Sample chapter

The Assurance Gap

Why Cybersecurity Compliance Is Broken — and What It Takes to Fix It

A Practical Guide for Executives, Board Members, and Security Leaders

William F. Leach Jr.

This excerpt contains *About This Book*, the contents, the preface, *The Argument in Brief* and Chapter 1 of the complete 399-page book, reproduced from the first edition.

Copyright © 2026 William F. Leach Jr. All rights reserved. Published by Transformyx Technology Services, LLC, Baton Rouge, Louisiana. ISBN 979-8-234-24281-5 (paperback). This sample may be shared freely in its complete, unaltered form.

Paperback and Kindle editions: amazon.com/dp/B0HLS5MWGS · williamleachjr.com

For every organization that has been told it was compliant,
and then discovered, too late, that it was not.

About This Book

Organizations pass their audits, hold their certifications, and sign their insurance attestations, and they are breached anyway. When they are, the distance between what they claimed and what was actually working becomes the most expensive gap in the enterprise. In 2024, ransomware crippled the City of Hamilton, Ontario. Its insurer paid nothing, because the multi-factor authentication its policy required had never been fully deployed.

The Assurance Gap argues that cybersecurity's costliest failure is not a shortage of tools but the absence of independent verification. Finance separates the people who keep the books from the people who audit them. Cybersecurity often lets the same providers operate the controls and vouch for them. Drawing on enforcement actions, court cases, insurance disputes, and the Pentagon's own hard lessons, the book shows how point-in-time compliance, conflicted service models, and diffuse accountability leave boards exposed.

It then sets out the fix: six pillars of cyber assurance, continuous evidence in place of annual attestation, an independent Cyber Assurance Officer, and a phased program that begins with a 90-day assessment. Written for executives, board members, and security leaders, it explains what to ask, what evidence to demand, and how to close the gap.

The expensive day is not the day of the breach. It is the day an examiner, an insurer, or a plaintiff's lawyer compares what was attested with what was running. Hamilton has already paid that invoice. This book is about not writing the next one.

Contents

Table of Figures ix

Preface: Why This Book Exists xi

How to Read This Book xix

The Argument in Brief xx

I How We Got Here 1

1 What Is Cybersecurity? 2

2 Shall We Play a Game? 36

3 When the Pentagon Went Commercial 47

II The Crisis 60

4 The Compliance Illusion 61

5 When Insurance Won't Pay 69

6 The Accountability Vacuum 84

III Structural Failures 105

7 The Conflict of Interest Nobody Talks About 106

8 Compliance Theater 116

9 The Regulatory Tsunami 124

10 The Forest, the Trees, and the Wheel That Keeps Getting Invented 146

11 Into the Abyss 158

12 The Weakest Link 170

IV The Six Pillars of Cyber Assurance **190**

13 Governance and Strategy **191**

14 Data and Identity Protection **210**

15 Technical Security Controls **228**

16 Security Operations **243**

17 The Human Factor **251**

18 Third-Party and Service Management **258**

V The Path Forward **277**

19 From Compliance to Continuous Assurance **278**

20 The Cyber Assurance Officer **288**

21 Building Your Assurance Program **312**

Appendix A: Questions for the Next Leadership Meeting **340**

Appendix B: HITRUST, CMMC, and the Crosswalk **343**

Glossary **352**

Acronyms **362**

About the Author **368**

Continuing the Work **370**

Index **371**

Preface: Why This Book Exists

A wise man, therefore, proportions his belief to the evidence.

— David Hume, *An Enquiry Concerning Human Understanding* (1748), Section X, Part I

In February 2024, the City of Hamilton, Ontario, discovered that ransomware had crippled approximately 80 percent of its municipal network. Emergency services scrambled. Citizen data hung in the balance. The attackers demanded C$18.5 million. The city refused to pay and turned to its insurance carrier for the recovery costs.

The insurer denied the claim in full.

The reason was not a technicality. Hamilton's cyber insurance policy contained a condition requiring multi-factor authentication (MFA) on critical systems. City staff had known since the fall of 2022 that the policy required MFA. An MFA pilot in 2023 reached only a few departments, and full deployment was still being prepared on February 25, 2024, when the ransomware arrived. When the insurer's forensic investigators confirmed that the absence of MFA — the policy's named condition — was the root cause of the breach, the contractual basis for denial was clear. The insurer covered none of the roughly C$5 million the city claimed, and Hamilton's taxpayers absorbed the entire C$18.3 million recovery cost because the distance between "we planned to do it" and "it was actually working" turned out to be millions of dollars wide.

Hamilton is not an outlier. It is an example.

Across every industry, in organizations of every size, a dangerous gap has opened between what they claim about their cybersecurity posture and what is actually true on any given day. Compliance programs certify a snapshot that begins degrading the moment the auditor leaves. Insurance applications contain attestations that no one continuously verifies. Board presentations describe a security architecture that may or may not reflect the current state of the environment. And when something goes wrong — as it inevitably does — the gap between what was claimed and what was real becomes the most expensive distance in the organization.

That is the pattern this book is written against. The board is briefed. The policy is signed. The control is on the roadmap. None of that is evidence. When the claim is denied, or the 8-K is due, or a regulator asks who knew, the distance between the slide and the log is no longer an IT problem. It belongs to whoever signed.

This book calls that gap the *assurance gap*, and the book's thesis is simple: the expensive problem is not a shortage of products. It is the absence of oversight, quality assurance, and return on investment for the controls already purchased. The cybersecurity industry has spent two decades getting better at building defenses while largely failing to build the governance structures that verify whether those defenses are actually working.

Why a Cat?

The cover of this book features a black panther — a big cat — leaping across a chasm. That image was not chosen for aesthetics alone. The cover's feline motif is deliberate, because few creatures in nature embody the principles of effective cybersecurity as precisely as a cat, and three of those principles carry the argument of this book.

Independence. Cats are not pack animals. They do not operate by consensus or defer to a hierarchy when making decisions in the field. This independence is the most important parallel of all. The entire thesis of this book rests on the principle that assurance must be independent of operations. The entity verifying the controls cannot be the same entity that implements them. A cat hunts alone, judges alone, and acts on its own assessment of the environment. The Cyber Assurance Officer operates the same way: independent of the managed service provider (MSP), independent of the CISO's reporting chain, accountable to evidence rather than to the team whose work is being evaluated.

Vigilance without fatigue. A cat does not sleep the way we do. Even at rest, its ears rotate independently, tracking sounds across a 180-degree arc, filtering signal from noise in real time. This is continuous monitoring — not the annual audit that checks a box and walks away, but the persistent, always-on awareness that catches the anomaly the moment it appears. An effective cybersecurity program does the same: it watches not once a year but every hour of every day, distinguishing

routine network traffic from the faint signature of lateral movement or data exfiltration. The other parallels are shorter, and the chapters that follow return to them: a cat sees in near darkness, as an assurance function must see into logs, configurations, and the gap between what a policy says and what a system does; it knows every boundary of its territory, because you cannot protect what you have not mapped; and it survives falls that would end most animals, because it is built to recover, which is what resilience means.

The leap. And then there is the image itself — the panther mid-flight over the chasm. The chasm is the assurance gap. Every organization stands on one side of it: the side where compliance certificates hang on the wall, where audit reports sit in binders, where the board has been told that everything is under control. The other side is where assurance actually lives — where controls are independently verified, where evidence is continuous, where peace of mind is earned rather than assumed. The gap between those two cliffs is real, and it is widening. The panther does not hesitate at the edge. It commits, it leaps, and it lands.

That is the purpose of this book: to help you make the same leap.

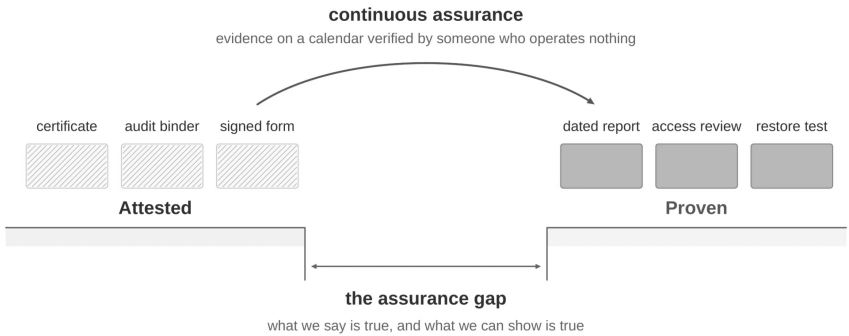


Figure P.1 The assurance gap. On the left, what an organization has been told. On the right, what it can show. The gap between them is not a shortage of products; it is the absence of independent verification.

Who This Book Is For

This book was written for leaders who make decisions about cybersecurity but may not be cybersecurity practitioners themselves — and for the practitioners who need to communicate the urgency of governance reform to those leaders.

If you are a **CEO, CFO, or board director**, this book will explain in plain language why your compliance program may be creating a false sense of security, why your cyber insurance coverage may not pay when you need it most, and what structural changes reduce your organization's risk exposure and your personal liability.

If you are a **CISO, CIO, or risk officer**, this book will give you the evidence and the framework to make the business case for continuous assurance — not as a technology purchase but as a governance model. It will also explain why the current model of concentrating all cybersecurity accountability in a single executive is structurally unsound and personally dangerous.

If you are an **MSP or MSSP operator**, this book will challenge you to examine the conflict of interest inherent in providing both security operations and compliance assurance from the same organization — and will present an alternative model that is better for your clients and, ultimately, better for your business.

If you are a **compliance professional**, this book will document the accelerating regulatory environment across SEC rules, NIS2, DORA, and the growing body of U.S. state privacy laws, and provide a practical framework for managing overlapping requirements without multiplying your workload.

If you are an **investor or advisor** evaluating the cyber governance market, this book will explain why the governance, risk, and compliance (GRC) market remains fragmented across hundreds of competing tools — and why none of them, on its own, verifies that an organization's controls work.

What This Book Is Not

This book is not a technical manual. You will not find firewall configurations, code samples, or step-by-step implementation guides for specific security tools. Those resources exist in abundance. What does not exist

— and what this book provides — is a comprehensive, evidence-based argument for why cybersecurity governance is failing and what a better model looks like.

