How to CISO

Volume 2

Risk

Version 1.0

By Andy Ellis

Introduction

As a CISO, you're often going to be asked to *measure risk*. You might think you're in the job of *mitigating risk*. You might think of it as *managing risk*. These phrases have a lot of different meanings, depending on who is speaking, so you're going to have to listen carefully to the speaker to understand what they're actually asking for. It's possible that you're being asked to provide a quantitative answer to the oft-asked (and very dangerously misleading) question, "Are we safe?" Or the question might be a variant of, "How do you know we're prioritizing the right things? Which problems are the *riskiest*?" Or possibly you're being asked for some measure of security efficacy - some KPIs to justify what you're spending your resources on.

Whichever *risk* phrase you're engaging in, you're actually in the job of *risk enablement*. Your employer exists to *take risks* in service of their objective (usually, making money), and you should be focused on guiding them to *wiser risk decisions*. Managing risk is about finding ways to prioritize away from dangerous, unprofitable risk choices (risk mitigation!) and towards safer, more profitable choices.

Enabling wiser risk choices requires that a CISO be an exquisite risk communicator.

About This Guide

The *How to CISO: Risk* guide is split into three sections - each stands on its own, so you can read them in any order you'd prefer. The first section will address the elephant in the room: understanding *why* we bother to measure risk at all, and exploring how to understand what you actually need to achieve your objective: wiser risk choices in your organization. The second aims to provide some alternate vocabulary to more precisely communicate about *risk*. If you want to skip and jump to later sections, and use this as a reference, go ahead. *You* don't need to use this language in your organization (if you plan to, I recommend reading the section on *Cultural Language* in the intro of 1% Leadership), but this guide will be consistent, to attempt to avoid the miscommunication and vagueness around risk language. The third and final section covers various ways that organizations tend to measure and score risk. All of them have weaknesses, but some of them are very good at communicating specific forms of risk to specific stakeholders. This section will also cover and address those weaknesses, so you understand exactly how you might end up in trouble with certain risk measurement strategies.

About How To CISO

How to CISO is a collection of security practitioner guides, intended to help current and aspiring CISOs, as well as their teams, to improve their security practice. *Volumes* are stand-alone, longer-form guides intended to deeply explore a specific topic, whether it's *The First 91 Days* of a CISO's role after a job change of *The Idealized CISO Job Description*. *Handbooks* are brief summaries of a topic area, discussing concepts like *Zero Trust* or the various *Environments* that a CISO will deal with. You can find the collection of How To CISO guides, as well as talks that have discussed these principles, at www.howtociso.com.

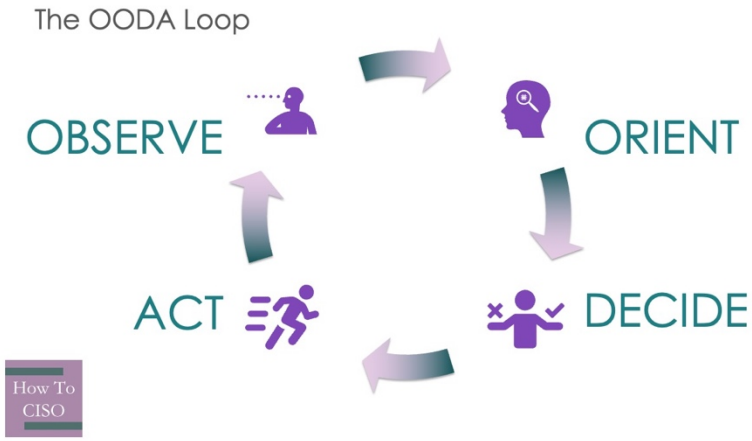
Within this guide, as well as other *How to CISO* guides, the term *CISO* is used interchangeably with "security leader" or "security decision maker." You don't have to be a CISO to engage in the practices you'll learn in this guide.

About the Author

Andy Ellis was inducted into the CSO Hall of Fame in 2021, so he should know something about being a CISO. He also wrote a book (1% Leadership), which has some small relevance in building teams and organizations. He is the Editor at How to CISO, Principal at Duha, cohost of The CISO Series Podcast, and dabbles with a dozen other part-time roles.

Talking About Risk

We *communicate risk* for two very similar goals: to convince someone to change course, or to convince them to *not* change course (we usually forget about this goal). For those of you that use the OODA Loop as a model for human decision-making, we're trying to change or reinforce the model that decisionmakers use to *orient* themselves.



One of the biggest errors that risk managers (whether they are CISOs or deeper in the organization) often make is to try to use the risk evaluation as a decision-*forcing* vehicle: that whatever the risk equation says will dictate what the business will do. Unless your business is actually bought into that model ahead of time, trying to force a decision simply because your model says to do so? That's a quick way to being driven into irrelevance, as your business counterparts will learn to *not* listen to your model (or you).

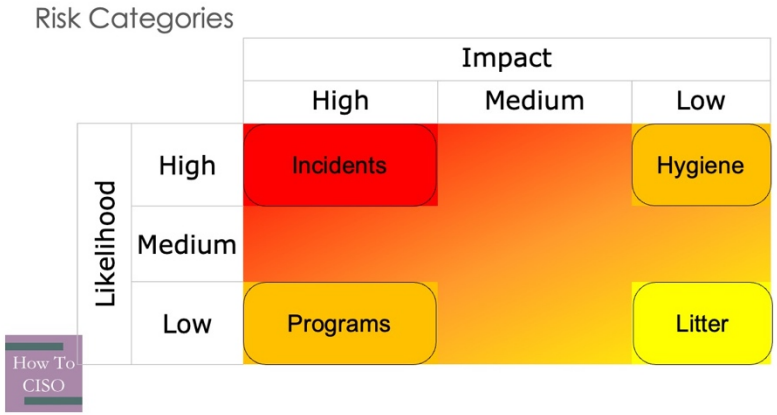
Whether we're trying to change or keep course, there are multiple types of actions a CISO might be trying to influence, from quick actions to long projects, small investments to large outlays. A CISO should use consistent language in talking about risk, but may use different tactics.

Categories of Risk

There are four fundamentally different categories of risk remediation, and an astute CISO should recognize that the language used for each is going to be slightly different, and that even if your counterparts aren't *explicitly* thinking about these separate categories, often they are *implicitly* considering them. These categories aren't defined by the bad outcomes as much as they are defined by the processes and means that management is going to interact with them.

At one end of the spectrum are *incidents*: really bad outcomes that are really likely (or already happening). *Incidents* are usually obvious to everyone involved: if they're already happening, then the damage is obvious; if they haven't yet happened, everyone expects them to happen in the near future. Rarely do CISOs need to argue that something should be done during an incident. It's more a conversation about how much needs to be done and how quickly. *Nascent* incidents may present a communications challenge: if it hasn't yet happened, but seems likely to happen, an organization needs to operate at incident tempo without having a clear fire to put out.

