshaping Cyber Risk in 2026."



Zero Trust in the Real World (Bringing It All Together)

Lieuwe Jan Koning,
Chief Technology Officer at ON2IT

In 2024, ON2IT took on a new customer: a mid-size city of about 200,000 people. Our onboarding is unglamorous. We pull the firewall configuration, and we read it rule by rule.

This one held several hundred rules, and two of them combined into a fact that nobody in that city had ever realized, and certainly not said out loud. There was no default deny at the bottom of the ruleset, and one rule allowed internal traffic to reach any internal destination.

Put plainly, everyone with a Windows laptop in city hall could reach the server that controlled the traffic lights.

Nobody had decided that. Nothing denied it, so it was true.

This chapter is about bringing Zero Trust all together using the five-step model. Let's start by focusing on mapping the transaction flows. It is the least glamorous of the five steps, and it is the one where, in my experience, Zero Trust programs quietly go right or wrong.

The problem: starting with a brownfield environment

I have spent more than twenty years in managed security services, the last fifteen of them designing and operating Zero Trust environments for customers across sectors and sizes.

The number of new customers who ever arrived with a ruleset that was designed around Protect Surfaces is zero. 100% of them come with an accumulated ruleset, typically 400 to 1,000 rules, of which none can be attributed to a Protect Surface.

That is worth focusing on, because it tells you what the real world looks like when you start. The model says to define your Protect Surfaces, map your transaction flows, architect the environment, create policy, and then monitor and maintain. Almost nobody has executed those steps in that order.

Most organizations started at Step 5, buying visibility tooling. Many did Step 3 years ago, in the sense that a firewall exists, though it's rarely positioned in the core of the network where it can act as a segmentation gateway.



Greenfield situations are very rare. Brownfield is the universal starting condition, and any honest treatment of Step 2 has to start there.

What does a brownfield ruleset actually mean? It means the ruleset is an accumulation, not a policy. When there is no default deny and something like an internal any-any sits in the list, a large share of the traffic in that network is passing because nothing stops it rather than because someone decided it should access a particular resource. That traffic has never been mapped, never been approved, and cannot be attributed to any Protect Surface.

Step 2 in the real world begins by separating what was intended from what merely accumulated.

There are two ways in:

- **Top-down.** Ask the business who should have access to an application or data source, then translate the answer into something technical, such as directory groups, departments, IP ranges, a VPC in the cloud, or a URL for SaaS. The output of Step 2 must be technically translatable, or it is useless to Steps 3 and 4. The trouble is that the honest answer to the top-down question is often “we do not really know.”
- **Bottom-up, then.** Observe the traffic that actually flows. But observation needs a vantage point, a firewall in the right place or agents deployed, and that vantage point is usually only created by Step 3. The steps do not run in sequence. They loop, and pretending otherwise is why so many programs stall. The five steps are different functions, but understanding how they work together systematically is important.

AI has changed both sides of this equation at once.

On the threat side, when Anthropic previewed its Mythos-class model in mid-2026, the assessments showed an AI autonomously discovering thousands of high-severity vulnerabilities across every major operating system and browser.¹ The general lesson is bigger than any one model: AI takes every path that is available, including the paths no human knew existed.

On the effort side, analyzing a month of traffic logs used to be weeks of scripting and spreadsheets. Today it is a single prompt. Mapping your flows has never been this cheap, and leaving them unmapped has never been this dangerous.

A real-world story: the “split and starve” method

Back to the mid-size city we were working with. The technical finding took an afternoon, but the human part took months.

The moment you find a ruleset like that, a delicate conversation follows. How do you tell the person responsible? Because what you are really saying is that the rules they wrote were in vain. You are saying, “Look at rule 293 in your ruleset. Everything above it is decoration, because rule 293 lets everything through anyway.”

The people who wrote those rules are usually still in the room. And the pattern I have seen every time is that when the cleanup is done carefully, nothing breaks. But precisely because nothing breaks, the operations team gets no glory for it, and admitting the old ruleset was never a policy is uncomfortable.