This book is not the work of a disinterested party. The author's firm has built a framework for continuous cyber assurance, CyberCAAT, and its structure shapes the six pillars and the recommendations that follow. Where that framework is referenced, it is named. Nothing the book recommends requires it: every recommendation can be carried out with any tools, or none. Where the analysis rests on independently verifiable evidence — and it does — that evidence is cited to its source. The reader should evaluate every claim on the strength of its evidence, not on the identity of its author.

This book does not claim that compliance is unnecessary. Compliance with applicable regulations is a legal obligation and, when done well, materially improves security outcomes. What this book argues is that compliance *alone* — particularly compliance verified at a single point in time — creates a dangerous illusion of security that evaporates when tested by real-world attack, regulatory scrutiny, or insurance forensics.

How This Book Is Organized

Part I: How We Got Here is the history, in three chapters.

Chapter 1: What Is Cybersecurity? opens with the vocabulary the rest of the book runs on (the CIA triad, risk and its treatments, and defense in depth) and then traces the full arc of the cybersecurity story, from the mainframe fortress through ARPANET, the personal computer revolution, the commercialization of the internet, the rise of organized cybercrime, and the AI era, showing how the same pattern of technology adoption, false confidence, and catastrophic surprise has repeated in every generation. This chapter also confronts the human element head-on, documenting with current data from Verizon, IBM, and Ponemon why the greatest threat to any organization is not a genius hacker but the gap between what people know they should do and what they actually verify is being done.

Chapter 2: Shall We Play a Game? asks where this field got its self-image and finds that almost none of it came from the field. The black hat and the white hat are inherited from Western films, in a genre whose own historians say the convention was never reliably kept.

The word *hacker* began as a compliment and in a single season of 1983 was taken from the people who coined it. And three films — *WarGames*, *Sneakers*, and *2001: A Space Odyssey* — each got something important right that the industry then discarded: that intrusions defeat access management rather than mathematics, that the real prize is the integrity of information rather than its secrecy, and that the most dangerous machine is not the one somebody breaks into but the one doing exactly what it was told by people who never reconciled their instructions. The chapter matters because a picture of the threat that is wrong in the imagination produces a defense that is wrong in the budget.

Chapter 3: When the Pentagon Went Commercial tells the cautionary tale of what happened when the Department of Defense traded purpose-built military systems for commodity hardware and commercial software. From the Perry memo of 1994 through the Eligible Receiver exercise, the Solar Sunrise and Moonlight Maze intrusions, and the Buckshot Yankee breach that led to the creation of U.S. Cyber Command, the chapter documents how the world's most powerful military needed thirteen years and a series of existential breaches to build a centralized defense for the commercial networks it had rushed to adopt — and why every enterprise has inherited the same vulnerability without anything approaching the same response capability.

Part II: The Crisis presents the evidence that cybersecurity governance is failing. Three chapters document the cost of governance failures (Chapter 4), the growing crisis in cyber insurance claims and denials (Chapter 5), and the personal liability now falling on security executives and corporate officers (Chapter 6). Every claim is sourced to independently verifiable data from IBM, Verizon, the SEC, federal and state courts, the insurance industry, and peer-reviewed academic research.

Part III: Structural Failures diagnoses *why* governance is failing. Chapter 7 examines the conflict of interest created when the same organization provides both security operations and compliance assurance. Chapter 8 dissects the fundamental flaw in point-in-time auditing. Chapter 9 maps the accelerating regulatory landscape that is making continuous compliance not merely advisable but legally mandatory. Chapter 10 pulls back to the widest lens: when IT practitioners lose sight of the business they were hired to protect, the organization funds

a second company inside the first — one that generates activity instead of assurance, and cost instead of control. Chapter 11 ventures into the abyss: AI does not merely connect more systems — it acts, speaks in a human voice, and widens the assurance gap at a pace no previous technology has matched. Chapter 12 follows the weakest link: when trust in a vendor, an open-source library, or a software update becomes the weapon that breaches thousands of organizations simultaneously, the supply chain transforms from a business enabler into the widest point of the assurance gap.

Part IV: The Six Pillars of Cyber Assurance presents a comprehensive framework for what effective cybersecurity governance requires. Six chapters (Chapters 13–18) cover Governance and Strategy, Data and Identity Protection, Technical Security Controls, Security Operations, The Human Factor, and Third-Party and Service Management. Each chapter translates technical domains into language that any business leader can act on, with self-assessment questions and evidence requirements that organizations can apply immediately.

Part V: The Path Forward describes the transition from compliance to continuous assurance. Chapter 19 makes the business case. Chapter 20 introduces the Cyber Assurance Officer: an independent verification function, separate from the people who run the systems and from the people who watch them, reporting outside the IT line, whose only job is to confirm that the controls the organization claims are the controls that are running. Finance has required that kind of independence by law since Sarbanes-Oxley. Cybersecurity, in most organizations, still does not. Chapter 21 provides a practical roadmap for organizations at any stage of maturity.

A note on where to start. Chapters 1 to 3 are the history: how the field got its vocabulary, its self-image, and its inherited weaknesses. They explain why the crisis looks the way it does. If you would rather begin with the crisis itself, start at Chapter 4, where Part II begins, and come back to the history when you want to know why the pattern keeps repeating. Later chapters point back to Chapter 1 now and then, but only four of its terms are needed to follow them, and the Glossary defines all four: the CIA triad, risk, defense in depth, and control drift.

What to Remember, What to Do, What to Ask

Remember one thing: The gap between what your organization attests and what it can prove is the single most expensive risk on the balance sheet — and no product, policy, or certification closes it without governance that continuously verifies controls.

Do one thing this quarter: Ask your security team for evidence — not a report, not a slide deck, but time-stamped evidence — that three critical controls attested in your last audit or insurance application are functioning right now.

Ask one question in the next leadership meeting: "If a breach happened tomorrow and the forensic examiner compared our documentation to our actual controls, would the evidence match?"

A Note on Evidence

Every statistic, case study, and regulatory citation in this book is traceable to its primary source. Notes appear at the close of each chapter, providing full citations for every claim made in its pages. Where scholarly research is cited, the specific studies are identified. Where regulatory text is referenced, the specific sections and effective dates are noted.

The author believes that the cybersecurity industry suffers from an excess of unsubstantiated claims and a deficit of verifiable evidence. This book was written to be an antidote to that problem. If a claim in this book cannot be independently verified, it should not have been included — and the author invites readers to hold every assertion to that standard.

This is not a book that asks for trust. It is a book that earns it — one cited source at a time.

How to Read This Book

This book has two main audiences, and they do not need the same pages.

If you are a CEO, CFO, board director, or general counsel, read the Preface, Part II (Chapters 4 to 6), Chapters 7 and 8, Chapter 13, and Part V (Chapters 19 to 21). That is about 160 pages, and it holds the whole argument: the evidence, the cost, the role, and the 90-day start. Each chapter ends with one thing to remember, one thing to do this quarter, and one question to ask in the next leadership meeting, and Appendix A collects the questions in one place. Read the other five pillars (Chapters 14 to 18) as a reference, one at a time, when a domain reaches your agenda.

If you are a CISO, CIO, director of cybersecurity, senior manager, or risk officer, read it all, in order. Part III diagnoses the structures you work inside. Part IV is the operating framework, with the evidence each pillar has to produce. Chapter 20 is the charter for the role many of you will be asked to stand up, or to work alongside, and Chapter 21 is the program, phase by phase, with its cost model.

If you are a compliance professional or auditor, take the executive's path and add Chapter 9, with its regulatory map and the notification clocks of Table 9.1, Phase Three of Chapter 21, for the unified control mapping and what due care buys under the state safe-harbor statutes, and Appendix B, on crosswalks, HITRUST, and CMMC.

If you run an MSP or MSSP, Chapter 7 is written to you, and Chapter 18 describes the evidence your clients will start asking for.

Part I, the history in Chapters 1 to 3, belongs to everyone, but it can wait. Start where your job starts.

The Argument in Brief

The whole book in seven sentences and one table. Chapter 21 returns to the sentences, and Chapters 7 and 20 to the table.

1. Compliance records that a control existed when someone looked. Assurance shows that it worked between looks.
2. The people who operate a control cannot be the people who grade it. That is not a slight. It is how finance already works.
3. A certificate, a SOC 2 report, and a signed insurance application are representations. They become liabilities the moment the environment and the paper diverge.
4. Evidence compiled after an incident is a narrative. Evidence dated when the control ran is a fact. Insurers, regulators, and courts can tell the difference.
5. You cannot buy your way across the gap. Tools without a verifier are inventory. Frameworks without an operating calendar are decoration.
6. You do not have to mature every domain. You have to be able to prove the controls that, if they failed tomorrow, would take the business, the policy, or the officer with them.
7. The work needs an owner, a reporting line, and a cadence, or it is not work. The owner must not report to the function it is supposed to check.

Function	Who	What They Do
Build and Maintain	Internal IT / MSP	Operate the technology environment
Detect and Respond	SOC-as-a-Service / MDR	Monitor for threats; contain incidents
Assure	Appointed Information Security Officer of the company + independent CAO overlay	Verify controls match claims

The third row is the one most organizations lack; this book calls the function that fills it the Cyber Assurance Officer. Parts II to V show what its absence costs and what it takes to fill it.

Part I

How We Got Here

Chapter 1

What Is Cybersecurity?

Those who cannot remember the past are condemned to repeat it.

— George Santayana, *The Life of Reason*, vol. I (1905)

A History of Losing Control — and the Long Road to Getting It Back

Every dollar your organization spends on cybersecurity is a response to decisions made decades ago by people who never imagined the world that followed. Understanding this history is not academic — it explains why the tools you already own are not enough, why the threats keep evolving faster than the defenses, and why the most expensive problem in your security budget is not a missing product but a missing function: the independent verification that what you purchased is actually working.

Before the history, the objective.

Ask a room of executives what cybersecurity is *for* and you will get a list of things to avoid: breaches, headlines, fines. Ask an information assurance practitioner and you will get three words. **Confidentiality** — information is not disclosed to anyone unauthorized to see it. **Integrity** — information is not improperly changed or destroyed. **Availability** — information is reliably there when an authorized person needs it. Together they are the CIA triad (Figure 1.1), and no, not that CIA. No relation to Langley, no clearance required, and a certain amount of explaining to do at parties. The name is unfortunate, it has stuck for forty years, and the idea behind it is the most useful thing in this chapter.