At the opposite end of the spectrum is *litter*: the really small, rare problems that are littered through your environment. Like trash on the street, it's rare that anyone wants to talk about specific pieces of litter, a CVSS 1.2 vulnerability might be *real*, but its existence on its own is only really relevant to a specific developer. Instead, CISOs



need to talk about the streetsweeping process: whether the litter as a whole is being dealt with in a timely manner.

In the middle of the spectrum are two opposites. *Hygiene*, to deal with frequent, low-damage problems, and *programs*, to deal with the improbable-but-dangerous hazards in the business. It can be tempting to use the same language to group these two categories together, but this can often be dangerous. Consider the difficulty in trying to prioritize between brushing your teeth on any given day (a hygiene item) and procuring a stockpile of iodine tablets (an apocalypse preparedness program). Similarly, executives will have challenges flipping between these two categories.

Avoid simplifying this to "High-Medium-Low". While High and Low fit naturally onto Incidents and Litter, the confusion you'll create with Medium encompassing both Hygiene and Programs isn't worth it.

Risk Appetite vs. Mitigation Appetite

Organizations often struggle with the question of "how much risk is the company willing to take on?" For some companies, the answer may feel unbounded: for the right amount of upside, almost any risk seems worth it. Risk managers often try to pin down the very slippery idea of *risk appetite*, which enterprise leaders have a hard time qualifying *or* quantifying.

There is an implicit third axis on the prior risk categories chart: cost of mitigation. If the cost to mitigate deviates significantly from the perceived risk, risk decisions are a *no-brainer*. Either it's way too expensive to bother with, or it's so cheap that it's an obvious decision to just solve it. Functionally, organizations have a *mitigation appetite*: how much they're willing to spend on mitigating risks in existing systems.

Decision-Making and Outcomes

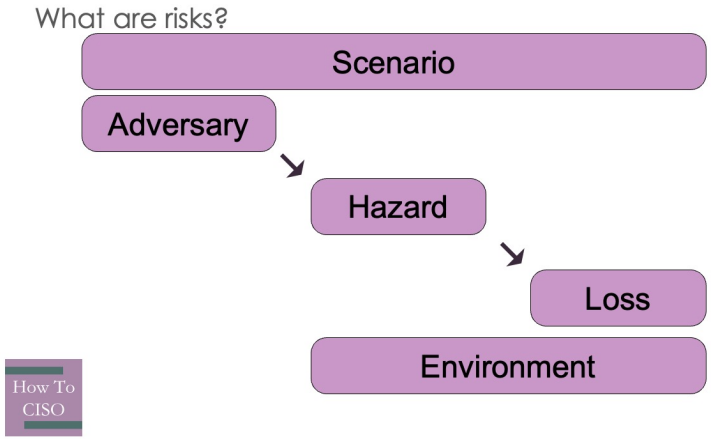
The goal of risk conversations is *not* to drive to a specific outcome. It is the rare CISO who owns a decision about an area so thoroughly that they can unilaterally decide how the business will adapt in major ways. Rather, the risk conversation needs to make the decision makers around the business truly *believe* that the risk is real, relevant, and something that *they* need to deal with. This may require multiple conversations across several risks to change their perspective; don't be fixated on getting an optimal outcome on each individual risk.

Why does this matter? Because humans defy description as rational decision makers. Rather, they are natural decision makers, more often acting on their gut beliefs and instincts than strictly abiding by a formulaic decision. If you'd like to study this topic more, you can find a few talks on the topic here.

Risk Vocabulary: A Taxonomy

You Can't Define "A" Risk

Let me open by laying one of my biases out on the table. *Risk* is a dangerous word. It means so many things to so many people, and you really can't pin it down with a single definition across a diverse audience.



To some, risks are *hazards* – vulnerabilities that are present in a complex system that an adversary, with or without intent, could exploit or trigger to cause you harm. To others, risks are *scenarios*, encompassing both a set of hazards and the exploit or trigger (so a single hazard risk maps to multiple risk scenarios). Maybe someone views risks as *adversaries*, the triggers that cause harm to come to the company. Risk might apply to an *environment*, a collection of systems that have an aggregate risk. For some, risk might be seen as a *cost*, the downside to a plan for the future. Every executive you interact with has a different definition of risk, and it's near impossible to teach them to use *your* definition.

This leaves you with at least one viable option: stop using the word "risk" to describe specific things. Use specific terms like adversaries, hazards, losses, and scenarios to make things clearer, and avoid energy-wasting arguments about the specific definition of the word "risk." The rest of this section is a taxonomy of (many of) the ways and places that organizations tend to talk about risk, with a description of what types of issues you might want them to think about.

Casper Weinberger, former United States Secretary of Defense, once quipped that the hardest part of his job was using the right words to tell people what to do. If he said to secure a building? The Marine Corps would kill everyone inside, blowing up the building for good measure. The Army would requisition sandbags and concertina wire, setting up a perimeter with zones of enfilading fire. The Navy would make sure to turn off all the lights and latch the doors. And the Air Force would take out a three year lease with an option to buy (I wonder what he'd say the Space Force would do – task a satellite to monitor entry and exit?).

Risky Outcomes (a.k.a. Losses)

Losses, the category of risks that describe *outcomes* that we might want to avoid, is a key aspect in any conversation about risk. After all, having weak passwords to access a semi-public wifi is much less of a problem than having weak passwords that access your datastore of all of your customers' personal information. The difference between these two scenarios is the severity of the outcome; and many executives are going to drive conversations about risk by focusing on that outcome, and ensuring that there are *no* possible ways for that outcome to occur. Perfect security like that can be a fool's errand. When we talk about losses, it's important to think about the difference between *partial* or *interim* losses and *end-state* losses. An outage might be an *interim* loss that you can easily recover from, while a data breach is an *end-state* loss (which might be a *partial* loss if the data was encrypted; you still have to do breach notification, but it's not quite as bad as if the cleartext data was lost).

Unacceptable Losses

Unacceptable loss is a phrase borrowed from the complex systems safety world. It's a phrase that has clear communicative value, as it describes a set of end-states that organizations need to avoid. In a sense, it's a replacement for *asset-oriented* risk management, and you can often begin inventorying unacceptable impacts by asking "would it be awful if an adversary destroyed, stole, or disclosed this asset and its contents?" That is rarely going to give a complete loss inventory for most organizations. Consider an airline. While you can easily move from its assets to risks (compute-based operations failing, for instance), the greatest unacceptable loss to an airline is *the loss of passenger lives*. It's so ingrained in the travel industry that loss of lives is measured in *souls*. Yet you wouldn't consider passenger lives to be an asset of the airline.

Focusing on unacceptable losses as the root of risk management helps focus conversations on a business risk that everyone can understand.