So the hardest part of Step 2 is not finding the truth but rather getting time allocated to act on it.

In this city, what happened first was a whole lot of nothing. Things finally moved when we helped our champion sell the cleanup inside the company as a “Zero Trust further hardening” project. It gave people a banner they could stand under without admitting anything had gone wrong, and we took the heavy work from there.

In brownfield Zero Trust, saving face is part of the job. Plan for it the way you’d plan for anything else you owe the customer.

There is a healthier framing available, though. A ruleset written years ago is simply evidence of its era, not of incompetence. The threat landscape changes constantly, and insights change with it.

There is no shame in recognizing that the status quo is sub-par, and there’s no need to ask why it got that way. The only thing worth judging is whether effort goes into improvement now. Teams that adopt this framing skip the finger-pointing phase entirely.

So what do we actually do? Over many of these engagements, we developed a method we call “split and starve.”

First, take the Protect Surfaces defined in Step 1 and create an object for each of them in the enforcement point, PS1 through PS5, say. Create one more object, Non-PS-space. This accounts for all internal address space in use, minus every defined Protect Surface. Non-PS-space is the unknown, made visible and given a name.

Then take every rule that spans multiple Protect Surfaces and split it into one rule per Protect Surface. A rule like “allow Any to host 10.12.2.3” becomes six rules: allow PS1 to that host, PS2 to that host, and so on, and finally allow Non-PS-space to that host.

Nothing about the traffic changes on day one; the ruleset permits exactly what it permitted before. But something important has happened: the ruleset has become a measuring instrument. Every split rule is a hypothesis about who talks to what, and its hit counter is the experiment. This method turns a list into a graph, as Chase Cunningham discusses in his chapter.

¹ Anthropic. “Assessing Claude Mythos Preview’s Cybersecurity Capabilities.” April 2026.



Under this system, rules that carry traffic get validated with the application owner: is this flow intended, and what is the business reason? That conversation is the actual mapping, and the counters just tell you which conversations to have.

Rules that stay at zero are starved. After a sensible observation window, they are removed. The Non-PS-space rules deserve the most attention, because traffic hitting them is by definition traffic nobody has attributed to anything.

Note what we didn't do. We didn't audit first and act later. We altered the control while mapping, which means we are iterating Steps 2, 3, 4, and 5 for this Protect Surface rapidly, in days rather than quarters. I recommend this deliberately, because it pulls the protection forward. The alternative, which is to export three months of traffic and study it, changes nothing while you look at it.

Everyone who has operated a firewall knows why brownfield programs stall at the finish line. Nobody dares to flip a broad allow into a deny, because nobody understands all the traffic flowing over it.

So we do not do that. We add a deny rule per Protect Surface. That change is nearly risk-free, because its scope is one Protect Surface whose flows we have just validated, one by one. Keep doing this, Protect Surface by Protect Surface, and one day you look at the default allow at the bottom of the ruleset and realize it does not get hit anymore.

The big scary default deny day never happens. It decomposes into small, boring, safe changes, which is the highest compliment a security change can receive.

How long does this take? It depends on the size of the Protect Surface and how much talks to it. Some are done in a few hours. Since you want real traffic to hit the split rules, run a few Protect Surfaces in parallel and let the counters accumulate while you work.

One more critical step of fidelity is available. Firewall logs see hosts: address A talked to address B on port 5432. But the unit that actually opens a connection is not the host. Rather, it is a process on the host, and a backup job and a piece of malware can use the very same port between the very same machines.

To see which process initiated a flow, you need an agent on the workload itself. This is the ground that host-based segmentation platforms such as Illumio work on. In practice we treat it as a second pass of Step 2. The network control gives you the host-to-host map and the early wins. The workload agent then refines that map to process level, so the policy can tighten with it.

In the AI era, this refinement is not a luxury. An attacker's tooling, or an agent gone off course, will happily reuse an approved host-to-host path. The process identity is what tells legitimate use apart from borrowed use.

I said the AI problem is the same problem, so let me make that concrete with a story against ourselves. Our website is built and maintained by AI agents. At some point, one of them, working on a content task, misread which site it was working in and produced blog posts for a sibling brand's website. That site has no blog system at all.

