SURFACE • SHIFT • SECURE

Before the Room

A field guide to leading outcomes you have no authority to command

MIKE SALARI

Staff mobile engineer · 500+ technical interviews from the hiring side

Contents

Preface: the view from the other chair IN THIS SAMPLE

A note on method and evidence IN THIS SAMPLE

How to use this book

PART ONE. THE GAP

1. The engineer who was right and lost
2. Responsibility without control
3. The decision is made before the meeting IN THIS SAMPLE

PART TWO. SURFACE

4. Find the hidden decision IN THIS SAMPLE
5. Map power before you argue
6. Read the incentive conflict
7. Credibility and decision rights

PART THREE. SHIFT

8. Pre-alignment is not politics IN THIS SAMPLE
9. Strategy that can be adopted
10. Architecture alignment and decision systems
11. Turn ambiguity into executable work
12. Scope, risk and deadline negotiation
13. Lead migrations when the cost is local
14. When green status hides red reality

PART FOUR. SECURE

15. Escalation without status warfare IN THIS SAMPLE
16. Incident leadership when the system failed
17. Conflict with high performers, product and executive pressure

PART FIVE. MAKE IT A SYSTEM

18. The weekly operating system

19. The thirty-day repair

20. The evidence built before the review

REFERENCE

Appendix: the artifact system

Notes

Further reading

Resources

Operational index

Afterword: Where to go from here

Chapter and figure numbers are those of the full book.

BEFORE THE ROOM

Preface: the view from the other chair

I work as a staff mobile engineer, and in about fifteen years of building mobile software I have sat on the interviewer's side of more than five hundred interviews. Debriefs, hiring committees, promotion calibrations, the quiet conversation after the candidate leaves the room. That seat shows you something the candidate almost never gets to see: the moment a decision about a person is made, and how often the reasons that settle it are not the ones anyone says out loud.

I have watched the same thing happen to technical decisions. The architecture call, the migration, the deadline, the launch that everyone agreed was green. For consequential decisions, the visible meeting is often not where the decision happens. It was shaped earlier, in a hallway, in a review comment, in an incentive nobody named, and the meeting largely ratifies it. I have seen engineers lose outcomes they should have won because they prepared to win the meeting, bringing a better diagram to a room that had already made up its mind.

This book is about the gap that opens when you become the person a result depends on before anyone gives you the authority to command it. Senior engineer, tech lead, staff engineer: the title changes, the gap does not. You are accountable for something you cannot order into existence. You can be completely right and still watch the outcome drift, and if you have felt that, this book is written for you.

I am not going to hand you leadership principles. There are good books full of those, and I will point you to some. What I could not find was the field manual for the specific moments where technical risk, incentives, power and decision rights collide, and where being right is the entry fee rather than the win. So I wrote it, from the vantage point I know best: the debrief room.

The scenes in this book are composites. The note on method and evidence that follows says exactly what in them is invented and what is not.

The book has one engine, three moves, and you will meet it in chapter three. The chapters before it exist to convince you that your problem is not a knowledge problem; everything after it applies the moves to the situations where technical leaders most often get stuck.

Read it with a current problem in mind. The book is built to be used on Monday, not admired on a shelf.

BEFORE THE ROOM

A note on method and evidence

This book is written from practice, not from a study. The vantage described in the preface is its advantage and also its limit, and it is worth being clear about both.

The scenes are composites. Each one is assembled from patterns I have seen many times rather than from any single team, person or company. Inside a scene, the names, the dialogue, the dates, the durations and the numbers are illustrative details, composed so the pattern can be followed; read them as a worked example, not as case-study data. What is not invented is the pattern underneath. The rule I have kept is about evidence: the book presents no fabricated statistics, studies, research findings or named real-world cases as support for its claims. When a scene shows a hiring manager's sentence or a debrief's verdict, it is a pattern I have heard in many forms, written as one line, not a transcript.

I have tried to keep the language of certainty honest. When I write that I have repeatedly seen something, that is a practitioner observation. When I write that this suggests a rule, that is my inference from those observations, and you may reasonably draw a different one. Where a claim rests on published research or on platform documentation, a numbered note names the source, and the notes say what the source does and does not support. Where a claim rests on what I have seen, the sentence says so, and no note pretends otherwise. The research is support and context; it is not where the ideas came from.

The individual tools are not mine. Decision records, stakeholder maps, escalation briefs, postmortems, explicit decision rights and pre-alignment are established practice, and many teams already use them well. What this book adds is the synthesis: naming the authority gap, running Surface, Shift and Secure in that order, carrying each decision in a durable artifact, and turning the whole thing into a weekly operating system.