Incidents

An *incident* is an instantiation of a *scenario* (don't worry, we'll get to those in a bit), which resulted in a loss. Incidents are a double-edged sword. On one hand, they often open the eyes of various parts of the businesses to real risks, as they observe a loss play out in front of them. On the other hand, they are simpler than the complex web of ways that a problem could happen. It's often easier for an executive to tackle the apparent "root cause" – really, the trigger – of an incident than the underlying hazards that made it possible.

Incidents are very useful in thinking strategically about risk because most companies have an explicit scoring system for damage. Incidents usually have *severities* – a scoring system, perhaps from "Sev 1" at the worst to "Sev 4" at the most trivial – which normalize a measurement of harm across a number of different axes. Incident severity can be considered in terms of the cost a company is willing to *pay* to get out of an incident. Low severity incidents don't come with much tolerance for added business disruption,

while a critical incident mobilizes senior executives and nigh unlimited real-time spending.
Losses, in the incident world, are often called *impacts*: the specific loss incurred by an incident or scenario.

Specific Losses

Often, we refer to a *breach* in describing a specific incident that resulted in an unacceptable loss, but "breach" is often an ill-defined term, beyond "an adversary violated our defenses or system boundaries." It's helpful to look at ways that a specific loss might affect an organization.

System Compromises

A *system compromise* can be a poor outcome on its own – the PR implications of loss of control of a system are problematic – but usually the compromise is a stepping stone in a worse scenario. An adversary with control of a system gains additional *powers* to trigger unacceptable losses – a database that wasn't internet-facing is now adversary-facing, for instance – but rarely is a system compromise, by itself, the unacceptable loss for an organization. It just increases the risk of another loss occurring.

Becoming part of a *botnet* may be the next step of a system compromise, either to conduct DDoS attacks, mine crypto, or otherwise use system resources directly for an adversary's benefit. These aren't exclusive with an adversary using a system compromise for specific, targeted attacks; a wise adversary who committed a data breach might choose to cover their tracks by installing a cryptominer to misdirect an incident response team.

Data Risks

Data Risks come in four major flavors: *data breach*, *data leakage*, *data tampering,* and *data loss.* A data breach is when an adversary gains access to your data, often in large quantities, and takes a copy of it for some use. Data leakage occurs when your organization inadvertently publishes data, often through a third party relationship. Data tampering happens after an adversary injects or alters data into your systems, such that your data stores are no longer reliable. And data loss is what happens when your data is no longer accessible to you.

Adversary Control

Sometimes, an adversary gains some *control* over part of your systems. Perhaps they can issue commands as an end-user (unlocking car doors or opening garage doors), or suppress security indicators, or engage in administrator actions (adding new users). Adversary control can be a more productive way to talk about *insider threats* – instead of talking about your employees doing something harmful, ask how an adversary who had the same privileges (perhaps by controlling an employee's system) could use those privileges to cause harm.

Downside Risk

In the non-cyber part of our businesses, *downside risk* is a routine part of the conversation. Consider it to be the known, possible, *acceptable* losses built into a plan. It's

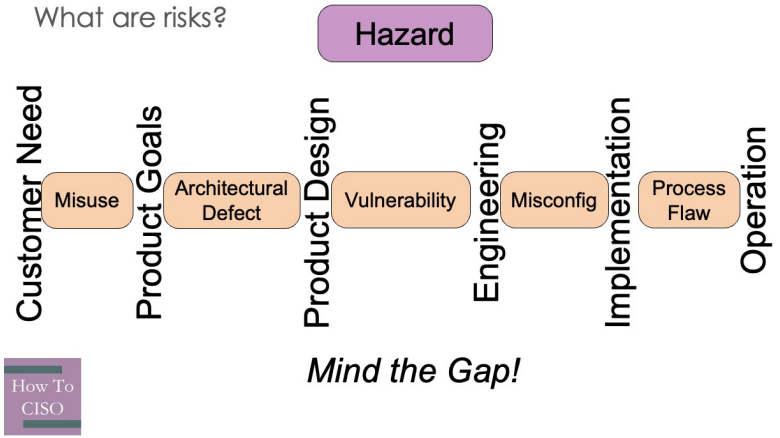
often matched with an *upside*: perhaps the revenue forecast for the year is $.95B-1.05B, but there is upside of $1.10B if at least three new products succeed wildly, and downside of $.9B if only one new product succeeds, or if foreign exchange rates shift.
In discussing risk in the cybersecurity arena, we should also consider the downside risks. Maybe we're seeing an incident rate of one major incident per week; that could easily go to two per week and still be within an acceptable norm in many enterprises.

Hallucinations

The prevalence of AI has brought forth a new risk: *hallucinations*. While we worry about the brand implication of a rogue employee saying something that puts the brand in trouble, the implications when an unfiltered AI tells our customers or the public to do something that we'd prefer they didn't is another unacceptable loss to keep our eye on.

Risky Items (a.k.a. Hazards)

Most of the time, when someone uses risk as a noun ("a risk"), what they're talking about is a *hazard*: a danger present in the environment which helps create a path to an unacceptable outcome. Hazards can often be considered as a gap between what you expect, and what you were given. This gap-focused definition yields different names depending on where the gap occurs.



Vulnerabilities and Misconfigurations

Probably the best known category of hazards is *vulnerabilities*, including both software defects and system misconfigurations. We can think of a vulnerability as an *implementation* failure, where the system does not do what it's expected to do. Fixing vulnerabilities in well-maintained software is generally *easy*, although engineering organizations that defer maintenance until vulnerabilities force their hand often blame the vulnerability for the cost of managing deferred maintenance and library dependencies.

A *misconfiguration* is just a user-selectable vulnerability (or sometimes, architectural defect): there is a known safe way to operate, and a system (often by default) is configured to *not* operate that way. Widespread misconfigurations often arise when the "safe" way to operate isn't user-friendly in some way; consider the difficulty of implementing HTTPS before Let's Encrypt made it cheap and easy to do for everyone.

Architectural Defect

An *architectural defect*, on the other hand, is a *design* failure: a system that is designed to something *incorrectly*. Most protocol flaws fall into this category, and many internet core services (DNS, SMTP, HTTP) have been riddled with architectural defects (if we can generously call those *architected*). Architectural defects are often quite challenging to fix, especially if they affect disparate systems or customer integrations. Consider when a vendor has to change its front-end API, how many different organizations will have to make changes before the vendor can consider the issue remediated.

Process Flaws

Sometimes the hazard exists in our processes. Process flaws generally fall into two types: processes that are directly unsafe, and processes that increase unsafety by, paradoxically, failing at creating perceived safety. There's a spectrum between the two; processes can

simultaneously create unsafety, while creating the appearance of greater safety!