Everything that follows in this book serves those three properties or it serves nothing. Every control, every framework, every regulation, and every dollar of the budget is ultimately a claim that one of the three is being protected. Information assurance adds two objectives that matter enormously once lawyers and insurers are in the room: **authentication**, the verification that a person or a message is who or what it claims to be, and **non-repudiation**, the evidence that stops someone from credibly denying an action they took. Hold on to that

last one. This book is, in a sense, about non-repudiation applied to an organization rather than to a person — the distance between what a company says its controls did and what it can still prove they did, a year later, to somebody who is under no obligation to believe it.

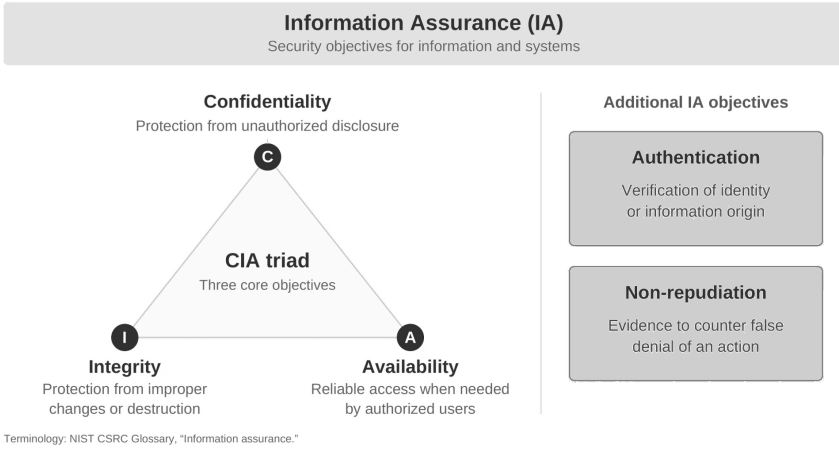


Figure 1.1 The CIA triad, and the two objectives information assurance adds to it. Everything else in this book is an argument that one of these five is being protected.

Risk is the other half of the vocabulary, and it is the half that gets mangled.

Figure 1.2 sets out the pieces. An **asset** is something of value: people, information, systems, services, reputation, trust. A **threat source** is who or what could cause harm — malice, error, failure, or a storm. A **vulnerability** is a weakness that lets harm reach the asset. A **safeguard**, or countermeasure, closes or narrows that weakness. **Risk** is what you get when you put them together: the possibility of loss, expressed as a likelihood and an impact, for a named asset. Not a color. Not a score out of five. And it comes in two flavors that are routinely confused — *inherent* risk, which is what you face before the safeguards, and *residual* risk, which is what is left after them. Residual risk is the number that matters, and it is never zero.



Figure 1.2 Risk begins with what you value. A threat source reaches an asset through a vulnerability; safeguards close or narrow it; what is left over is residual risk.

Notice the direction of the figure: risk begins with what you value, not with what might attack you. That is the reverse of how most security programs actually run. They start with threats — the briefing, the intelligence feed, the vendor's latest campaign report — and work inward, which produces a great deal of activity and no way to prioritize it. An organization that cannot list its assets cannot compute risk at all, because half the equation is missing: impact is a property of the asset, not of the attacker. Without value at stake there is no loss to assess. A report describing an adversary interested in something you do not have is not risk. It is news.

Of the assets, information is the one most reliably overlooked, and it is the one the adversary is usually after. Laptops get inventoried because they are countable and somebody signs for them. Servers get inventoried because they cost money and appear on a depreciation schedule. The customer list that was exported to a spreadsheet in 2019, was emailed to three people, and now lives in two personal cloud accounts is inventoried nowhere. Information has no serial number, and, unlike a laptop, it multiplies rather than moves — copying it leaves the original sitting in place, which is a large part of why the average breach runs for months before anyone notices.

The discipline that addresses this is information lifecycle management, and it turns on ownership. Every material information asset should have a named owner in the business — not in IT — who is accountable for four decisions: how sensitive it is, and therefore what protection it warrants; who may access it, in what role, and who reviews that list; how long it is kept, and under what legal or contractual obligation; and how it is destroyed when that period ends, with evidence that the destruction happened. IT can operate the controls that carry those decisions out. It cannot make them, because it does not own the business consequence of getting them wrong. An information asset with no owner has no sensitivity classification, and an asset with no classification has no defined protection level — so whatever protection it happens to have is an accident. Most organizations discover this during an incident, when the first question anyone asks — *what was in there?* — has no documented answer.

Once a risk can be stated, there are four things to do with it: **mitigate** it by implementing controls, **transfer** it by insuring or contracting it away, **avoid** it by discontinuing the activity that creates it, or **accept** it and record that you did, with a name and a date against the decision. Chapter 13 sets out that process. One further term belongs here, because the rest of this book leans on it: a **compensating control** is what stands in when the expected control is genuinely not feasible — a different control achieving a comparable result, documented as a substitution, approved by someone with the authority to approve it, and given an expiration date. Used honestly, it is a legitimate and necessary instrument. Used dishonestly it is the most popular way for an organization to tell itself it has closed a gap that is still open, and it is among the first things a competent assessor will test.

Defense in Depth

Residual risk is never zero, and that single fact decides how controls are arranged.

If any one control could be trusted absolutely, security would be a purchasing problem: buy the control, install it, go home. None can be. There is no silver bullet: no product, framework, or certification protects an organization on its own. Every control has a failure mode — a bypass, a misconfiguration, an expired certificate, a patch not yet applied, a capable person having a bad morning. **Defense in depth** is

the practice of arranging controls so that the failure of any one of them is survivable, because something else is still standing between the adversary and the asset. NIST's definition, carried in the glossary of SP 800-53, is "an information security strategy that integrates people, technology, and operations capabilities to establish variable barriers across multiple layers and missions of the organization."[1] The word doing the work in that sentence is *variable*. The barriers are not meant to be identical, and they are not meant to be equally strong. They are meant to fail differently.

Figure 1.3 arranges them. Notice that, like the risk figure before it, this one is drawn from the asset outward: information, critical services, and operational continuity at the center, then data protection, then the devices and networks and applications that carry the data, then identity, then the people, processes, and physical safeguards at the perimeter. The arrangement is fixed by what is being protected, not by what is attacking it. Which is the second reason an organization that has never inventoried its information cannot defend it: it has no center to arrange the layers around.

Take a single sequence through it. A phishing email captures a password — the outermost layer, the one made of people and training, has failed, and it will go on failing periodically, because it is made of people. Phishing-resistant multi-factor authentication can still stop the sign-in that the stolen password was supposed to enable, which is the identity layer doing the only job it was ever there to do.[2] If access is gained anyway, least privilege limits how far it reaches. Monitoring makes the intrusion visible while it is still happening. Recovery restores operations when monitoring has not. Every one of those controls exists for a day on which an earlier control has already failed. That is precisely what makes them layers rather than a list.

It is worth separating this from the compensating control described a moment ago, because the two are conflated constantly and the difference matters to anyone who will ever be assessed. A compensating control substitutes for a control that is *missing*. A layer backs up a control that is *present and working*. The first is an exception with an expiration date; the second is the design. An organization that presents its layers as compensating controls has told the assessor, without meaning to, that it does not have the primary controls at all.

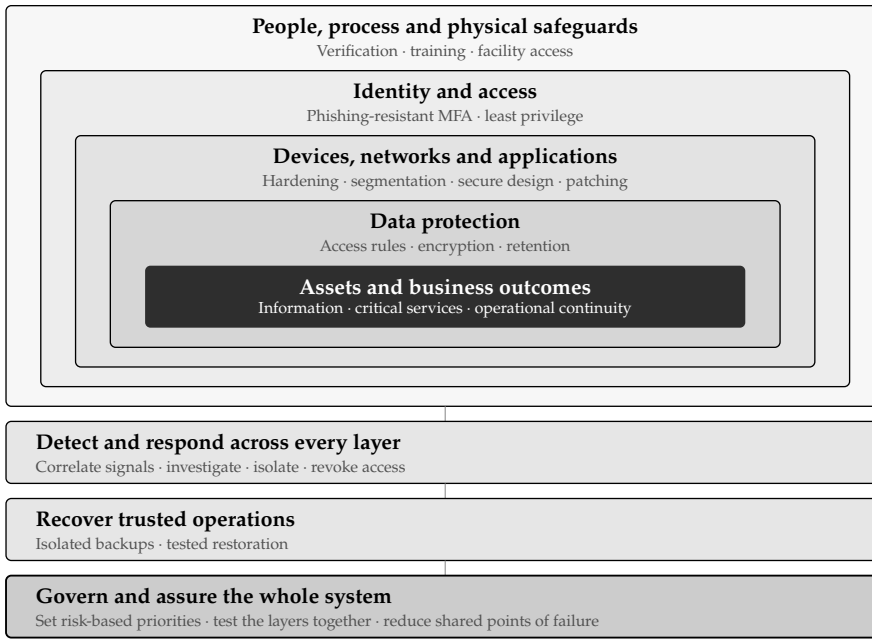


Figure 1.3 Defense in depth. The layers are drawn inward to the asset, because what is being protected fixes the arrangement — not what is attacking it. Detection, recovery and governance are not rings in the stack: they run across all of them. Illustrative layers; attack paths and control placement vary by organization.

Which brings us to why defense in depth is the most abused phrase in the field. "We have layers" is what an organization reaches for when it cannot demonstrate that any individual control works, and it is an argument only under two conditions that no control inventory records.

The first is that the layers fail independently. **Layers that share a failure mode are one layer wearing several names.** If five controls all depend on the same directory service, the same privileged credential, the same vendor agent, or the same overworked administrator, the depth exists on the diagram and not in the environment, because one failure takes all five at once. This is what the figure means by reducing shared points of failure, and it is the question a competent assessor asks that a checklist never does: not *are the layers present*, but *do they fail independently*?

The second is that somebody has checked. **An untested layer is not a layer; it is a hope with a budget line.** The backup that has never

been restored, the alert that has never fired, the segmentation rule nobody has tried to cross — each is counted as depth, and none of them has been shown to act. A layer's entire value is its behavior on the day something else fails, and behavior on that day is not a thing a configuration screenshot can report.