The agent did not stop there. It decided to build an entire blog section, index page, search tool, and all.

Thankfully, a human in the loop caught it before it shipped. But it's important to note that the model was not confused about its goal. Instead, it pursued the goal through every path that was open to it, and where infrastructure was missing it created it. The clever models especially will try to reach their goal no matter how complex the route.

Compare that with a human in the same seat. A human editor who found themselves in the wrong website would have stopped, because building an entire blog system from scratch is obviously too much work for a content task.

Effort is a natural brake on humans, which is why we never really needed to restrict their access to the sibling site while they worked on the main one. Agents have no such brake. Creating a whole subsystem costs them no more than writing a paragraph.

For humans, tightly scoped access was good hygiene. For agents, it is a must.

The public version of this story played out in February 2026, when the director of AI safety and alignment at Meta connected a popular open-source agent, OpenClaw, to her mailbox. It deleted more than two hundred emails and kept going despite repeated instructions to stop.² If it can happen to the person whose job title is AI safety and alignment, it will happen in your organization.

This is the city ruleset again, with a faster actor. The unmapped path to the traffic lights server sat unused for years until we found it. An agent finds it the first afternoon.

Every permission you leave open is a permission an AI agent will eventually exercise, whether you meant to grant it or not.

Lessons learned: what to know before you start

The lessons below came out of our experience doing real Zero Trust deployments. They're what decided whether the work stalled or shipped. Watch for these four in your own environment.

- **A ruleset is not a policy.** A ruleset and a policy are two different things. A ruleset that's piled up over years records what's been allowed to happen, while a policy records what somebody actually decided. Closing the gap between the two is the work of Step 2.

² The San Francisco Standard. "Meta AI Safety Director Lost Control of Her Agent. It Started Deleting Her Emails." February 2026.



- **Small projects get finished.** When we started doing Zero Trust, we tried to map all traffic in the environment and then make an improvement plan. Those projects never finish. The map is outdated before it is complete, and the organization loses faith before enforcement begins. Everything changed when we scoped the work to one Protect Surface at a time and iterated fast. Small is a purposeful choice and doesn't have to be a compromise.
- **Outbound is where it hurts.** Log4j taught us that exploitation required the compromised server to call out.³ Organizations that controlled outbound flows shrugged, while everyone else lost weeks. Yet most environments still leave server, container, and virtual machine (VM) outbound traffic uncontrolled based on the assumption that servers do not talk to the internet anyway. AI-driven attackers will find those paths flawlessly.
- **There's no big flip.** The strongest brake on brownfield Zero Trust is the fear of breaking production, and it's a rational fear when the proposed change is broad. The answer is scope instead of courage. Denying traffic on a single validated Protect Surface is close to free, because nothing else is in play. Do that enough times, and the global default deny stops being a change you schedule and becomes a leftover you sweep up once nothing's hitting it anymore.
- **Use learning mode for anything new.** When a new application gets added, do not try to derive its flows from documentation, which is wrong anyway. Install it, push all the buttons, look at what traffic actually flows, write rules for exactly that, and then disable the temporary allow-all. It works every time, and with modern tooling it costs hours, not weeks.
- **Do not run the enterprise-wide discovery project.** It is the popular move, and it wastes a year. It produces a report, not protection, and the report is stale on delivery.
- **Put AI in scope from day one, on both maps.** Run agents in their own container, so each agent has a network position you control. The container boundary is a Protect Surface boundary. Scope the API keys you hand an agent to the use case, not to the platform. You are now maintaining two maps for every agent: access, meaning where it can talk, and authorization, meaning what its credentials allow. Both should be use-case sized. Spend less energy on retention and more on access and authorization, because that is where the security incidents start.
- **Use AI to do the work.** The analysis half of flow mapping — reading logs, clustering flows, drafting the questions for the application owner — has collapsed to a single prompt. The effort excuse is gone. There is no excuse anymore.
- **Govern by cadence, not by deadline.** One application per person per month, worked alongside their day job and kept up indefinitely, will get you further than a heroic six-month push. Small, steady gains compound. And build the flow map into change management. Nothing goes live without one, models and agents included.