The book also has boundaries, and it names them where they matter. Not every organizational failure can be solved by a sufficiently skilled engineer. Some technical objections are simply correct. Some costs cannot be made fair, and some decisions belong above you, including risks that no one at your level can rightly accept. Where a technique could be used to silence dissent, shift risk onto someone without the authority to hold it, or paper over structural harm, I have tried to say so directly. Three sections show the method failing in full: an objection misread as an incentive in chapter six, a decision nobody can legitimately make in chapter seven, and an organization that punishes the person who raises a risk in

chapter fifteen. Surface, Shift and Secure are tools to test, not laws proven in a lab: treat them as hypotheses to run on your own situation. The evidence that should convince you is the outcome you manage to move.

PART ONE

The gap

why being right stopped working

CHAPTER 3

The decision is made before the meeting

The architecture review is scheduled for Thursday afternoon. By Wednesday night, the decision is largely made. Nobody has said so, and half the people walking into that room believe they are walking into an open question. They are walking into a ratification.

It happened in a pattern I have watched play out on many teams. A senior engineer, call her the proposer, spent three weeks on a proposal to consolidate two divergent services into one platform. It is a good proposal. It removes a class of recurring bugs, clarifies ownership, and cuts duplicated on-call load. She books the review, invites the right people, and prepares to make her case.

What she does not notice is that on Monday, in a hallway, the director who owns one of the two services told a staff engineer he was "not thrilled about losing control of the data model." The staff engineer, who will be in the room Thursday, took that as his cue and decided not to spend his credibility backing the consolidation. The product manager, asked in passing whether her team could absorb a migration this quarter, had already said no to her own manager. By Wednesday night, three of the five people who matter have privately settled their positions, none of them in a meeting, none of them on the merits, all of them before the proposer has said a word.

On Thursday she gives the best version of her case to a room that has already decided. She cannot see it, because she prepared to win the meeting.

The insight the whole book runs on

Many consequential decisions are substantially shaped before the visible event; the meeting is often just where that shape first shows. The book's shorthand: the decision is made before the visible event. (figure 1)

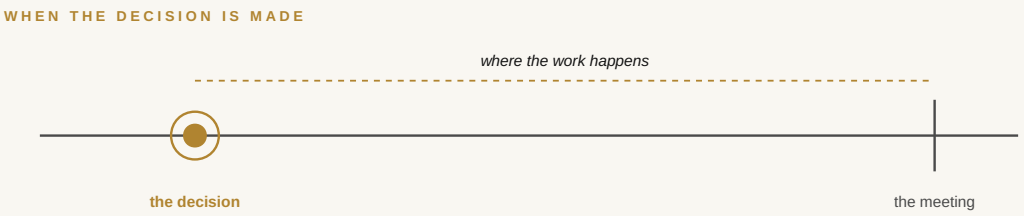

Figure 1. The decision is often shaped before the meeting. The meeting frequently ratifies it. The work happens in the span between.

I saw it first in hiring debriefs; chapter twenty turns it on your own review. Technical decisions show the same pattern. The architecture call, the migration go-ahead, the deadline, the launch approval: the meeting where they appear to be decided often sits on top of positions formed earlier, in hallways and reviews and one-on-ones and, above all, in incentives. Among the engineers I have watched on these outcomes, the ones who lost had usually poured their preparation into the meeting, and the ones who won had usually been working on the decision days earlier, while it was still forming. That is what I saw, and it is no law; early work loses too, and some meetings are decided in the meeting.

The claim is about timing and location, not politics: to move an outcome, you intervene where and when the decision is being shaped.

The engine, in three moves

Three moves, always in this order. They are the spine of every chapter that follows, and you should be able to say them out loud. (figure 2)

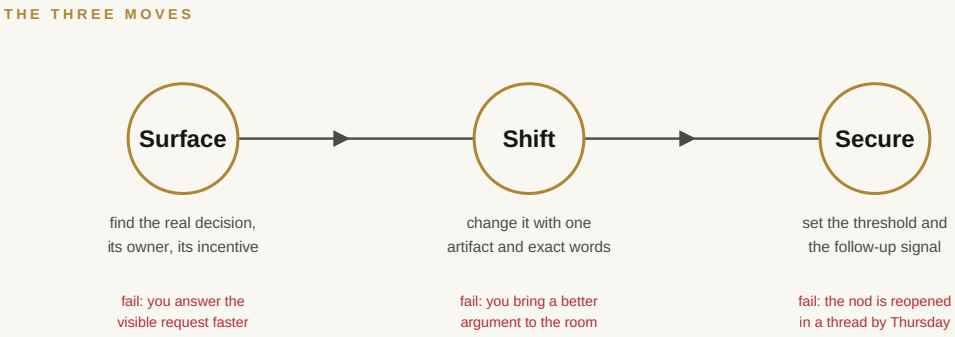

Figure 2. Three moves, in order. Surface the decision, shift it with an artifact, secure it with a threshold. Under each is the failure that skips it.