So the whole argument of this book arrives early, in miniature, inside a concept most readers already thought they understood. A checklist can count layers. Only a test can establish that the second one acts when the first one fails, and only an independent test can establish it without the answer being supplied by the people who built the thing. Defense in depth is not a claim about an architecture. It is a claim about behavior under failure — and the distance between those two kinds of claim is the assurance gap. Nearly every chapter that follows is an account of some organization discovering how wide it had become.

That is the objective, and the vocabulary for talking about it. The history comes next, and it starts further back than most security briefings do: with the room-sized machines of the 1950s, when the whole problem fit behind one locked door. The trip down memory lane is not nostalgia. Each era of computing solved the security problem of the one before it and created the next, and the pattern is easier to see from the beginning than from the middle. Once it is visible, today's challenges stop looking like a string of surprises and start looking like the latest turn of the same wheel.

The Fortress: Computing in the Mainframe Era

The earliest computers were, by modern standards, extraordinarily secure — not because their designers were security geniuses, but because the architecture left almost no room for mischief.

In the 1950s and 1960s, computing meant a mainframe: a single, room-sized machine managed by a small priesthood of operators. Users did not touch the hardware. They submitted jobs on punched cards or through dumb terminals wired directly to the machine. There was no network to intercept, no portable media to steal, and no software marketplace from which to download an infected application. Access was physical. If you could not walk into the data center, you could not reach the data. Figure 1.4 shows one such room in 1961.



Figure 1.4 The fortress. The IBM 7090 at NASA's Ames Research Center, photographed on 6 September 1961, soon after it was installed. Tape drives line the walls and the operator's console stands at the right; the whole system fills one room, and the door to that room was most of its security. NASA photograph, public domain.

Security in this world was a matter of locks, clearances, and operating procedures — governance, risk, and compliance (GRC) in its most literal form. The U.S. Department of Defense formalized much of this thinking in what became known as the "Orange Book" — the *Trusted Computer System Evaluation Criteria* (TCSEC), published in 1983 but rooted in concepts developed throughout the 1970s. The Orange Book defined levels of trust from D (minimal protection) through A1 (verified design), and it assumed a model in which a central authority controlled every aspect of the computing environment.[3]

That assumption held — until the environment itself changed.

Time-Sharing: The First Crack in the Fortress

Before the personal computer distributed computing to every desk, another revolution quietly redrew the security boundaries of the mainframe era.

Beginning in the early 1960s, researchers at MIT, Dartmouth, and other universities developed **time-sharing systems** — architectures that allowed dozens or even hundreds of users to share a single mainframe

simultaneously, each interacting through a terminal as if the machine were theirs alone. MIT's Compatible Time-Sharing System (CTSS, 1961) and its successor **Multics** (1965) were the most influential. Multics, in particular, was designed from the outset with security as a first-class concern: it introduced hierarchical access rings, access control lists, and the concept of a trusted computing base — ideas that would echo through every operating system built in its wake, from Unix to Windows NT.

But time-sharing also introduced a problem that the single-user mainframe had never faced: **multi-user conflict**. When dozens of users shared memory, storage, and processing cycles, one user's carelessness — or malice — could affect everyone else on the system. Password files had to be protected. Sessions had to be isolated. The question of *who is allowed to do what* moved from a physical gatekeeping problem to a software enforcement problem. And software, unlike a locked door, could be circumvented by anyone clever enough to find a flaw.

Time-sharing was, in effect, the rehearsal for the internet. It proved that shared computing was enormously productive — and it proved, equally, that shared environments demanded authentication, authorization, and accountability mechanisms that pure hardware controls could never provide. The lessons Multics taught were real, but they were learned by a small community of researchers and largely ignored when commercial computing took a very different path.

ARPANET and the Seeds of Connectivity

In 1969, the Advanced Research Projects Agency (ARPA), a branch of the U.S. Department of Defense, connected four university computers into a network called ARPANET. The goal was resilient communication: a network with no single point of failure, where data could route around damaged nodes. It was a military concept, but the researchers who built it were academics, and they designed it for collaboration and openness, not for security.

ARPANET's underlying protocol suite — which would eventually evolve into TCP/IP, the foundation of the modern internet — was built on trust. Packets traveled in the clear. Nodes accepted traffic at face value. Authentication was minimal. The architects were solving a connectivity problem among a small community of known, trusted

peers. The idea that millions of strangers — some of them adversaries — would one day share the same infrastructure was not part of the design specification.

This is one of the most consequential design decisions in technological history. The internet was engineered for availability and resilience, not for confidentiality or integrity. Every security technology developed in the decades since — firewalls, encryption, intrusion detection, zero trust architecture — is, at some level, a patch on top of a foundation that was never designed to be defended.

Creeper, Reaper, and the Birth of the Virus Concept (1971)

The first demonstration of self-propagating code came just two years after ARPANET's debut. In 1971, a researcher named Bob Thomas wrote a program called Creeper that moved between DEC PDP-10 computers running the Tenex operating system, printing the message: **"I'M THE CREEPER: CATCH ME IF YOU CAN."**

Creeper was not malicious. It was an experiment — proof that software could traverse a network autonomously. But Ray Tomlinson (the same man who had recently invented email) saw Creeper and modified it to be self-replicating: instead of simply moving from machine to machine, the new version copied itself, leaving instances behind on each host. Tomlinson then wrote Reaper — a program that hunted down Creeper copies and deleted them. In the space of a few months, the world had its first computer worm and its first antivirus program.[4]

Nobody panicked. The entire ARPANET community consisted of a few hundred researchers. But the conceptual precedent was set: if code could replicate across a network, it could be weaponized. And weaponized it would be — once the network grew large enough to make the effort worthwhile.

How the Models Shifted

Then IBM changed everything. Not deliberately. Not maliciously. Just profitably.

When IBM introduced the Personal Computer in 1981, and especially when the IBM PC/AT and its clones became affordable business

tools in the mid-1980s, computing migrated from the data center to the desktop. The mainframe priesthood lost its monopoly overnight. Suddenly every department, then every desk, then every briefcase had a machine — and that machine could run any software its owner chose to install, whether from a commercial vendor, a bulletin board system, or a floppy disk borrowed from the guy down the hall.

Understand what this meant for security. The mainframe model was a fortress with one gate and one gatekeeper. The PC model was a thousand unlocked doors spread across every floor of the building, each one controlled by someone who had never been told there were wolves outside. There was no central authority governing what ran on those machines. There was no patch management. There was no inventory of installed software. The genie of distributed computing was out of the bottle, and nobody — not the IT department, not the vendor, not the executive suite — had a framework for managing it.

Figure 1.5 puts the two models side by side. In the first, one machine does the application work, and processing, data, and control sit together under central IT. In the second, the work is spread across PCs, servers, phones, and the cloud, joined by networks, and control sits wherever someone decided to put it, or wherever it was left. Distributed computing can keep central control. The PC era, for the most part, did not.

Table 1.1 makes the same comparison aspect by aspect. The row that matters for this book is cybersecurity. The centralized model had few places to put controls and concentrated its risk in one room; the distributed model multiplied the endpoints, identities, and trust boundaries to secure, and most organizations inherited it without deciding who would verify the controls at each one.

The shift was not merely technological. It was organizational. The mainframe era had a clear chain of accountability: the data center manager controlled the machine, the security officer controlled access, and deviations were visible because the environment was small enough to see. The PC era shattered that chain. Accountability was distributed to hundreds or thousands of individuals who had no training in security, no incentive to prioritize it, and no understanding that their choices — what they installed, what they clicked, what they plugged in — affected everyone else on the network.

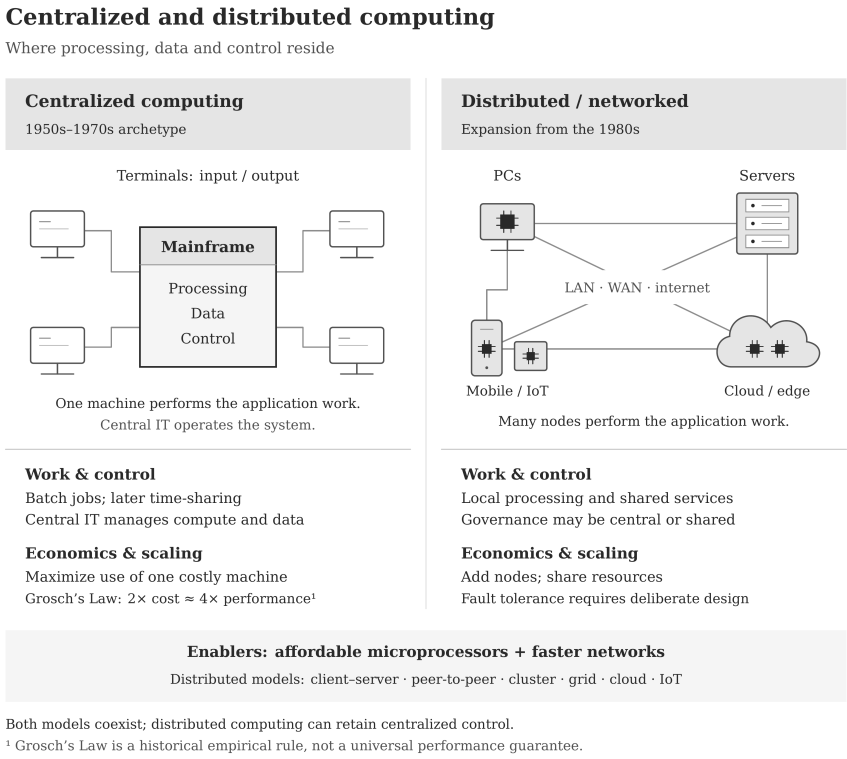


Figure 1.5 Centralized and distributed computing. In the mainframe model one machine holds the processing, the data and the control; in the distributed model many machines share the work, and control has to be designed in. Both models still coexist.

Organizations responded the way organizations usually do when confronted with a problem they do not understand: they bought products. Antivirus software. Firewalls. Intrusion detection systems. Each one addressed a symptom. None addressed the structural fracture underneath, which was that nobody owned the whole picture anymore.

When the Computer Became Entertainment

The PC's invasion of the office was disruptive enough. What came next was an order of magnitude worse.