Directly unsafe processes can range from processes that are ill-thought (deploying straight to production) to *error traps*: conditions that make it difficult to work safely (consider under-resourced teams). Many unsafe processes rely on humans to provide safety by *noticing* that something is going wrong; as the workload scales, humans can't provide that safety check at the same scale: noticing problems in a task you do once a month may be easier than noticing in a task you do dozens of times a day; because you can spend ten extra minutes thinking about safety once a month, but you can't afford that ten extra minutes dozens of times a day.

Processes can also be unsafe through a lack of process control. While there might be a preferred, safe version of a process that is safe, numerous *process variants* arise which have significantly different steps, which can create unsafe outcomes. Consider an employee termination process which *notifies* numerous system owners to remove permissions, but does not verify the removal of permissions – a variant where some system owners don't act on those notifications is an unsafe process.

Processes can also rely on untrustworthy inputs to humans: consider how many business processes are started with "a person receives an email instructing them to …." In the absence of authenticated email, and without a verification step as a control, any process like this is inherently unsafe.

Processes that create unsafety through perceived safety emerge through the concept of risk homeostasis (also known as The Peltzman Effect): humans have a set-point of risk that they'll tolerate, which is entirely based on their *perception* of risk. When they add (perceived) safety into a system, they'll become tolerant of more risk. This is why people drive *faster* as we add in mandatory safety features to cars. Sometimes, however, we add steps to processes that don't add in safety improvements … but the process owner *thinks* the process is now safer, and so might approve other changes to the process that decrease safety.

Environmental Hazards

Sometimes, the environment adds some hazards into our systems. Cosmic rays, for example (no joke!) can cause the bits to flip in TCP/IP packets. *Most* of the time, the error detection in TCP will identify a bitflip. If the *right* number of bits flip, the checksum will validate the now-corrupted packet – creating *data tampering* without a human actor.

As entertaining as cosmic rays are (and yes, I've dealt with incidents triggered by cosmic rays), more mundane environmental hazards can impact your risk. A fire in a sensitive facility might expose your data to firefighters. Anything that creates a long-term power outage is going to impact availability.

Human Error

Human error is a symptom of a system in need of redesign[1]. In more words: human error is not a hazard; a system that allows an unintentional error to create harm is the hazard. Often this is an architectural defect in the form of a lack of input validation.

Misuse / Unintended Use

Often, system designers make assumptions about the ways that a system will be used; and these assumptions drive safety design decisions. When the system gets repurposed and used in a different use case, those design decisions might look like architectural defects. This can range from using a log transport service designed for *eventual consistency* to transport *real-time* alerts at high volume, all the way to adding a web server to an embedded medical device. The safety assumptions made by the original system designer may not hold in the new environment.

[1] I've included Human Error here as a hazard simply so that if you looked for it, you'd see this note. This statement is attributed to Prof. Nancy Leveson.

Risky Actors (a.k.a. Adversaries)

It's hard to think about risk without considering an *adversary*: the entity that sets in motion the sequence of events that results in an unacceptable outcome. The challenge with language around *adversaries* is that it often limits us to only think about groups that willfully mean harm to our organization.

Threat Actors

The most commonly acknowledged adversaries are the vaguely-named *threat actors*: outsiders who are generally planning to do harm to your organization. This can include nation-states, profit-motivated entities, hacktivists, and various other adversaries.

Auditors & Regulators

For most organizations, their most likely source of harm comes not from their threat actors, but from auditors and regulators (before you argue this point, consider what percentage of your security spend is driven by *compliance*, not *risk analysis*). Failing an audit can be more disruptive than a major breach. Having a regulatory body decide to prosecute your organization and executives post-breach is now a realistic concern.

Law Enforcement / Lawful Intercept

If you have data about end-users and their activities, there's a very good chance that at some point, you'll be approached by a nation-state actor directly, asking you to directly extract information from your systems to give to them. Different countries have different beliefs about end-user privacy, and you need to consider all of the locales that you operate in, as well as all the ones you have end-users in, when planning your security strategy. Law enforcement often has special capabilities most adversaries don't, like *detain all of your in-country staff until you comply with their request*.

Chaos Monkey

Random issues can often act as the adversary, especially in cases of data leakage. When a security perimeter system is overloaded, it might fail open. What if the overload is triggered not by an adversary, but by some random confluence of events. A routine outage of one system might cascade through multiple hazards to produce an unacceptable loss. While incidents triggered this way are often considered *safety* incidents instead of *security* incidents, the scenarios share enough hazards to warrant considering the random confluence events in our risk management plans.

Insiders

Malicious insiders, while a rare occurrence for most enterprises, are absolutely essential when considering risk – not merely because the humans we employ might act maliciously, but also because the insider, *or their computer*, could be compromised or induced to take an action. While this might make the insider more of a hazard than an adversary, the simplicity of considering their

possible adverse actions merits discussion. Insiders might also be *temporarily* malicious, as they are either disgruntled by their employer, or trying to defeat a system that gets in their way.

Risky Situations (a.k.a Scenarios)

Now that we've considered adversaries, hazards, losses, and incidents, we can discuss risk in language that resonates with many stakeholders: scenarios. A scenario is a story, a fairy tale if you will, of how an adversary might exploit some set of hazards to cause an unacceptable loss. A scenario is a *class* of possible bad situations; when one happens, it becomes an *incident*.

Scenarios are *often* useful in more senior conversations when they capture more bad situations. Sometimes, one situation is enough to trigger action to prevent it (consider one public-facing system with high-value sensitive data and a known vulnerability), but usually scenarios like that are easy to prioritize remediation actions.

Triggers (vs Root Causes)

An incident, as an instantiation of a scenario, often has a trigger, which is just the first hazard that was engaged by the adversary. Often, in incident post-mortems, there is a focus on identifying a single root cause, rather than identifying all of the hazards that might have come into play. It's worthwhile to focus on the *proximate trigger* for an incident as a substitute for root cause – not to satisfy the need to identify exactly one problem, but to redirect that need towards the hazards that might need remediation.

Attack Paths

An attack path, or kill chain, is a specific set of hazards that could be exploited by an adversary. While this is the simplest form of a scenario, many attack paths look sufficiently similar to group them into a single overarching scenario that comprises many attack paths.

Risk Mitigations (a.k.a. Controls)

No conversation about measuring and managing risk is complete without discussing ways to mitigate risks. Risk mitigations generally fall into two categories: eliminating potential unacceptable losses, or controlling hazards. When we're thinking about risk *measurement*, controls raise an interesting question: how do you measure the *value* of a control?

Eliminating Losses

One way to mitigate risks is to identify an unacceptable loss, and eliminate the underlying asset. If a data breach of user data is a concern, identifying ways to eliminate the data from your environment entirely is one way to reduce your overall risk. While eliminating assets is rarely going to happen, often there are important conversations to have around unnecessary potential losses.