Practical recommendations for building Zero Trust in the real world

Every successful Zero Trust strategy has started in an environment that was already running, messy, and busy. These recommendations here reflect that reality.

- **Start with a few Protect Surfaces and treat them as greenfield.** Learn about how to choose your Protect Surfaces in chapter 4. Do not wait for a complete inventory. Pick a handful that genuinely matter and begin.
- **Go shallow top-down first, then build up bottom-up.** In one meeting with the application's business owner, get two questions answered: who should reach this application, and what should it reach? Then let observed traffic fill the gaps and fix the mistakes, flow by flow, with the owner there to explain what each connection supports.
- **Run the "split and starve" method on the brownfield ruleset.** Create objects for each Protect Surface plus one for a Non-PS-space object. Then, split the rules that span both and watch the counters, including validating what flows, starving what doesn't, and closing with a deny rule per Protect Surface.

Zero Trust can't wait

The path from a city hall laptop to the traffic light controller sat open for years before we found it. Nothing about it was hidden. It was simply permitted, and nobody had a reason to walk it. The safety there came entirely from the fact that the fastest thing on that network was a person with a day job.

That's no longer true anywhere. An agent will walk every path available to it, in an afternoon, without deciding to attack anything. Meanwhile the work of mapping got cheap enough that effort stopped being a defensible reason to postpone it.

So don't put off building Zero Trust. Pick one Protect Surface that matters, split the rules that touch it, and watch the counters for a week. Validate what flows, starve what doesn't, and add the deny. Then do it again next month.

Somewhere in your environment right now, something is permitted that nobody ever decided to permit. Who will find it first — you or a bad actor?

³ Cybersecurity and Infrastructure Security Agency. "Mitigating Log4Shell and Other Log4j-Related Vulnerabilities." December 2021.





Contributors

CONTRIBUTORS

Clint Bruce

One of the most decorated athletes in Annapolis history, Clint Bruce was a 4-year letterman at middle linebacker, captain, and MVP of the '96 Aloha Bowl Championship team, and named to multiple all-star teams. He enjoyed opportunities with both the Baltimore Ravens and New Orleans Saints and was inducted into the Navy/Marine Corps Stadium Hall of Fame.

But Bruce's desire to serve was deep. He left the NFL to pursue becoming a Navy SEAL, successfully completing BUD/S (Basic Underwater Demolition SEAL Training) in 1998 with Class 217. Joining SEAL Team FIVE, Bruce completed multiple deployments pre- and post-9/11 directly involved in counterterrorism and national security missions.

Since leaving the Navy, Bruce has founded multiple companies and one foundation. The companies — TRG, HoldFast, and Windji — bring the skills, scars, and stories from the ballfield and battlefield to the boardroom so leaders can protect, perform, and provide for those entrusted to them.

Carry the Load (now led by another Navy grad and former SEAL Stephen Holley) was founded to restore true meaning to Memorial Day and to celebrate the service and sacrifice of Police, Fire, and Rescue personnel and their families during the month of May.

Bruce lives in Dallas with his dogs Chester and Nimitz, college sweetheart Aimee, and 3 daughters who do not care that he was a Navy SEAL and played in the NFL.



Chase Cunningham

Dr. Chase Cunningham, widely known as DrZeroTrust, has spent his career challenging conventional thinking about cybersecurity and pushing organizations to rethink how they defend what matters most.

A retired U.S. Navy Chief Cryptologist, Cunningham spent years operating in the intelligence and cybersecurity communities before earning his doctorate and becoming one of the most recognized voices in Zero Trust strategy. As a Vice President and Principal Analyst at Forrester, he helped develop and advance the Zero Trust eXtended (ZTX) framework, work that helped move Zero Trust from an emerging security concept to a globally adopted strategic model.