By the mid-1990s, the personal computer had migrated from the workplace into the home. Families bought machines for homework, for games, for that strange new thing called the World Wide Web. America

Aspect	Centralized computing *Mainframe-centered archetype*	Distributed computing *Networked systems*
Architecture	One central mainframe	Many networked computers
User devices	Terminals with little or no local application processing	PCs, laptops, phones, and other capable endpoints
Processing	Concentrated on the central host	Shared across cooperating nodes
Data storage	Primarily central	Partitioned, replicated, or both
Control	Central administration	Centralized, shared, or decentralized
Cost	Large investment in shared capacity	Incremental capacity; network and coordination costs
Scalability	Scale up: expand central capacity	Scale out: add nodes where software supports it
Reliability	Shared failure risk unless protected by redundancy and recovery	Fault tolerance requires redundancy, failover, and recovery
Performance	High throughput; central resources can become bottlenecks	Parallelism and load sharing; network coordination adds overhead
Management	Concentrated administration; specialist skills often needed	Coordination across nodes, services, and data
Cybersecurity	Fewer control locations; concentrated risk	More endpoints, identities, and trust boundaries to secure
Example	IBM mainframe with terminals	Cloud platforms (AWS, Google Cloud), client-server, and peer-to-peer

Both models coexist. Reliability, security, and performance depend on design and operation.

Table 1.1 Centralized and distributed computing, side by side.

Online shipped tens of millions of installation discs. Dial-up modems screamed their handshakes in dens and kitchens across the country. By 1997, the internet had roughly 70 million users worldwide; by 2000 it had surpassed 400 million.[5]

Every one of those users was untrained. They had no understanding of network protocols, no awareness of what a phishing email looked like (the term did not even exist yet for most of them), and no reason to believe that the glowing box on the desk was anything more dangerous than a television with a keyboard. They reused passwords. They opened attachments from strangers. They installed free software that

came bundled with spyware. And increasingly, the machine that little Timmy used to play *Doom* in the evening was the same machine that Mom used to check the family bank account the next morning.

The attack surface did not just grow. It exploded. And the people standing on that surface had no idea they were standing on anything at all.

Worse, the entertainment model trained an entire generation to think of computers as consumer appliances — devices that should "just work" without the user having to think about configuration, updates, or risk. That expectation carried directly into the workplace. Employees who grew up clicking "Accept" on every dialog box and "Skip" on every update prompt brought those habits through the office door. They expected convenience. Security was someone else's problem.

It still is, in most organizations. And that expectation — that the machine should serve the user without the user having to understand or protect the machine — is one of the deepest roots of the assurance gap. It is also the seedbed of what the industry would later call *Shadow IT*: the sprawling, ungoverned ecosystem of applications, cloud services, and AI tools that employees adopt on their own because the approved path is too slow, too rigid, or too many clicks away. The consumer mindset that said "just install it" in 1997 now says "just sign up for the free tier" — and the security team is no more aware of it than the IT department was aware of the shareware on Dad's Compaq.

The Screen That Changed Jobs

Had you walked into an auto dealership's Finance and Insurance (F&I) office fifteen years ago, you would have seen a green-screen terminal connected to exactly one application: the dealer management system. The F&I manager processed credit applications. The shop floor had a parts terminal. The general manager had a reporting console. Each screen did one thing because it was wired to do one thing. The attack surface was the surface of the job.

That terminal is now a general-purpose PC running a web browser, an email client, a customer relationship platform, a lender portal, and whatever else the user decided to install during a slow afternoon. The same machine that processes credit applications — full Social Security numbers, income verification, bank account details — also streams

music, checks personal email, and provides a portal into the unfiltered internet. Personal banking belongs on the phone at lunch, not on the machine sitting two clicks from a credit bureau pull.

Guest wireless on the work floor is another conduit into a network that touches regulated data. If proxy tools — Websense, Cisco Umbrella, whatever the current flavor — show hours of nonwork browsing per day per user, that is not a culture problem. It is unbudgeted attack surface that nobody priced into the risk register.

Healthcare learned this lesson earlier, or at least the better wards did. The bedside terminal stayed locked to the electronic health record. Bored employees on a locked terminal stay bored. Bored employees on an open browser enlarge the attack surface of a system that contains protected health information.

The pattern is the same everywhere: a tool built for one purpose became a platform for every purpose, and the security model never caught up. The screen changed. The rules did not.

Only What the Job Requires

There is a principle in security as old as the mainframe era itself. It goes by several names: *least privilege*, *need to know*, *minimum necessary access*. The idea is simple: give every user, every application, and every process exactly the permissions required to do its job, and not one permission more. A payroll clerk does not need access to engineering schematics. A marketing intern does not need administrator rights on the database server. A web application does not need to execute system commands.

Every security framework on the planet endorses this principle. NIST enshrines it. ISO/IEC 27001 requires it. The CIS Controls list it in the top ten. It is taught in every introductory security course and appears on every certification exam. It is, in theory, universally accepted.

In practice, almost nobody follows it.

The reason is not technical. It is organizational. Enforcing least privilege means saying "no" to people — and in most corporate cultures, the IT department does not have the political capital to say "no" to a vice president who wants administrator rights on her laptop, or to a developer who insists he needs root access to a production server, or to

an entire department that has been sharing a single set of credentials for years because nobody ever set up individual accounts. Least privilege creates friction. Friction slows people down. Slowing people down costs money — or at least it *looks* like it costs money, right up until the moment a single compromised account with excessive privileges gives an attacker the keys to the kingdom. Chapter 14 says what those keys actually are and why the usual controls cannot make them safe to carry.

The pattern repeats at every scale. Organizations grant broad access because it is easier than managing granular permissions. They leave legacy accounts active because deprovisioning is tedious. They allow service accounts to accumulate privileges over years of incremental requests, each one reasonable in isolation, that collectively create a catastrophic blast radius.

This is not a technology failure. Least privilege is trivially implementable with modern identity and access management tools. The failure is one of governance — a failure of policy enforcement, a failure of accountability, and, above all, a failure of will. The tools exist. The knowledge exists. What does not exist, in most organizations, is the discipline to do what everyone already knows should be done.

The 1980s: Malice Goes Mainstream

The Brain Virus (1986)

In January 1986, two brothers in Lahore, Pakistan — Basit and Amjad Farooq Alvi — wrote what is widely considered the first virus for IBM PC-compatible computers. They called it **Brain**. The virus infected the boot sector of floppy disks and spread when users shared those disks. Remarkably, the Alvi brothers included their names, address, and phone number in the code; they later claimed the virus was intended as a copy-protection mechanism for their medical software. Whatever the intent, Brain demonstrated that self-replicating code could spread through the sneakernet — the hand-to-hand exchange of physical media — at a pace nobody had anticipated.[6]

The KGB Hack (1986)

That same year, a far more dangerous breach was unfolding. Markus Hess, a German hacker working on behalf of the KGB, penetrated an

internet gateway at Lawrence Berkeley National Laboratory and used it as a springboard into ARPANET. Over the course of months, he accessed approximately 400 military computers, including systems at the Pentagon, searching for classified information to sell to Soviet intelligence.

Hess's detection was almost accidental. Clifford Stoll, an astronomer managing computer systems at Berkeley, noticed a 75-cent accounting discrepancy and traced it to an unauthorized user. Over the next ten months, Stoll tracked Hess using an early form of what we would now call a honeypot — fabricated files designed to lure the intruder into spending enough time online to be traced. Stoll later documented the investigation in his book *The Cuckoo's Egg*, which remains one of the best first-person accounts of early cyberespionage ever written.[7]

The KGB hack demonstrated a truth that the cybersecurity community is still grappling with four decades later: detection is often accidental, attackers can operate for months without being noticed, and the compromise of a single trusted node can provide access to an entire network. Stoll found the intruder because he was curious about 75 cents. How many intrusions were never found because nobody looked?

The Morris Worm (1988)

On November 2, 1988, Robert Tappan Morris, a graduate student at Cornell University, released a worm onto the internet. His stated intent was to measure the network's size by infecting machines and having them report back. A flaw in the worm's code caused it to reinfect machines that were already compromised, consuming so much processing power that approximately 6,000 computers — roughly 10 percent of the internet at the time — were effectively disabled.

The Morris Worm was a watershed moment. It was the first worm to cause widespread disruption across the internet. Morris became the first person convicted under the Computer Fraud and Abuse Act (CFAA) of 1986. And the incident led directly to the creation of the **Computer Emergency Response Team Coordination Center (CERT/CC)** at Carnegie Mellon University — the first formal organization dedicated to coordinating responses to internet security incidents.[8] CERT/CC became the prototype for every national and organizational incident response team that followed.

The worm also exposed a fundamental vulnerability in networked systems: the *sendmail* utility it exploited was a standard component of Unix, present on virtually every internet-connected machine. A single vulnerability in widely deployed software could be leveraged to compromise the network at scale. This pattern — one flaw, thousands of victims — would repeat itself with devastating regularity in the decades to come. It is, in many ways, the defining pattern of cybersecurity history.

The 1990s: The Internet Goes Commercial, and Everything Changes

When the National Science Foundation opened the internet to commercial traffic in 1991, and Tim Berners-Lee's World Wide Web made it accessible to nontechnical users, the network's population exploded. With commercialization came e-commerce, online banking, customer databases, and intellectual property stored on servers accessible from anywhere in the world.

The security implications were staggering. Organizations that had never thought of themselves as technology companies were suddenly running mission-critical systems on internet-connected infrastructure. The attack surface — the total number of points where an unauthorized user could attempt to enter — grew by orders of magnitude every year.

The Rise — and Limits — of the Antivirus Industry

The antivirus (AV) industry emerged in the late 1980s and scaled rapidly through the 1990s. Early products worked by scanning files against a database of known "signatures" — unique byte patterns associated with specific viruses. If the scanner found a match, it quarantined or deleted the file.

This model worked reasonably well when the number of known viruses was measured in the hundreds. By the mid-1990s, the count was in the thousands. By 2007, AV-TEST was recording about 5.9 million new malware samples a year. By 2014, the figure was 143.1 million, roughly 390,000 new samples *every day*.[9]

The signature-based model was drowning. No team of analysts could write signatures fast enough to keep pace with adversaries who could generate new variants automatically. The antivirus industry

had created a necessary but fundamentally reactive defense — a game of perpetual catch-up that the defenders were slowly, inevitably losing.