Controlling Hazards

Most risk mitigation is oriented around implementing controls to mitigate hazards. Controls often begin with a *control objective* definition ("all user accounts must require MFA") which hopefully connects to a known hazard ("password-based authentication, susceptible to password theft"). An organization then implements a control to meet this objective, and will need to assess how much the control effectively mitigates the risk.

This can often be challenging, as controls might include new hazards (how do you reset an MFA token?) that aren't obvious on first inspection. Most compliance regimes attempt to provide comprehensive control objectives to target related hazards, but many tend to spend less energy assessing the process flaws that might impact security controls.

Whew

Okay, that was … a lot of risk taxonomy, hopefully defining things you already knew. Now, on to how to measure risk.

Measuring Risk

At a very high level, risk is a measurement of "expectation of a bad outcome." It's often calculated, in some fashion, as "probability of a bad outcome" multiplied by "the damage incurred in that bad outcome." Most risk measurements target specific scenarios, rather than environments as a whole. In an attempt to measure risk, one should consider the effects on your risk approach based on how scenarios aggregate, the time-cost of money, incident recurrence rates, and enterprise planning horizons.

You Can't Measure Risk

Editor's Second Bias: if it's not a regularly occurring scenario, you can't put an objective number on any of the things commonly called a *risk*. If you have a retail store, you *can* measure the damage from shoplifting. An insurance carrier can model a lot of scenarios across a vast population. The discipline of actuarial risk is *fairly* well understood, and, generally, the frequently occurring events of today will repeat at a comparable rate tomorrow, barring large environmental changes. You can think of actuarial risk as a form of *descriptive analytics*: you aren't actually predicting what will happen in the future, you're describing what has happened in the past (and expecting the future will just continue what has happened in the past).

If you're worrying about risk that has an actuarial component, congratulations! You have a *relatively* easy task, with a fast feedback loop: if you predict that tomorrow you're going to lose $40,000, then you can check tomorrow, and update your model as needed.

Most of the time when we talk about cybersecurity risk, though, there isn't a good actuarial model to fall back on, and we're stuck trying to talk about risk in ways that feel objective and scientific. We're now in the field of *predictive analytics*, where we base our expectations on assumptions that may or may not be entirely accurate. We have to lie a bit to ourselves about the fidelity of our models, and make up a few numbers here and there, to produce risk scores that we can use to prioritize action.

And most of the time, the specificity and accuracy of those numbers aren't worth the electrons used to display them in Powerpoint.

That's okay. It doesn't matter what *I* believe about the value of a risk measurement, nor does it matter what *you* believe. What matters is whether our communication about risk is effective at producing wiser choices about risk in our business. If your business wants risk communicated in dollars for hazards, then that's a model you should adopt. If they wanted it reported in units of bazelquux, then figure out how to do *that*. The goal of measuring risk isn't to have measurements or score. It's to prompt action to reduce the risk the business faces, while generating the least amount of business friction along the way.

Why do we want to measure risk?

There are two ways that risk measurements tend to get used. One, in a world of constrained resources, is *prioritization*: the allocation of predefined scarce resources to one project over another. The second is *investment*: the allocation of *new* resources to a project. Note that at the macro level, investment is just another form of prioritization, as those new resources are generally taken from somewhere else. Unless your CFO has a money tree growing in their backyard.

Prioritization

One common use of risk measurement is to enable an organization to choose between competing types of work. For cybersecurity work, we're generally discussing *risk mitigation* projects, and the size of the benefit – or the measured *risk reduced* – is a key driver of choosing the importance of the projects. In many organizations, the benefit is discussed and prioritized *separately* from an analysis of the costs, which may make for conversations that run in loops: a hazard may be prioritized to be mitigated, but there is no project cheap enough to make it worthwhile. If those conversations don't happen in the same room, then a lot of wall clock time will be wasted cycling between different phases of project definition.

Investment Justification

Risk measurement is also often used to justify incremental security expenses. If we deploy a security system that costs $50,000, it feels easier to justify if we can find that we've reduced at least $50,000 of risk. To do so, it means we would already need to measure security risk in dollars. But peer executives may be happy to talk in scenario terms ("I'd pay $50,000 to not have this thing happen") instead of a strict ROI approach.

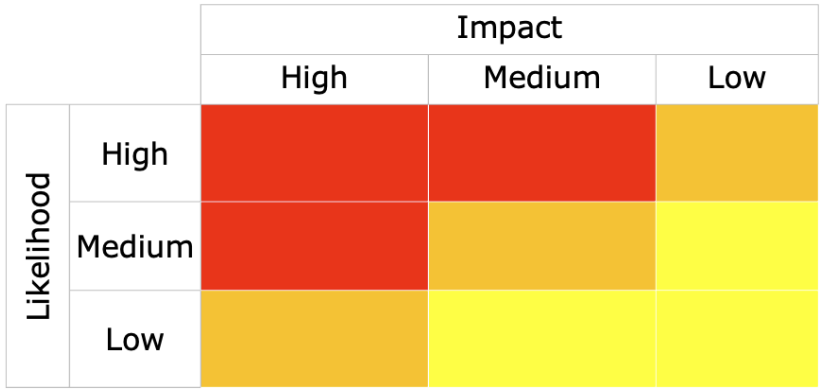
Risk Measurement as Nerd-sniping

Recognize that when an organization is already past their *mitigation appetite*–that is, there is very little chance that they'll spend incremental time or budget on a new activity–they may choose a non-confrontational approach. Rather than declining to work on a mitigation activity, they'll ask for more analysis on the hazard. A risk manager may end up spending more time doing analysis than it might take to mitigate the problem, but that cost has been transferred from an operational team to the risk management team.

Qualitative Approaches

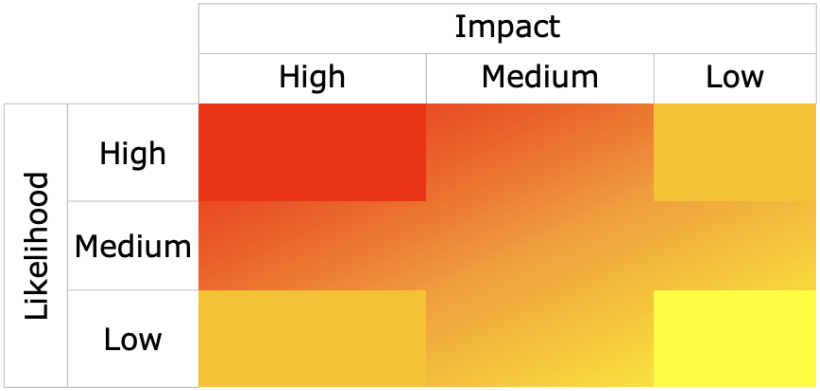
Nine-Box: Hazards

Most risk scoring systems start from the basic idea that you're trying to measure how much exposure you have: there is some harm that *could* happen, and there is a *likelihood* that it could happen. With those two dimensions, it's pretty logical to place them in a chart. On one axis, we have *likelihood*, and on the other we have *damage*. The Nine-Box is a classic example:



Note that we're using very coarse buckets for both axes: High, Medium, and Low. Nine-boxes resonate with people because those buckets are easy for them to personally assess. If you say the damage from a hangnail is High, they'll immediately question your ability to understand risk. The first problem of a nine-box shows up as soon as it gets used for prioritization: is a High/Medium risk more important to address than a Medium/High risk, or not? You can waste hours arguing about how to prioritize Low/High against High/Low, and what about Medium/Medium?