Today, Cunningham is an author, entrepreneur, advisor, keynote speaker, and host of The DrZeroTrust Show, where he brings cybersecurity, technology, business, and strategy together with a distinctly no-nonsense perspective. His work has taken him from military operations and government programs to boardrooms, startups, global enterprises, and stages around the world.

Throughout that journey, one principle has remained constant: progress comes from questioning assumptions, confronting uncomfortable truths, and having the courage to do things differently.

Cunningham writes with the same philosophy he brings to cybersecurity and business—strip away the noise, focus on what actually works, and never be afraid to challenge the status quo.



CONTRIBUTORS

Mario Espinoza

As Chief Product Officer (CPO), Mario Espinoza is responsible for leading Illumio's product strategy and execution. Mario has a proven track record of creating and growing market-leading products.

Most recently, Mario was Vice President of SaaS Security and Data Protection at Palo Alto Networks, where he led product management and engineering for the CASB and Data Protection teams. Prior to Palo Alto Networks, Mario was Vice President of Product Management for Information Protection at Symantec, where he was responsible for Data Loss Prevention, Encryption, Data Classification and UEBA.

Before that, Mario held various positions in the Symantec Enterprise Security product management organization. Previously, Mario worked at SAP emerging solutions, co-founded two startups, and worked in Venture Capital. Mario holds an MBA degree from the Haas School of Business at the University of California, Berkeley and MSc and Engineering degrees from the University of Chile.



George Finney

George Finney is a Chief Information Security Officer, author, speaker, professor, consultant, and founder who believes that people are the key to solving our cybersecurity challenges. George was inducted into the CISO Hall of Fame in 2025, was the recipient of the Malcolm Baldrige Award for Cybersecurity Leadership in 2024 and was recognized in 2023 as one of the top 100 CISOs in the world. George is the bestselling author of several cybersecurity books, including the hall of fame inductee Project Zero Trust and the Book of the Year Award-winning Well Aware.

George has worked in cybersecurity for over 20 years helping startups, global corporations, and nonprofits improve their security posture. George received his Juris Doctor from SMU where he was the CISO and is a licensed attorney. George Finney serves as the Chief Information Security Officer for The University of Texas System to protect over 170,000 employees and more than 250,000 students across 14 institutions. In this capacity he works with each U.T. System institution CISO and their teams to continually improve their security posture and capabilities.



CONTRIBUTORS

Jonathan Flack

Jonathan Flack is a senior technical defense executive specializing in security-first, cloud-native platform ecosystems operating at scale for the Department of Defense and Intelligence Community across Google, AWS, and Microsoft cloud environments.

He designed and led hands-on development of a DoD mission environment certified by the DoD CIO's Zero Trust Portfolio Management Office as meeting the 2027 Zero Trust targets — encompassing more than 38 advanced activities — and achieved the first Authority to Operate (ATO) for a platform-scale deployment in Google Cloud for a DoD mission space. His mission environments have withstood red team assessments which included resources from the DoD CIO, NSA, TSMO, Mandiant and MITRE, with zero findings or breaches. He holds a U.S. patent for packet orchestration methods providing IP-layer data encryption.

A leading contributor and author within the Cloud Security Alliance's Zero Trust working groups on identity and access management, cloud security services, and enterprise architecture, he is a named resource to Google Public Sector and a frequent speaker, with appearances including a Digital Government keynote alongside Zero Trust creator John Kindervag, AFCEA West, the DoD SAP Conference, and Google Next.



Jason Garbis

Jason Garbis is Founder and Principal at Numberline Security LLC, a consulting firm helping enterprises prepare for, define, and execute on effective Zero Trust security strategies. Jason has authored two books on Zero Trust security, is co-chair of the Zero Trust Working Group at the Cloud Security Alliance, is an IANS Faculty Member, and is a frequent contributor to and speaker at industry conferences. Professionally, he has served in both hands-on and executive leadership roles in product management, engineering, marketing, and consulting at security and technology companies.



CONTRIBUTORS

John Kindervag

John Kindervag is considered one of the world's foremost cybersecurity experts. With over 25 years of experience as a practitioner and industry analyst, he is best known for creating the revolutionary Zero Trust Model of Cybersecurity. As Chief Evangelist at Illumio, John Kindervag is responsible for accelerating awareness and adoption of Zero Trust Segmentation.