And here is the part that matters for this book: organizations *felt* protected. They had antivirus software on every endpoint. The vendor dashboard showed green. The annual audit confirmed the software was installed. By every measurable compliance criterion, the control was "in place." But "in place" is not the same as "effective." The signature database was always a step behind. The scanner consumed resources that slowed machines, so users disabled it. The product caught known threats and missed novel ones. The checkbox was checked. The control was failing. Nobody verified the difference.

The 2000s: Organized Crime Discovers Cyberspace

The early 2000s saw a qualitative shift in the threat landscape. Hacking was no longer the province of curious teenagers and ideological hacktivists. Organized criminal enterprises recognized that digital crime offered enormous profit with relatively low risk of prosecution, especially when operating across national borders.

Botnets — networks of compromised computers controlled remotely by an attacker — became the infrastructure of cybercrime. A single botnet could consist of hundreds of thousands of infected machines, used to send spam, launch distributed denial-of-service (DDoS) attacks, harvest credentials, or mine cryptocurrency. The owners of the compromised machines often had no idea their computers had been conscripted.

Figure 1.6 traces the most visible of those uses, the distributed denial-of-service attack: the attacker issues commands through a command-and-control server, and thousands of hijacked computers, servers, routers, and cameras flood a single service at once. The target is never broken into; it simply runs out of capacity, and availability, the third property of the CIA triad, is what its legitimate users lose.

The botnets have only grown since. Cloudflare reported that the number of DDoS attacks it mitigated more than doubled in 2025, to 47.1 million, and that the largest peaked at 31.4 terabits per second, for just 35 seconds, launched from a botnet built mostly from infected Android TVs, somewhere between one and four million of them.[10]

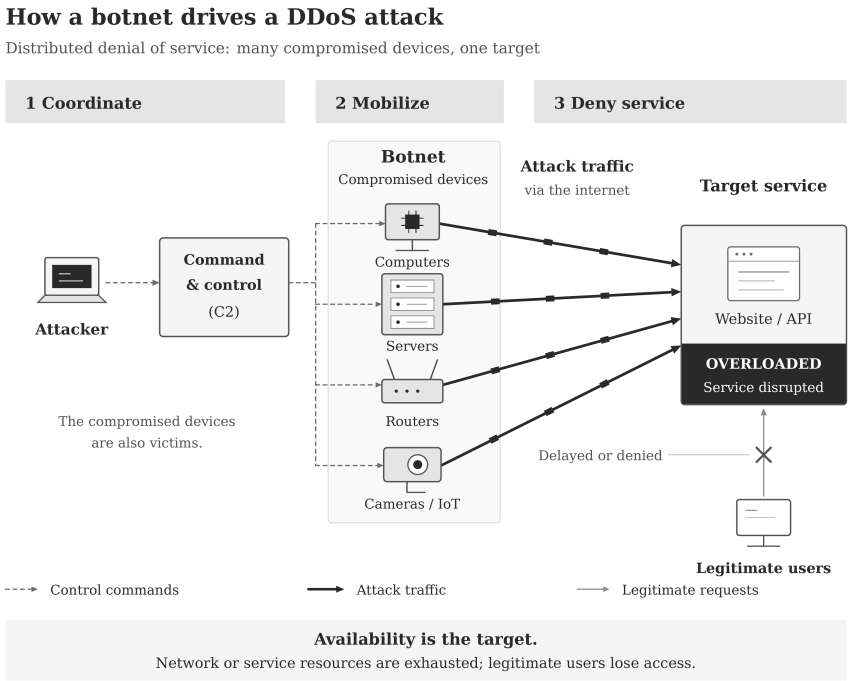


Figure 1.6 How a botnet drives a DDoS attack. The attacker commands the botnet through a command-and-control server; the compromised devices, themselves victims, flood the target until legitimate requests are delayed or denied. A simplified example: command-and-control architectures vary.

Phishing matured from crude, misspelled scams into sophisticated social engineering campaigns that could fool even security-conscious professionals. And the economics of cybercrime began to resemble those of any illicit market. Stolen credit card numbers, login credentials, personal health records, and corporate trade secrets were bought and sold on underground forums. Malware-as-a-service platforms allowed criminals with no technical skill to rent attack tools from developers who took a percentage of the proceeds. Cybercrime became an industry — complete with customer support, escrow services, and reputation systems.[11]

The professionalization of cybercrime did not just increase the volume of attacks. It changed their character. Individual hackers explored and vandalized. Criminal enterprises *optimized*. They calculated return on

investment, targeted the organizations least likely to have effective defenses, and automated every step of the kill chain that could be automated. The defenders were facing not a hobby but an economy — and economies scale.

Stuxnet (2010): Cyberwarfare Crosses the Rubicon

In 2010, security researchers discovered a worm unlike anything they had seen before. Called **Stuxnet**, it was extraordinarily complex — employing four zero-day vulnerabilities, stolen digital certificates, and rootkit techniques to hide its presence. It targeted a very specific industrial control system: the Siemens Step 7 software used to manage programmable logic controllers (PLCs).

Stuxnet's payload was designed to subtly alter the speed of centrifuges at Iran's Natanz uranium enrichment facility, causing them to tear themselves apart while it reported normal operations to monitoring systems. Stuxnet is widely attributed to a joint U.S.-Israeli operation.[12]

The significance of Stuxnet extends far beyond its immediate target. It demonstrated that cyberweapons could cause physical destruction — not by deleting files or stealing data, but by manipulating the industrial processes that the physical world depends on. Power grids, water treatment plants, transportation systems, manufacturing facilities — all of these run on the same categories of industrial control systems that Stuxnet compromised. A line had been crossed: cyberspace was no longer just a domain for espionage and theft. It was a theater of warfare.

And Stuxnet demonstrated something else, something more subtle and more troubling for our purposes: the systems it attacked *thought they were operating normally*. The centrifuges were destroying themselves, and the monitoring dashboards showed green. That is the purest expression of the assurance gap imaginable — the instrument designed to tell you whether something is working has itself been compromised, and nobody knows it until the damage is done.

The Ransomware Era: From Nuisance to National Security Crisis

Ransomware — malware that encrypts a victim's data and demands payment for the decryption key — existed in crude forms as early as 1989, when the AIDS Trojan was distributed on floppy disks at a World Health Organization conference. But ransomware did not become a dominant threat until the convergence of three enabling technologies: strong encryption, anonymous cryptocurrency payments, and widespread internet connectivity.

CryptoLocker (2013)

CryptoLocker was the first ransomware strain to use strong public-key cryptography that was effectively unbreakable without the attacker's private key. Distributed primarily through phishing emails and the Gameover Zeus botnet, it infected more than 234,000 computers and, by the Justice Department's account, extracted more than $27 million in ransom payments in its first two months, before law enforcement disrupted the operation in 2014.[13] CryptoLocker proved the business model: victims would pay, and the economics were overwhelmingly in the attacker's favor.

WannaCry (2017)

On May 12, 2017, a ransomware worm called **WannaCry** spread across the globe with shocking speed, infecting more than 230,000 computers in more than 150 countries within 24 hours. WannaCry exploited **EternalBlue**, an exploit for a vulnerability in Microsoft's Server Message Block (SMB) protocol that had been discovered — and stockpiled — by the U.S. National Security Agency (NSA). When the Shadow Brokers hacker group leaked the NSA's toolkit in April 2017, the exploit became available to anyone.

Microsoft had actually released a patch for the vulnerability before the attack. The organizations that were devastated — most famously the United Kingdom's National Health Service, which was forced to cancel 20,000 appointments and divert ambulances — were running unpatched systems.[14]

WannaCry crystallized a painful truth: the gap between *knowing* about a vulnerability and actually *fixing* it across an enterprise is where most

breaches live. Patching is not a technical problem. It is an organizational, procedural, and governance problem — an assurance problem.

NotPetya (2017)

Just six weeks after WannaCry, on June 27, 2017, a far more destructive attack struck. Initially disguised as ransomware, **NotPetya** was in fact a wiper — malware designed to destroy data, not to hold it for ransom. It spread using the same EternalBlue exploit, combined with credential-harvesting techniques, and it moved laterally through networks with devastating efficiency.

NotPetya was seeded through a compromised update to M.E.Doc, a tax accounting software package widely used in Ukraine. From there, it spread globally. Shipping giant Maersk lost access to all of its IT systems for ten days; the company estimated $300 million in damages. Pharmaceutical company Merck reported $870 million in losses. FedEx subsidiary TNT Express, Mondelez International, and dozens of other multinational corporations were crippled. Total estimated damages exceeded $10 billion, making NotPetya the most economically destructive cyberattack in history at the time.[15]

Double Extortion and the Colonial Pipeline

By 2020, ransomware operators had refined their tactics further. Instead of simply encrypting data, they began **exfiltrating** it first — stealing copies before locking the originals. Victims who refused to pay the ransom were threatened with public release of their data on "leak sites." This double-extortion model nullified the primary defense against ransomware (maintaining offline backups) and introduced regulatory risk: leaked data often included personally identifiable information subject to privacy laws.

The **Colonial Pipeline attack** in May 2021 brought ransomware to the level of a national security crisis. The DarkSide ransomware group compromised the pipeline's billing system, and the company preemptively shut down fuel distribution to the entire southeastern United States. Gas stations from Texas to New Jersey ran dry. The company paid a $4.4 million ransom within hours of the attack.[16]

Every element of this case is instructive. The initial compromise vector was a single exposed VPN credential — a problem that proper access management and multi-factor authentication would have prevented.

The company shut down *operations* because it could not trust its *billing systems* — a failure of segmentation. And the decision to pay was made by executives who had no independent visibility into which systems were compromised and which were not. They paid because they did not know. They did not know because nobody had built the assurance architecture that would have told them.

SolarWinds (2020): When Trust Itself Is Compromised

In December 2020, the cybersecurity firm FireEye disclosed that it had been breached by a sophisticated adversary. The investigation revealed something far worse: the attackers had compromised the software build system of SolarWinds, a company whose Orion network management platform was running at tens of thousands of sites; its compromised update reached up to about 18,000 of them, including multiple U.S. federal agencies, Fortune 500 companies, and cybersecurity firms themselves.

The attackers — later identified as Russia's SVR intelligence service — had inserted a backdoor (code-named **SUNBURST**) into a routine Orion software update.[17] Every organization that installed the update unknowingly invited the adversary inside. The backdoor was designed to lie dormant for twelve to fourteen days before activating, and it checked for the presence of specific security tools, exiting silently if it detected them. This level of patience and operational security suggested a nation-state operation with virtually unlimited resources.