In practice, a nine-box is often really a five-box: any placement in the four corners we tend to believe, but any item with a Medium score on axis is confusing – is it really a Medium, or was it a borderline High that you don't feel like you can convince others of? Maybe you end up in long arguments with peers about risks that are right on the border – the damage isn't as "High" as the worst scenarios, but it's scarier than most of what's in Medium.



You may notice that the five-box here looks just like the box used to describe categories of risk in the introduction, and the insight here is profound: The real issue here is that we have very nebulous processes (if any) for handling hazards that don't fall into one of those four extreme corners.

25-box: Hazards

Instead, we can attempt to solve the derailing argument about Medium-to-High items by adding in new buckets for Medium-High and Medium-Low on both axes. It doesn't make the chart any easier to read, and now you can have scores like "Medium-High/Medium-Low", which just adds confusion. And having 25 total buckets means you have a greater *risk* of not scoring any of your issues into a given bucket, and that makes it hard to proceed – an argument executive might wonder if you've really done a complete risk assessment if the Medium/Medium-High bucket is currently empty.

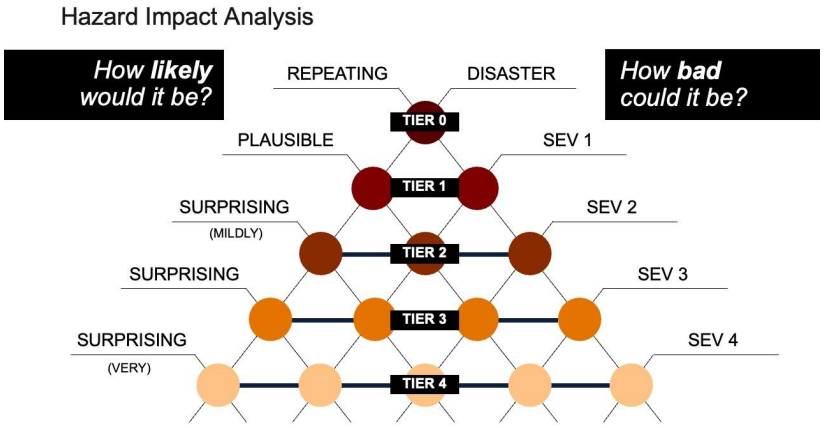
25-Box

Likelihood \ Damage	High	Medium-High	Medium	Medium-Low	Low
High					
Medium-High					
Medium					
Medium-Low					
Low					

Pyramid of Pain

Based on the 25-box qualitative approach, the Pyramid of Pain turns simple qualitative models on their head (quite literally). While it uses two axes that closely resemble likelihood and damage, it uses proxies for each of those, and then rotates the output to clearly put the highest likelihood/highest danger at the top.

The Pyramid of Pain gets a deeper treatment here because it isn't well-documented. It was developed at Akamai while I was the CSO there, after we struggled with most of the above risk scoring methods, and found we were spending more time arguing about priority than actually fixing things.



Most risk scenarios rely on the judgement of humans to decide what the likelihood of a given scenario is, and the Pyramid of Pain leans into this. The five different buckets for likelihood are based on *who would be surprised* to see that scenario unfold. At the top are *Repeating* events: a scenario that is happening on a regular basis. Below that are *Unsurprising* events: scenarios that we would consider to be *expected*, as few would be surprised by this happening, but for some reason it hasn't happened yet (the pyramid doesn't use the term *expected* because that word potentially has materiality issues for SEC disclosures). Below that are the three *surprising* categories (*mildly*

surprising, *surprising*, and *very surprising*), and we define them based on *who* in our organization would be surprised. Consider a Product Manager: people who are paid to be optimistic about outcomes, and so, as a profession, tend to blindly (or knowingly) accept more risk. Scenarios that would surprise them, but no one else, are considered *mildly surprising*. Scenarios that would surprise our standard business executives as well are considered *surprising*, and those that would startle even paranoid security professionals get scored as *very surprising*. A nice feature of this axis is that it is very difficult for one person to argue with a score: an engineering director can't assert that a scenario should be very surprising, because a security engineer saying that *they* wouldn't be overly surprised is sufficient to keep a scenario in the *surprising* bucket.

For damage, the Pyramid relies on companies already having a well-understood incident management program, with severities that are commonly accepted across a business. Most incident systems have four severities, and while naming conventions differ, the Pyramid uses severities one through four. A *Sev1* incident might be one which is majorly disruptive to a significant percentage of the company's customers, like an outage or data breach. *Sev2* incidents have a major impact on a small group of customers, or a moderate impact across all customers. *Sev3* incidents are those with some noticeable impact to customers, and *Sev4* are those with no meaningful impact to customers. By mapping cybersecurity scenarios onto an existing incident severity scoring system, security teams adopt the language of their stakeholders to talk about risk scenarios. The Pyramid adds one severity level at the top: *Disaster*. Disasters are Sev1 incidents that are *also* transformative: the business that comes out of the other side of the incident is a different company than the one that entered the incident. SolarWinds had a Disaster. Major incidents that cause companies to completely reprioritize security and resilience for an extended period afterwards are disasters.

Tier 0 risks are obvious: Repeating Disasters. You know them when they happen (by definition, nothing can be Tier 0 until a scenario unfolds), and you understand how transformative they are. Tier 0 risks don't require prioritization, because they're so obvious that they are dealt with by a full company effort.

Tier 1 risks encompass two categories: Repeating Sev1, and Unsurprising Disasters. Often, a given *hazard* will have scenarios in both locations: a hazard being repeatedly triggered to create Sev1 incidents at a notable cadence likely has a nearby scenario in which it can be triggered for far greater harm. This concept repeats across all Tiers, and leads to one of the strengths of the Pyramid of Pain: it doesn't attempt to prioritize within a Tier. All Tier 1 risks are problematic for a business, and should be dealt with urgently. If the scenarios for a hazard clearly fall into this tier, rarely is deeper conversation warranted about prioritization.

Tier 2 risks are where most CISOs operate: Repeating Sev2s, Unsurprising Sev1s, and Mildly Surprising Disasters. These risks feel dangerous to *most* of the business, but how urgently they need to be addressed is often up for debate. CISOs are often asked to prioritize within the class of Tier 2 risks, and this can be a time-consuming quest. The Pyramid of Pain explicitly *does not prioritize within tiers*, and this is most important with Tier 2.