Surface. Find the decision that is being made, who owns it, and the incentive that will decide it.

Shift. Change that decision with one artifact and the exact words to say, placed in front of the right owner, before the room.

Secure. Set the threshold that triggers escalation and the follow-up signal that proves the change held.

Before the room does not mean behind people's backs. It means surfacing constraints early enough that the room can make an honest decision, and chapter eight draws the line between that and rigging the room.

In my experience a stuck outcome usually traces back to a skipped move. Surface gets skipped because the visible problem looks like the real one. Shift gets attempted with a better argument, aimed at the meeting instead of the owner. Secure gets skipped when the meeting nods, and the decision is reversed a week later.

Move one: Surface

Most requests arrive as the top of a stack. "Can you confirm we are on track" sits on a decision about what evidence counts as on track, and who defines it. "Review this design" sits on a decision about who owns the data model afterward. The request is the tip; the decision is the mass underneath, usually with an owner who has not said so. Answering the tip faster does not move the mass.

So before answering, ask three questions. What decision is being made here? Behind "we need to pick a queue technology" might sit "does the payments team keep owning throughput or hand it to platform." Who owns that decision, which is frequently neither the person asking nor you? And what incentive will settle it: the roadmap someone is protecting, the on-call load someone is avoiding, the credit someone is guarding? Chapter four turns these into a working method.

Here is Surface as a weak sentence and a strong one. The weak version, said in the meeting: "I think we should consolidate these services." The strong version, said to yourself first and then to the right person: "There are two decisions here. Whether his team gives up ownership of the data model is the director's call. Whether the services consolidate is the portfolio sponsor's, and the director cannot approve it, but his objection carries weight there. He will decide the first, and argue the second, based on whether consolidation makes his team look essential or redundant next quarter." You cannot move a decision you have not located.

Move two: Shift

Once you have the decision, the owner and the incentive, you have to change the decision, and the common mistake is to attempt it with a better argument.

Conversation decays. It cannot reach the people who were not there, by the following week it exists in one version per listener, and a decision held only in memory tends to belong to whoever remembers it most forcefully. The proposer had a wonderful argument on Thursday, and nobody outside that room ever heard it.

Influence becomes durable when it leaves the conversation and enters an artifact.

An artifact is any object that carries the decision: a one-page memo, a decision log entry, an evidence table, a scope memo, an architecture decision record. None of these is new. A good one names the decision and its owner, turns the trade-off into a choice between named costs, reaches people who were never in the meeting, and states what would reopen it.

Watch it on a scope fight. The weak move, said to a product manager under deadline pressure: "There is no way we can hit that date." That invites "well, make it work." The strong move is a one-page memo with three options, handed over with one sentence: "Option A ships on the date and defers the fraud checks to a fast follow, option B ships the full scope two weeks later, option C cuts the international flows. I recommend B. Which do you want?" The argument about whether the date is possible has become a choice, sitting with the person who owns the priority. Chapter twelve builds that memo in full.

Move three: Secure

A decision that is not secured often does not stay decided. It gets reopened in a chat thread, undermined by someone who nodded but never agreed, or eroded because nothing changed in how the work is tracked. Secure has two parts, and you set both when the decision is made.

The first is the escalation threshold: the condition under which you will raise the decision to someone with more authority, stated in advance so it is neither a surprise nor a threat when it fires. "I will escalate if this goes badly" is a vague warning. "If the integration test is not passing by Friday, I will bring the launch decision to the sponsor and ask them to choose between slipping the date and cutting the international scope" is a threshold. Set in advance, it reads as a professional instrument; improvised under pressure, the same escalation tends to read as going over someone's head.

The second is the follow-up signal: the observable evidence you will check, and when. After a week, did the owning team change what it tracks? After a month, did the new behavior become the default? Without a signal you are trusting that a nod became a change.