SolarWinds exposed the fragility of **supply chain trust**. Organizations did everything "right" — they kept their software up to date, they applied vendor patches promptly — and that diligence was precisely what compromised them. The attack demonstrated that security does not end at your own perimeter. It extends through every vendor, partner, and service provider in your digital ecosystem, and a single poisoned link in that chain can expose thousands of downstream victims.

For the thesis of this book, SolarWinds is an especially instructive case. The affected organizations believed their controls were in place because audits said so and dashboards showed green. Nobody had independently verified, with evidence, that the controls actually worked. The vendor's own build pipeline — the most fundamental control

in software delivery — had been subverted, and neither the vendor nor its customers detected the compromise for months. That is not a technology failure. It is an **assurance failure** of the highest order.

SolarWinds was also the first of a series. Chapter 12 follows the same pattern through Log4Shell, Change Healthcare, the CrowdStrike outage, Salt Typhoon, the Bybit theft, and the help-desk attacks of 2025, and asks why organizations keep overlooking it.

Enemy Number One Is Not a Genius in a Basement

Ask someone outside the industry to picture a cyber threat, and they will describe a hooded figure in a dark room, fingers flying across a keyboard, breaching firewalls with arcane code. Where that picture came from is the subject of the next chapter. Whether it is accurate can be settled here, with evidence.

The most dangerous threat to any organization is not a genius. It is not a nation-state. It is not a zero-day exploit sold on a dark web marketplace. It is the employee who clicks the wrong link in an email, the administrator who misconfigures a cloud storage bucket, the contractor who reuses a password across 12 different systems, and the executive who approves access exceptions because the request came from someone he trusts.

The data on this point is not ambiguous. It is overwhelming.

The Verizon *2026 Data Breach Investigations Report* — the most widely cited empirical study of breach data in the industry, covering tens of thousands of security incidents in its 2026 edition — found that **62 percent of breaches involved the human element**: social engineering, misuse, or error.[18] Not sophisticated malware. Not novel exploits. People making mistakes or being manipulated into making them.

IBM's *2026 Cost of a Data Breach Report* confirmed the same picture from a financial angle. Of all breaches studied, **23 percent were caused by human error** — misdelivery, misconfiguration, lost devices, the mundane failures that nobody writes movies about. Another 16 percent started with phishing. Voice and SMS phishing attacks (vishing and smishing) carried the highest average cost of any attack vector: **$5.29 million per incident**. The global average cost of a data breach reached **$4.99 million**, but organizations that had deployed AI and

automation in their security operations spent **$1.93 million less** per breach than those that had not — evidence that the right tools, properly implemented and *verified*, make a measurable difference.[19]

The Ponemon Institute's *2026 Cost of Insider Risks* study, sponsored by DTEX, examined 7,490 insider-related incidents across 354 organizations and found an average annualized cost of **$19.5 million per organization**. The largest share — **$10.3 million** — came not from malicious insiders or credential thieves, but from **negligent employees**: people who failed to secure their devices, ignored policies, forgot to patch, or sent sensitive data to unsecured locations. Negligent insiders accounted for an average of 13.8 incidents per organization per year, at a cost of $747,107 each. And the clock is unforgiving: incidents contained in fewer than 30 days cost $14.2 million on average; those that dragged past 90 days cost $21.9 million. The average containment time was 67 days.[20]

And yet the most telling statistic of all may come from the training data. KnowBe4's *2025 Phishing by Industry Benchmarking Report*, analyzing 67.7 million simulated phishing tests across 14.5 million users in 62,460 organizations, found that **33.1 percent of untrained employees** clicked on a simulated phishing email or submitted their credentials when tested. One in three. Not because they were stupid. Not because they were careless. Because nobody had ever shown them what a phishing attack looks like, what to watch for, or what to do when something feels wrong.

After twelve months of regular training and simulations, that click rate dropped to approximately **4 to 5 percent** — a reduction of roughly 85 percent.[21] The KnowBe4 data proves two things simultaneously: that the human element is the largest vulnerability in any organization, and that it is also the most *improvable*. Training works. Awareness works. But only if someone actually delivers them, measures them, and holds the organization accountable for the results.

That last phrase — *holds the organization accountable for the results* — is the crux of the assurance gap. Every organization knows that phishing is a threat. Most have a security awareness program on paper. Far fewer measure its effectiveness. Fewer still tie those measurements to executive accountability. And almost none submit the entire chain — training completion, simulation results, incident correlation — to

independent verification. The control exists on the org chart. Whether it works is anyone's guess.

The enemy is not a genius in a basement. The enemy is the gap between what we know we should do and what we actually verify is being done. Every statistic in this section points to the same conclusion: **The problem is not that we lack the tools. The problem is that we lack the discipline to confirm the tools are working.**

That discipline — the governance of who watches, who verifies, and who is accountable for the evidence that controls are actually working — is the subject of this book.

The AI Era: The Next Transformation

As of this writing, artificial intelligence is reshaping both sides of the cybersecurity equation simultaneously.

On offense, attackers are using AI to generate more convincing phishing emails (eliminating the grammatical errors that once served as warning signs), to automate the discovery of vulnerabilities in software, to create deepfake audio and video for social engineering, and to adapt malware in real time to evade detection. AI lowers the barrier to entry: attacks that once required skilled programmers can now be orchestrated, using commercially available tools, by individuals with no technical background.

On defense, AI powers behavioral analysis engines that can detect anomalies across millions of events per second, correlate signals from disparate sources, and respond to threats in real time without waiting for a human analyst. Machine learning models have replaced static signature databases as the primary detection mechanism on modern endpoints. Security orchestration platforms use AI to automate incident response workflows, reducing the time from detection to containment from hours to seconds.

But AI also introduces new categories of risk. Models can be poisoned by adversaries who contaminate training data. AI-generated code may contain subtle vulnerabilities that traditional review processes miss. And the speed of AI-driven attacks may outpace human ability to understand and oversee defensive responses, creating a reliance on

systems whose decision-making processes are opaque even to their operators.[22]

The AI era reinforces a truth that has been constant since the days of ARPANET: technology alone is never the answer. Every advance in defensive capability is met by a corresponding advance in offensive capability. The asymmetry always favors the attacker, who needs to find only one way in, while the defender must cover every possible entry point. What tips the balance is not better tools, but better **assurance** — knowing, with evidence, that the tools and processes you have are actually working.

The Evolving — and Confusing — Lexicon

One of the most persistent obstacles to effective cybersecurity governance is the language itself. The field has grown so rapidly, and from so many different roots — military, academic, regulatory, commercial — that its vocabulary is riddled with overlapping terms, shifting definitions, and false synonyms.

Consider just the top-level terminology:

- **Cybersecurity** emphasizes the protection of digital systems and data in the context of cyberspace — networks, the internet, and connected devices.
- **Information security (InfoSec)** is broader, encompassing the protection of information in any form — digital, paper, verbal — against unauthorized access, use, disclosure, disruption, modification, or destruction.
- **IT security** is often used interchangeably with both but technically refers to the security of information technology infrastructure.

These are not the same thing, yet they are routinely used as if they were. An organization that conflates "cybersecurity" with "IT security" may protect its servers and firewalls while ignoring the business processes, third-party relationships, and human behaviors that represent its greatest vulnerabilities.

The confusion deepens at the operational level. Is a *vulnerability* the same as a *weakness*? (Not exactly — a vulnerability is exploitable; a weakness may not be.) Is a *threat* the same as a *risk*? (No — a threat is a potential cause of harm; a risk is the likelihood and impact of that harm

actually occurring.) What is the difference between a *control*, a *safeguard*, and a *countermeasure*? (Depends on which framework you are reading.) Does *compliance* mean the same thing as *security*? Emphatically not — but many organizations behave as though it does, which is a central theme of this book.

The proliferation of frameworks and standards has compounded the problem. NIST, ISO/IEC 27001, CIS Controls, COBIT, SOC 2, PCI DSS, HIPAA, CMMC, GDPR — each introduces its own vocabulary, its own taxonomies, and its own way of categorizing controls. A "control family" in NIST maps imperfectly to a "clause" in ISO/IEC 27001, which maps imperfectly to a "domain" in COBIT. Organizations operating under multiple regulatory regimes must navigate these overlapping lexicons simultaneously, often with no Rosetta Stone to translate between them.

This is not a trivial problem. Language shapes thought, and imprecise language produces imprecise thinking. When a board member asks, "Are we secure?" the answer depends entirely on what "secure" means in that context — and if the chief information security officer's (CISO's) definition, the auditor's definition, and the board member's definition are all different, the resulting conversation is theater, not governance.

Establishing a **consistent nomenclature** — a shared taxonomy with precise, agreed-upon definitions — is not an academic exercise. It is a prerequisite for meaningful communication about risk, for accurate assessment of controls, and for genuine assurance. Without a common language, there can be no common understanding. And without common understanding, there can be no accountability.

Beyond the general terminology, several specific concepts recur throughout this book and deserve precise definition here:

Assurance vs. Compliance. Compliance is a snapshot — evidence that controls existed on the day an auditor checked. Assurance is a continuous process — evidence that controls are present, adequate, functioning, and maintained on any given day. This book argues the gap between them is where breaches live.

Control Drift. The gradual degradation of a security control's effectiveness between assessments. A firewall rule that was correct on audit day may be modified, disabled, or bypassed within weeks. Control drift is the engine of the assurance gap.

Cyber Assurance Officer (CAO). An independent role, separate from both the operational IT/managed service provider (MSP) function and the security monitoring/managed detection and response (MDR) function, whose mandate is to verify that cybersecurity controls match the organization's claims. The institutional fix this book proposes.

Red Flags Rule. The Federal Trade Commission's (FTC) identity-theft detection rule (16 CFR Part 681, implementing FCRA §615(e) as narrowed by the Red Flag Program Clarification Act of 2010) applicable to financial institutions and creditors under FTC jurisdiction — including, notably, automobile dealers who finance or lease vehicles and regularly obtain consumer reports for those transactions. Requires written programs to detect, prevent, and mitigate identity theft.

Safeguards Rule. The FTC's rule requiring financial institutions (including auto dealers, mortgage brokers, and tax preparers) to develop, implement, and maintain a comprehensive information security program. Substantially strengthened in 2023.