Tier 3 risks are those that a CISO should be aware of, but you should have methods for getting these triaged and managed within your organization. CVSS/EPSS might be sufficient for software defects; architectural issues should be addressed with change requests, and interactions between security architects and engineers. Surprising Disaster scenarios will rarely be compelling to most executives, and a CISO will likely waste political capital engaging on them.

Tier 4 risks generally fall into two categories: the extremely low likelihood bad things ("Chicken Little risks"), and frequent but annoying problems. If the cost is low for annoying problems, get them dealt with using routine mechanisms, and focus on creating operational efficiencies to get more of them fixed at lower costs. For Chicken Little risks? Generally, leave them alone.

The Pyramid of Pain shares the same benefit of the nine-box and 25-box: being extremely fast to score a risk. Further, by using qualitative judgements based on the *rest* of the organization's definitions, it removes the weakness of relying solely on a security professional's judgement, and the arguments that necessitates. And it allows for quick triage into different risk tiers, because the way in which you'll address risk is often substantially different from tier to tier.

"There's plenty of good work available." Kathryn Kun, who led and built much of Akamai's Severe Vulnerability management program (which addressed Tier 2 risks) was fond of this saying. Most companies – even most teams within a company – have more Tier 2 risks near them than they can afford to work on at once. Having one team simultaneously address multiple issues that don't share mitigation strategies is a recipe for wasting time in context switching; none of the mitigation projects will make significant progress, and the longer a team works on a project without success, the less important the work feels. Offer a team the opportunity to pick from the good work that is necessary. Four hazards that they need to address allows them to pick the one that either worries them the most or that they can solve the most easily, allowing them to focus their mitigation appetite on the activities that have the highest return.

Actuarial Approaches

Recurring Loss

If you have an event happening at a regular frequency, and you can tightly measure the costs from those events, you can use a simple calculation: your risk is simply the cost per event times the rate of events. If shoplifters steal, on average, $100 worth of merchandise, and you have 10 shoplifters a day, then your recurring loss is $1,000/day, or $365,000 per year. You can continuously measure this loss, and, when it comes to testing out mitigations, you can usually quickly determine if the mitigation is worthwhile. If you hire security guards, and you're paying two guards a day for eight-hour shifts at $30/hour, with 22% overhead (payroll taxes, benefits), their presence needs to reduce that $1,000/day to $414.40/day *just to break even,* before you even start to count the cost of managing the guards.

Annualized Loss Expectancy (ALE)

ALE is a formula that expands from measuring recurring loss to calculating how much loss you expect to occur from some low-probability risk scenario over time. In its simplest form, we're measuring the probability of an event against the damage from the event. Consider having a house in a floodplain. If you're in the hundred-year flood zone, you have a 1% probability of a flood hitting. If your house is worth $500,000, and would be a total loss in a flood, your annualized loss expectancy is $5,000/year.

The problem with ALE in cybersecurity is that the result is highly sensitive to its inputs. Often, both of those numbers are estimates or guesses. Take the hundred-year flood plain. Historical data might predict a flood every hundred years, but recently those flood plains are now looking like forty-year plains. Do you assume that will continue, or use the hundred-year number? Is being flooded out of your house only going to cost you $500,000, or do you add in estimates of moving costs? And what about the loss of priceless items?

For many cybersecurity risks, there are no hard numbers to fall on. What's the true cost of a breach? Depending on which analyst or vendor you ask, they'll have different answers. And do the scenarios they drew from for their aggregate estimates match the scenario you're considering? As for probability, cybersecurity scenarios are hard to predict. The odds of ransomware in 2018 seemed really low, but in 2021, they spiked, as adversaries better operationalized their attacks and companies were more willing to disclose an incident and make claims. How your existing controls mitigate the probability of an event, or the harm from the event, may vary over time.

Claiming that a hazard is a $44 million dollar risk invites argument. Most businesses are going to take that number *very* seriously, and the methodology that calculates it needs to stand up to scrutiny. If your company is worth $4.4 billion dollars, *every* hundred-year company-ending hazard is a $44 million dollar risk. Does your company actually care about hundred-year hazards, though? Most

don't, and likely *shouldn't*. There are more than a hundred scenarios that you could plausible argue will end a company's existence in the next hundred years, which means that every year your cumulative ALE is *larger* than the total value of the company. No single scenario would be worth addressing in that circumstance.

Fermi Problem Approaches

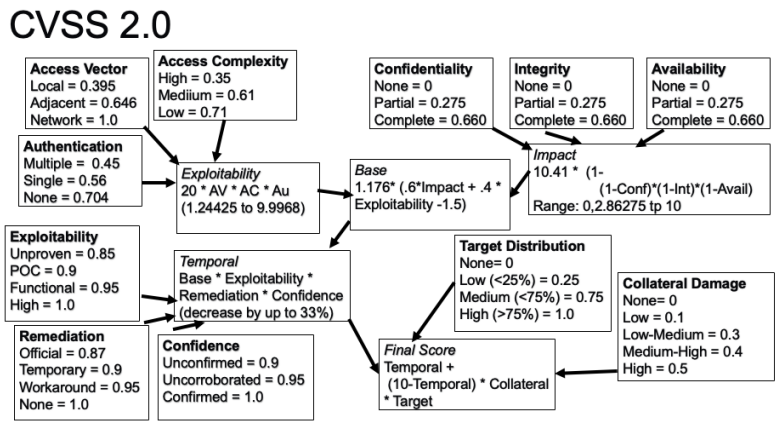
Fermi approaches to solving numeric problems are a well-understood model. If you were asked how many piano tuners were in New York City, you might make a few assumptions to begin: how many pianos can a tuner tune in a year? How often does a piano need to be tuned? What percentage of residents of New York have a piano? How many people live in New York City? You could then multiple these together, coming up with an estimate of how many piano tuners are in New York City. You ignore some inconvenient details to make this work – it's obvious that a piano on display in a penthouse isn't getting tuned at the same frequency as one in a pianist's studio.

What makes a Fermi approach work is that, if you string together enough guesses in your formula, you know that some of your guesses will be wrong, but you hope the wrongness will balance out – you'll be high on a few and low on a few, and those will hopefully cancel out to give you a good guess.

That same approach shows up in a number of risk scoring models. In these, you're invited to use a formula that looks like a much more complex version of ALE, in which we insert guesses. In some of the formulas, the guesses are tightly controlled, and you might select from "Low / Medium / High", and each of those choices inserts a different number into the equation. In other formulas, you're invited to use your own estimates for those numbers instead.