The weak version of Secure is to leave the meeting relieved. The strong version, said as it closes: "Good, we are going with option B, full scope two weeks later. I will update the decision log today with the trade-off we accepted. If the integration test is not passing by

Friday, I will bring it back to you before the new date can slip. And I will check next Tuesday that the tracking reflects the new scope." Thirty seconds, and a fragile agreement has a tripwire and a checkpoint.

The three moves as one motion

In a real week the three moves are often one conversation and two artifacts.

Run the proposer's week again. On Monday she names the real decision: the director's ownership of the data model, driven by his need to keep his team essential. On Tuesday she brings him a one-page memo showing how consolidation could make his team the owner of the new platform, framed as a choice about how the ownership transfer is structured. The staff engineer gets the same memo early enough that his concerns can change the proposal instead of ambushing it. By Wednesday the director agrees to hand over the data model and to support consolidation, but not this quarter: his team has a launch to land first.

On Thursday she gives a much shorter case to people who have already heard every cost, and the portfolio sponsor decides in the open: consolidation approved, start date pushed a quarter. She writes the delay into the decision log as the price of the yes, with a threshold (if the director's launch slips, the start date goes back to the sponsor instead of drifting with it) and a check the week the quarter opens. Meanwhile the duplicated on-call load she wanted gone stays.

Same engineer, same proposal. What changed between the two Thursdays was where and when she intervened and what she carried in with her, and even then she won the choice and lost the date.

IN THE FIELD

The move. All three, introduced together: Surface the decision, Shift it with an artifact, Secure it with a threshold and a signal.

The artifact. No template here; the artifact is the one-page memo or decision log the moves produce, whenever a decision needs to travel to people who were not in the meeting.

The signal. A week later the decision is still closed: the log reflects it, including any cost accepted to get it, and nobody has reopened it in a thread.

The level. A senior engineer runs the three moves on a decision their own team owns; a staff engineer runs them on a decision owned elsewhere, reaching the director and the skeptical staff engineer before Thursday. A principal makes the ownership question a standing part of how consequential reviews are prepared.

The boundary. This does not apply when there is no open decision to move, only a settled directive from someone with the authority to give it.

The line. "Which decision are we making here, who owns it, and what will keep it closed a week from now?"

PART TWO

Surface

see what is being decided

CHAPTER 4

Find the hidden decision

The message lands at 4:12 on a Tuesday, from the director sponsoring the release. "Hey, can you confirm we are still on track for the fifteenth?"

The engineer who receives it does what a conscientious engineer does. He opens the board, walks the open tickets, checks the two dependencies that worried him last week, pings the one teammate whose task looked soft, and by six he has assembled a careful answer. Yellow, trending green. Two risks, both with mitigations. He writes it up cleanly and sends it back, a little proud of how fast he turned it around.

Nothing happens. The date does not get easier. Ten days later the same director announces in a planning channel that the fifteenth is now a hard commitment to a partner, and the engineer feels the floor tilt, because he never agreed to a hard commitment. He answered a status question. The status question was scaffolding around a decision the director was making, whether to promise this date to someone outside the team, and a careful yellow-trending-green from a respected engineer gave him the cover to say yes.

The trap is competence pointed at the wrong object

In my experience this is one of the most common ways strong engineers lose outcomes, and it is hard to see from the inside, because it looks like being responsive. The faster and cleaner you handle the request, the more it feels like leadership.

This chapter is about getting from the request to the decision underneath it at 4:12 on a Tuesday, when answering feels so natural. (figure 3)

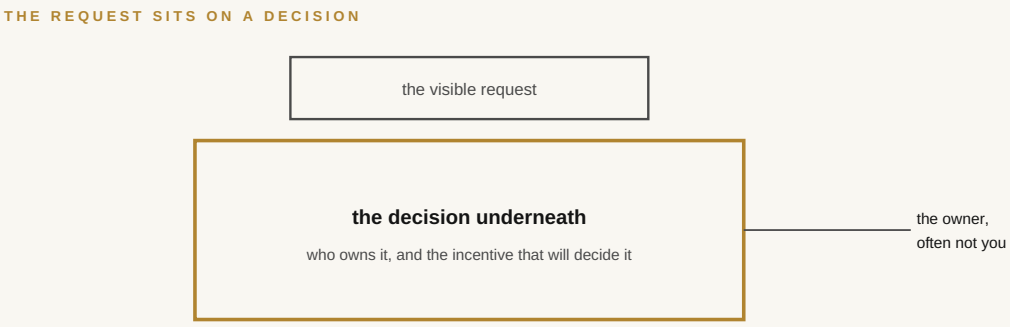

Figure 3. The request is the tip. The decision underneath is the mass, and it usually has an owner who has not said so.