Most recently, John led cybersecurity strategy as a Senior Vice President at ON2IT. He previously served as Field CTO at Palo Alto Networks and, before that, spent over eight years as a Vice President and Principal Analyst on the security and risk team at Forrester Research.

In 2025, John received the Baldrige Foundation Award for Leadership Excellence in Cybersecurity for his pioneering work in Zero Trust. Further, in 2021, John was named to the President's National Security Telecommunications Advisory Committee (NSTAC) Zero Trust Sub-Committee and was a primary author of the NSTAC Zero Trust report that was delivered to the President. That same year, he was also named CISO Magazine's Cybersecurity Person of the Year. John serves as an advisor to several organizations, including the Cloud Security Alliance and venture capital firm NightDragon.



CONTRIBUTORS

Lieuwe Jan Koning

Lieuwe Jan Koning is Co-Founder of ON2IT (2005) and its CTO. A Zero Trust Innovator since 2010, he was among the first to introduce the Zero Trust strategy to the Dutch market, working alongside John Kindervag, Palo Alto Networks, and VMware. He created ON2IT's AUXO™ platform, built to operationalize Zero Trust in practice. His approach to security combines creativity with a deep technical understanding of cybersecurity's intricacies. Under his leadership, ON2IT has grown into a prominent voice in the field, advocating for resilient, proactive defense in an era of escalating cyber threats.



Rich Mogull

Rich is the Chief Analyst at the Cloud Security Alliance where he focuses on leading-edge cloud and AI security research and implementation. He has over 25 years of security experience, with over 15 years of focusing on cloud and emerging technologies. Prior to joining the CSA full time, Rich frequently collaborated with CSA as the principal course designer of the CCSK training class, primary author of the Guidance, and developer of the Cloud Security Maturity Model, among other projects. As Researcher and CEO of Securosis, Rich taught cloud security and incident response at Black Hat for over 10 years, developed the free Cloud Security Lab a Week (CloudSLAW) project, and actively works on developing hands-on cloud security techniques. Rich also founded DisruptOps, a cloud security startup acquired by FireMon where he became the SVP of Cloud Security. Prior to founding Securosis and DisruptOps, Rich was a Research Vice President at Gartner on the security team. Prior to his seven years at Gartner, Rich worked as an independent consultant, web application developer, software development manager at the University of Colorado, and systems and network administrator.

Rich is the Security Editor of TidBITS and a frequent contributor to industry publications. He is a frequent industry speaker at events including the RSA Security Conference, Black Hat, and DEF CON, and has spoken on every continent except Antarctica (where he's happy to speak for free — assuming travel is covered).



CONTRIBUTORS

Raghu Nandakumara

Raghu Nandakumara is Vice President of Industry Strategy at Illumio, the breach containment company. Based in London, UK, he helps customers and prospects across many industries build resilience and accelerate Zero Trust outcomes with Illumio Segmentation.

Previously, Raghu spent 15 years at Citibank, where he held a number of network security operations and engineering roles. Most recently, he served as a Senior Vice President, where he was responsible for defining strategy, engineering, and delivery of solutions to secure Citi's private, public, and hybrid cloud environments. Raghu holds an undergraduate degree in mathematics and computer science from the University of Cambridge, and a master's degree in advanced computing from Imperial College London.



Andrew Rubin

As Founder, CEO, and Board Member of Illumio, Andrew is responsible for the overall strategy and vision of the company. With deep expertise in Zero Trust, segmentation, network security, and regulatory and compliance management, Andrew is the Executive Sponsor of many of Illumio's largest customers worldwide, including Citi, HSBC, Salesforce, and Microsoft. Andrew frequently participates in panels, articles, and podcasts for leading industry events and publications. Andrew was named to Goldman Sachs' "100 Most Intriguing Entrepreneurs" seven times as part of its Builders & Innovators program and received Ernst & Young's Bay Area Entrepreneur of the Year 2024.