Common Vulnerability Scoring System (CVSS)

CVSS was designed to score a very specific type of hazard: vulnerabilities in third-party software. CVSS originated in 2004 by NIST and FIRST, and has had four major versions. CVSS scores a vulnerability with three sub-formulas: the base score describing intrinsic risks to systems that have the vulnerability (How hard is this to exploit? What level of system compromise does this result in?), a temporal score evaluating factors that change globally over time (Is there an exploit in the wild? Is there a patch?), and an environmental score mapping the vulnerability into a specific environment (how valuable is the system in your environment? How many systems do you have that are vulnerable?). Each of these scores is calculated by taking some qualitative inputs, giving those numeric values, and calculating them in a complex formula that outputs a score from 0.00 to 10.00. Below you can see how this was calculated for CVSS 2.0. CVSS 4.0 "simplified" this by having the initial qualitative scores combined through a set of logic gates, and then having a lookup table based on those results.



CVSS has all the hallmarks of a risk score: elements to infer probability, and elements to infer damage. CVSS's greatest benefits in talking about risk are two-fold: by not outputting a *monetary* number, it removes a point of argument around the reality of that number; and by hiding the details of the calculation in a complex formula that outputs a precise number, it removes the opportunity to have an argument about something like Medium versus High: someone outside the CVSS scoring system can't easily tell the difference between a 6.83 and a 7.15, other than "our auditors measure that we patch vulnerabilities within 30 days if the CVSS score is above 7.00."

CVSS does *not* exist to prioritize all the vulnerability remediation work that needs to be done. Instead, it exists because *most* vulnerabilities will not be remediated, and instead allows organizations to focus limited remediation energy on fixing the perceived *serious* dangers, and to identify the *critical* dangers that need rapid remediation.

CVSS has a subtle hazard: its reliance on a person scoring a vulnerability to understand all of the ways that vulnerability might be part of complex attack paths with dangerous but non-obvious scenarios. A remote code execution on a marketing web server might just be an embarrassment for a company … but if that server happens to have credentials that allow lateral movement into a CRM, that's a much more dangerous risk. Misscoring the environmental inputs may be a process flaw that results in more hazards than expected.

Exploitation Prediction Scoring System (EPSS)

If CVSS exists to help organizations prioritize the use of limited resources, EPSS takes that one step further. Correlating CVSS scores for vulnerabilities with breach data from publicly disclosed breaches, it's easy to see that CVSS outputs high scores for vulnerabilities that ultimately are not widely exploited, and sometimes provides low scores for vulnerabilities that are exploited. EPSS can be seen as an adjacent model to CVSS, which heavily weights one input: the exploitability of a given vulnerability. EPSS uses a model to predict if a vulnerability will appear on the Known Exploited Vulnerabilities (KEV) list. If so, the vulnerability is seen as a great candidate to remediate quickly, even if you don't know exactly how a system compromise will turn into a major breach for you.

EPSS addresses the flaw in CVSS requiring a deep understanding of an environment by instead so heavily weighting the probability of anything bad happening that the damage is of much lower weighting. Relying on the

accuracy of the KEV list, proposing we dedicate vulnerability remediation resources almost exclusively on solving for the possibility of any compromise at all.

Both CVSS and EPSS are rooted in the assumption that the majority of vulnerabilities cannot be remediated. Twenty years ago that might have been a valid assumption. Companies had little automation in their software management, development, and release pipelines. Software dependency tracking was rarely done at scale, and chasing down vulnerabilities and the teams that owned them was a fool's task. Is that really still the case? With modern CI/CD pipelines, automated QA, release orchestration, cloud workloads, and agile processes, we should challenge the assumption that enterprises can't, by default, maintain their software to be relatively up-to-date. CVSS and EPSS should be used to prioritize the remediation of vulnerabilities that present real-time risk that needs to be addressed in days, rather than routinely in weeks or months.

Factor Analysis of Information Risk (FAIR)

FAIR (Factor Analysis of Information Risk) is intended for the analysis, understanding, and quantification of information risk in monetary terms. It distinguishes itself from other Fermi-style risk assessment methodologies by calculating a monetary value. Since FAIR modeling produces a dollar value, its accuracy is a double-edged sword: a powerful tool to prioritize, but inaccuracies can inflict damaging wounds on the reputation of the risk manager.

FAIR relies on accurate data. Its effectiveness is *heavily* dependent on the availability and reliability of data concerning the frequency of scenarios, the costs of their impacts, and the effectiveness of security controls. In many instances, particularly for novel or emerging threats, such data is scarce, speculative, or highly uncertain. With a rapidly evolving threat landscape, historical data may not even be a reliable predictor of future risks.

FAIR is to ALE as CVSS is to the original Nine-Box: a more complex way to achieve a similar result. FAIR, however, loses some of the benefits CVSS has in producing an abstract score; as a monetary score will always be subject to deeper inspection. However, for organizations that seek a monetary score, FAIR can make for valuable conversations, especially if built atop a coherent model of your environment.

Aggregate Risk Scoring

The scoring models above are all focused on scoring individual things: either specific hazards (CVSS) or scenarios (FAIR). An important challenge for most CISOs is to look at how to measure aggregate risk. If you have an environment with 20 scored risks, an important concept becomes the risk score *for that environment*. It is unlikely to be as simple as just adding together the risk scores in that environment

Attack Paths

Attack paths, as a collection of hazards with access to trigger an unacceptable loss, represent one way to look at aggregating risks at a lower level. You can consider an attack path as a collection of hazards which can lead to a loss, scoring an attack path then becomes assessing the likelihood of the attack path being exploited (probability of each hazard in the path multiplied together) against the harm caused in that scenario. You can look at critical hazards that appear in multiple attack paths to identify hazards that might be significantly more relevant than a single attack path would highlight.

Compliance Regimes

Various compliance regimes generally provide a completeness score against some set of security control objectives, counting how many of a specific set of controls your environment has implemented to meet those objectives. Rather than attempting to evaluate specific risks and aggregate them, these regimes assert a standard set of controls, and measure compliance. These can range from full organizational compliance regimes (SOC2, FedRAMP) to scoped regimes (PCI-DSS) all the way to very specific environments (SSLLabs for webservers, Hardenize for Internet-facing domains). The power in a completeness score depends on an agreement in the value of *all* of the controls in scope for the compliance regime; one which has a significant set of control objectives perceived to be useless will be hard to use to effect change: 80% compliance against a regime perceived as only 70% relevant may seem like an overinvestment.

The real bad outcome from failing to meet a compliance regime may not be the unacceptable loss of certain scenarios: rather, it's the direct loss of business from not meeting the compliance regime requirements. Compliance to a standard can often be viewed as a *product* feature, rather than as risk mitigation activities.

On to Mitigation!

Go have great risk conversations. Don't spend more time on risk measurement or prioritization than you do on actually mitigating risk and enabling the business to move faster. If you've just started a new role, you may want to take a look at [The First 91 Day Guide] to being a CISO.