Purdue Model (PERA). The Purdue Enterprise Reference Architecture — a hierarchy (Levels 0–5) defining where industrial and enterprise systems live within a manufacturing or operational technology environment. It describes layout, not security controls.

ISA/IEC 62443 (Zones and Conduits). The international standard family for industrial automation and control system (IACS) security. Whereas the Purdue Model tells you where a system sits, IEC 62443 tells you how hard the door should be between zones. They are complementary, not interchangeable.

Navy-Marine Corps Intranet (NMCI). The U.S. Navy's enterprise IT consolidation program (c. 2000), which replaced hundreds of independent LANs with a single managed network. Relevant as a case study in the tension between operational control and user resistance to standardization.

This book adopts a deliberate approach to terminology. Key terms are defined where they first appear and used consistently throughout. Where industry terminology is ambiguous, the book states the specific meaning intended. The goal is not to settle every definitional debate in the field, but to ensure that within these pages — and within the governance frameworks they describe — words mean what they say.

The Pattern That Never Changes

Step back from the individual events, and a single, unbroken pattern emerges across five decades of cybersecurity history:

1. **A new technology connects more systems and more people.** (ARPANET, the PC, the commercial internet, cloud computing, the Internet of Things, AI.)
2. **The new connectivity creates new attack surfaces that were not anticipated in the original design.**
3. **Adversaries exploit those surfaces faster than defenders can secure them.**
4. **The security industry responds with new tools** — antivirus, firewalls, intrusion detection, endpoint detection and response, extended detection and response, security orchestration.
5. **Organizations adopt the tools and assume they are protected.** Dashboards turn green. Audits pass. Compliance checkboxes are checked.
6. **A catastrophic breach reveals that the tools alone were not enough** — that the real failure was not technological but organizational: nobody verified, with independent evidence, that the controls were actually operating as promised.

This is the cycle. It has repeated with every generation of technology, and it will repeat with AI and whatever comes after AI. The technology changes. The adversaries adapt. The tools evolve. But the fundamental gap — the **assurance gap** — remains.

The tools are not the problem. The people are not the problem — or rather, people are a *known* problem with *known* solutions, as the training data in this chapter demonstrates. The problem is that nobody checks. Nobody verifies. Nobody submits the entire program — tools, people, processes, and evidence — to independent scrutiny and demands proof that it works.

Closing that gap is the subject of the rest of this book.

What to Remember, What to Do, What to Ask

Remember one thing: The pattern has not changed in fifty years — a new technology creates a new surface, defenses lag, and the gap between what is claimed and what is real becomes the breach vector.

Do one thing this quarter: Ask your CISO, or whoever leads security, to define "assurance" in one sentence. If the answer sounds like "compliance," the rest of this book is about the difference.

Ask one question in the next leadership meeting: "If we removed every product purchased in the last three years, which of our original controls would still be operating correctly?"

Chapter 1 Notes

1. NIST, *Security and Privacy Controls for Information Systems and Organizations*, NIST SP 800-53 Rev. 5 (September 2020), glossary, s.v. "defense in depth." NIST's publications for industrial control systems carry a second definition, drawn from IEC 62443, that states the mechanism rather than the strategy: layering *heterogeneous* security technologies so that an attack missed by one is caught by another (NISTIR 8183). "Heterogeneous" is the operative word there, and the reason shared failure modes collapse depth back into a single layer.
2. CISA, *Implementing Phishing-Resistant MFA* (fact sheet, October 2022), which calls phishing-resistant MFA "the gold standard for MFA" and notes that the Office of Management and Budget requires federal agencies to adopt phishing-resistant methods. Not every form of multi-factor authentication is phishing-resistant: a one-time code, whether delivered by text message or generated by an authenticator app, can be captured and replayed by the same fraudulent site that captured the password.
3. U.S. Department of Defense, *Trusted Computer System Evaluation Criteria* (DoD 5200.28-STD), National Computer Security Center, 1985. Commonly known as the "Orange Book," the TCSEC established the first systematic evaluation framework for computer security, defining classes D through A1.
4. Thomas's Creeper and Tomlinson's Reaper are documented in numerous histories of computing. See, e.g., Yost, J. R., "The Origin and Early History of the Computer Security Software Products Industry," *IEEE Annals of the History of Computing* 37, no. 2 (2015): 46–58.
5. Internet World Stats and International Telecommunication Union (ITU) historical estimates. By year-end 1997, global internet users were estimated at approximately 70 million; by 2000 the figure had exceeded 400 million, driven largely by consumer broadband adoption in North America and Europe.
6. Ferbrache, D., *A Pathology of Computer Viruses* (Springer-Verlag, 1992), 37–42. Brain is widely cited as the first IBM PC-compatible virus; the Alvi brothers confirmed its origin and intent in multiple subsequent interviews.
7. Stoll, C., *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage* (Doubleday, 1989). Stoll's account of tracking Markus Hess remains a foundational text in incident response and forensic methodology.

8. The CERT/CC was established at Carnegie Mellon University's Software Engineering Institute in November 1988, directly in response to the Morris Worm. See Spafford, E. H., "The Internet Worm Program: An Analysis," *Purdue University Technical Report CSD-TR-823*, 1988.
9. AV-TEST Institute, which registers over 450,000 new malware samples and potentially unwanted applications per day as of 2024. Historical growth figures from AV-TEST's annual malware statistics archive.
10. Cloudflare, "2025 Q4 DDoS threat report: A record-setting 31.4 Tbps attack caps a year of massive DDoS assaults," Cloudflare blog, February 5, 2026. Cloudflare calls the botnet Aisuru-Kimwolf and estimates it at one to four million infected hosts, primarily Android TVs.
11. For an overview of the professionalization of cybercrime, see Lusthaus, J., *Industry of Anonymity: Inside the Business of Cybercrime* (Harvard University Press, 2018).
12. Zetter, K., *Countdown to Zero Day: Stuxnet and the Launch of the World's First Digital Weapon* (Crown, 2014). The most comprehensive public account of Stuxnet's development and deployment.
13. U.S. Department of Justice, "U.S. Leads Multi-National Action Against 'Gameover Zeus' Botnet and 'Cryptolocker' Ransomware," press release, June 2, 2014.
14. NHS England, "Lessons Learned Review of the WannaCry Ransomware Cyber Attack," 2018. Microsoft Security Response Center, MS17-010 bulletin, March 14, 2017.
15. Greenberg, A., *Sandworm: A New Era of Cyberwar and the Hunt for the Kremlin's Most Dangerous Hackers* (Doubleday, 2019). NotPetya damage estimates compiled from Maersk, Merck, and FedEx SEC filings and public statements.
16. Turton, W. and Mehrotra, K., "Colonial Pipeline Paid Hackers Nearly $5 Million in Ransom," Bloomberg, May 13, 2021. Colonial Pipeline CEO Joseph Blount confirmed the payment in subsequent congressional testimony.
17. FireEye (now Mandiant/Trellix), "Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims Via SUNBURST Backdoor," December 13, 2020. U.S. government attribution to SVR: White House statement, April 15, 2021.
18. Verizon, *2026 Data Breach Investigations Report* (DBIR), 2026. The human-element figure of 62 percent encompasses social engineering, privilege misuse, and human error across more than 22,000 confirmed data breaches. The report also found that exploitation of vulnerabilities accounted for 31 percent of initial access vectors, surpassing credential theft for the first time.
19. IBM Security, *Cost of a Data Breach Report 2026*, conducted by Ponemon Institute, 2026. Key findings: global average breach cost $4.99 million; human error root cause 23 percent; phishing as initial vector 16 percent;

voice/SMS phishing (vishing/smishing) average cost $5.29 million; organizations with AI and automation in security operations averaged $1.93 million less per breach than those without.

20. Ponemon Institute, *2026 Cost of Insider Risks: Global*, sponsored by DTEX Systems, 2026. Seventh annual benchmark study; 354 organizations, 7,490 insider-related incidents. Average annualized cost $19.5 million per organization. Negligent insider incidents averaged 13.8 per organization at $747,107 each ($10.3 million total, up 17 percent year over year). Average containment time 67 days; incidents contained in fewer than 30 days averaged $14.2 million vs. $21.9 million for those exceeding 90 days. Credential theft incidents averaged $842,462 each.
21. KnowBe4, *Phishing by Industry Benchmarking Report 2025*, 2025. Study analyzed 67.7 million simulated phishing tests across 14.5 million users in 62,460 organizations worldwide. Global baseline Phish-prone Percentage (PPP) for untrained employees: 33.1 percent. After 12 months of regular simulated phishing and security awareness training, average PPP dropped to approximately 4 to 5 percent, representing a reduction of roughly 85 percent.
22. For a synthesis of AI-related cybersecurity risks, see National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," NIST AI 100-1, January 2023; and European Union Agency for Cybersecurity (ENISA), *AI Cybersecurity Challenges*, 2020.

Continue reading

Chapter 1 ends with the pattern that never changes. The twenty chapters that follow show what it costs — audits passed before the breach, insurance claims denied, accountability that belonged to no one — and set out the fix: six pillars of cyber assurance, continuous evidence in place of annual attestation, the independent Cyber Assurance Officer, and a 90-day plan to begin.

Paperback on Amazon

Kindle edition

399 pages · 21 chapters · two appendices, glossary and index. Also free to read with Kindle+.

Scan for the paperback

About the author

William F. Leach Jr. is a cybersecurity executive and practicing virtual CISO with more than 26 years in cybersecurity and information assurance: CISO and CIO of a publicly traded bank, CISO of a healthcare quality-improvement organization, Vice President of Cybersecurity at a national managed security services provider, and Information Assurance Manager for more than 600 Navy Reserve centers. A 20-year Navy veteran, he retired as a commanding officer, founded Transformyx Technology Services, and has held the CISSP since 2004.

He speaks to ISACA, ISC2 and IIA chapters, security and audit conferences, and boards on the assurance gap and the Cyber Assurance Officer role. Sessions and a bio for programs are at williamleachjr.com.

Author site	williamleachjr.com
Firm	tfmxts.com
LinkedIn	linkedin.com/in/williamleachjr
Amazon	Author page

The Assurance Gap: Why Cybersecurity Compliance Is Broken — and What It Takes to Fix It. Transformyx Technology Services, 2026. ISBN 979-8-234-24281-5.