Two sentences, same engineer. The weak one, typed straight back into the thread: "Yes, we are on track, two risks but both have mitigations, should be fine for the fifteenth." The strong one: "Before I confirm, are you deciding whether to commit this date to someone outside the team? My confidence for an internal target and my confidence for a hard external promise are different numbers, and I want to give you the right one." Once the director has to answer that, a status update can no longer be spent as a commitment the engineer never made.

The opposite reflex, deciding privately that the date is shaky and resolving to make it work yourself, is chapter one's working harder, and it hides the decision from the only person who can make it.

Three cuts down to the decision

When a request lands and you feel the pull to answer, run three cuts first. Each takes a sentence.

The first cut asks what decision this request is scaffolding. Behind "can you review this design" often sits "who gets paged at three in the morning when this fails, and does that team know yet." Ask what someone would have to decide for this request to make sense. The answer is usually written down nowhere, which is often why it stays stuck.

The second cut asks who owns that decision. Frequently it is neither the person who brought you the request nor you. The engineer with the launch date did not own the decision to make an external promise; the director did, and naming him tells you where to aim. Assuming you own a decision because you care most about it is the trap chapter seven takes apart.

The third cut asks what incentive will settle it. The director's decision was never going to turn on a risk analysis. It would turn on how much the partner relationship mattered to him this quarter and what slipping would cost him. Ask the question and sit with the answer even when it is uncomfortable, because your eventual move has to speak to it.

Run the three cuts and you have the decision, its owner and its driver, before you have argued, proposed or built anything.

When the request is honest and the decision still hides

Sometimes the person asking is hiding nothing. The decision is hidden from them too, which changes how you surface without changing whether you should.

A product manager asks, "Can we get the search feature out by end of month?" Underneath is a decision nobody has made: what gets cut from the roadmap to fund a month of search, and who tells the stakeholders waiting on it. Answer the scheduling question, "sure, if we drop the accessibility pass and the two bug fixes, probably," and you have told the truth and still lost, because the resourcing decision stays invisible and you will be blamed when the dropped work surfaces as a fire in six weeks.

Surfacing here is a gift to the person who asked, because the decision was otherwise going to get made by default, in the dark. Say: "We can, and here is the decision that comes with it. A month on search means the accessibility work and the two carryover bugs slip to next cycle. That is a fine trade if search matters more right now, but it is a trade, and I do not think it is mine to make. Whose call is that?" "Whose call is that?" is one of the most useful questions in technical leadership, because it forces ownership into the open before drift makes the decision.

When the decision hides across several requests

The harder case is a decision that no single request reveals. In one week you are asked by the mobile lead whether the new endpoint can tolerate older app versions, by support whether the old order screen can stay for another quarter, and by the product manager whether the widget work can start before the backend is final. Each is a reasonable question with a reasonable answer. Read together, they are scaffolding one unmade decision: whether the organization is going to maintain two versions of the order flow for the next six months, and who pays for that. Answer each request on its own and the two-version world arrives by accumulation, with no owner and no budget.

The move is the same three cuts, run on the pattern instead of the message: write the requests side by side, name the decision they share, and take it to the one person who can own it.

Not every request hides a decision

The move has its own failure mode. Treat every message as the tip of a buried decision and you become the engineer who cannot answer a simple question without a discovery meeting. "What is the status of the flaky test" is often a request for the status of the flaky test, and the third time you answer it with "help me understand what you are deciding," people stop bringing you questions.

The tell is weight. A request carries hidden weight when a wrong answer would commit someone to something, when the asker has more riding on it than the words suggest, when the timing is odd, or when your stomach tightens as you read it. The launch-date message had the first three. Over-read, and the owner says "no, I just wanted the status," which costs a minute; under-read, and it costs a commitment you never made. Err toward surfacing on anything costly to get wrong, and toward speed on everything else.

The hidden-decision worksheet

After a few months the three cuts run in your head. Until then, and for anything that matters, write them down in four lines before you respond; the appendix has the worksheet and a completed example.

The request, in the exact words it arrived in, because tidying it is where you smuggle in your own assumption. The decision, phrased as a choice between named alternatives: whether to commit the date externally, whether to fund search by cutting accessibility. If you cannot phrase it as a choice, you have not found it yet. The owner, named as a person or a role, never "the team" or "leadership"; if you are not sure who owns it, that uncertainty is the most important thing the worksheet surfaced. And the incentive that will settle it, stated without polish. Nobody else reads the worksheet, which is what lets you be candid, and the candor is for understanding what the owner needs, never for finding where to press.