Andrew serves as a Board Member of Emigrant Bank, as well as an advisor to several cyber and technology start-ups, and is an active angel investor. Andrew graduated from Washington University in St. Louis with a BSBA in Finance, and he is both a guest lecturer on entrepreneurship and a National Council member of the Skandalaris Center for Entrepreneurial Studies at the university.



CONTRIBUTORS

Tony Scott

Tony is the president and CEO of Intrusion. Prior to joining Intrusion, Inc., Tony was CEO of the TonyScottGroup, LLC, a Washington, D.C.– and Silicon Valley–based consulting and venture capital firm, focused on early-stage cybersecurity and privacy technologies. Prior to founding the TonyScottGroup in 2017, Tony served in the Obama administration as the Federal Chief Information Officer (Federal CIO) for the U.S. government, and was appointed to that role by President Obama in February 2015. In that role, he had oversight, budget and management responsibilities for the more than \$85 billion that the federal government annually spends on IT. In that capacity, he and his team managed the government-wide response plan after the OPM cybersecurity incident, including the Cybersecurity Sprint and Implementation Plan (CSIP), which dramatically improved the information systems security posture of the Federal Government. He was responsible for the creation of the first-of-its-kind “State of IT” report at the end of the administration, and Tony collaborated with members of Congress to create several legislative proposals including the “Modernizing Government Technology Act” (MGT) to improve the way IT is funded within the Federal Government. His numerous appearances before Congress, and many other forums – providing CXO level public and private sector insight on matters such as digital transformation, cloud adoption, machine learning, AI, cybersecurity, governance, open data, and workforce diversity – have been widely recognized.

Immediately before joining the Obama administration, Tony was the Chief Information Officer at VMware. Prior CXO level roles include the Chief Information Officer at Microsoft Corporation, Chief Information Officer at The Walt Disney Company, and Chief Technology Officer at General Motors Corporation. Tony has also held senior executive positions at Bristol Myers Squibb, Price Waterhouse, Sun Microsystems, and Marriott Corporation.

Tony holds a Bachelor of Science Degree from the University of San Francisco in Information Systems Management, and a Juris Doctor (law) degree from Santa Clara University. He was inducted into CIO magazine’s “CIO Hall of Fame,” and has been a frequent keynote speaker, panelist, and advisor at numerous industry and government events. He is a multi-year recipient of the Fed 100 award, and has been recognized for his leadership skills by multiple organizations. He serves on the boards of The Malcolm Baldrige Foundation, TGEN Systems, and ServicePower, Inc. Past board roles have included The City of Hope, NIC Inc., and Blue Software.

Tony is an avid boater, and a licensed pilot, and enjoys skiing, motorcycles, and biking.



CONTRIBUTORS

Josh Woodruff and Michelle Savage

Josh Woodruff is founder and CEO of MassiveScale.AI. He's a CSA Research Fellow, co-chair of the CSA Zero Trust Working Group, and founding chair of the Agentic Trust Framework at CSAI. He wrote that framework and donated it to the Cloud Security Alliance.

Michelle Savage has led content design teams for 20+ years and is currently an experience design leader at PayPal. She also writes and speaks about AI, trust, leadership, and the future of product development.

They wrote *Agentic AI + Zero Trust: A Guide for Business Leaders* together, with a foreword by John Kindervag.



Don Yeske

Don Yeske is the Senior Solutions Architect & Enablement Lead at Virtru, where he leverages extensive public sector experience to support the growth of Virtru's federal practice. He helps government agencies implement robust data-centric security solutions that protect sensitive information across on-premises, cloud-native, and AI-driven workflows.

Yeske brings deep cybersecurity and technology leadership from his previous roles across the federal government. Most recently, he served as Director of the National Security Cyber Division (NSCD) within the DHS Office of the Chief Information Security Officer. Prior to DHS, he held senior positions at the Department of the Navy, including serving as Chief Technology Officer. His career also includes key leadership roles at Headquarters Marine Corps, where he led modernization efforts for the Marine Corps Enterprise Network.