The engineer who confirmed the fifteenth was short sixty seconds, the time it takes to ask what the question was for before answering it.

IN THE FIELD

The move. Surface: find the decision hiding under the request before you answer it.

The artifact. Hidden-decision worksheet, appendix template 1. Reach for it when a request feels heavier than its wording, the kind that makes your stomach tighten.

The signal. A week on, the owner you handed the decision to is the one carrying it, and no status answer of yours has turned into a commitment you did not agree to.

The level. A senior engineer catches the decision under a request aimed at them, like the date question, before answering it; a staff engineer notices when several requests across teams are scaffolding the same unmade decision and hands it to one owner. A principal puts a named owner on every external date, so no status answer can become a promise by default.

The boundary. Not every request hides a decision. Some asks are exactly what they say, and treating a simple question as a buried commitment makes you look evasive instead of careful.

The line. "Before I confirm, are you deciding whether to commit this date to someone outside the team? My confidence is different for the two."

APPENDIX, ARTIFACT 1 OF 16, THE TOOL FOR CHAPTER 4

Hidden-decision worksheet

What it is: a short worksheet that cuts past a visible request to the decision underneath it, the person who owns that decision, and the incentive that will settle it. It converts the pull to answer a task into the harder question of what is being decided and by whom.

When to use it: chapter 4. Reach for it when a request lands and answering it feels natural, especially a status check, a design review, or a confirmation that commits you without saying so to something.

When not to use it: when the request really is simple and the owner is obvious; answering fast is the right move.

Minimum viable version: four lines: the request in its exact words, the decision it scaffolds, its owner, and the incentive that will settle it.

Failure mode: you name a hidden decision the owner does not recognize and treat your guess as fact. Test it with one question before you act on it.

Completed example:

FIELD	ANSWER
Visible request (in its exact words)	"Hey, can you confirm we are still on track for the fifteenth?"
Decision it is scaffolding	Whether to turn a soft internal target into a hard external promise.
Who owns that decision	Director sponsoring the release, not the engineer confirming status.
Incentive that will settle it	How much the partner relationship matters to the director this quarter, and what slipping would cost him.
What I am really being asked to fund	Cover for a commitment I did not make, backed by my credibility.
Stronger response to give	Before I confirm, are you deciding whether to commit this date to someone outside the team? My confidence for an internal target and my confidence for a hard external promise are different numbers, and I want to give you the right one.
What changes once I surface it	The date returns to its real owner, and my status stops being used as a commitment.

PART THREE

Shift

change the decision

CHAPTER 8

Pre-alignment is not politics

Six people are on the call to approve a new on-call model, and before the call ends the approval is dead.

The proposal is not the problem, and you have probably sat on this call. The tech lead has a clean plan to move overnight paging for a shared service onto a single rotation instead of three. The document is tidy, the trade-offs are named, the handoff is sensible. Then the platform lead learns, live, that the plan routes his engineers into pages for code they did not write. The product manager learns, in the same minute, that the transition needs a two-week change freeze nobody warned her about. The reliability reviewer hears "temporary shared rotation" and asks who owns an incident at three in the morning when the rotation is ambiguous.

Now the call is not deciding anything, only discovering. Five people are learning the hard parts of the proposal for the first time, in public, and each new cost lands as a small ambush. The tech lead feels attacked, the stakeholders feel handed a bill they never saw, and the one person who could approve it looks at a room full of fresh objections and does the rational thing: defers.

The tech lead leaves believing the room was difficult. The room was fine; it had been asked to approve a decision it never had the chance to shape.

The meeting is the hardest place to persuade

The tech lead saved all the persuading for the moment it is hardest: five people at once, hearing the costs live, watching each other for cues.

In the reviews I have watched, people in a group often do not update the way they would alone. They protect face and take their cue from the most senior skeptic. Saying "let me think about that" can look slow, so they say "I have concerns" instead, and concerns seem to spread: one raised eyebrow from the platform lead can tell everyone else that this proposal is safe to resist. Laboratory research on conformity points the same way, though a design review is not a laboratory and this is my reading of the rooms I have watched.

Alone, the same person can ask the question they would never ask in a group, tell you the real objection, and move from no to maybe with no witness to the change of mind. Pre-alignment, an old practice that goes by several names, is doing the persuading where it is most likely to land: one conversation at a time, before the meeting. (figure 7)

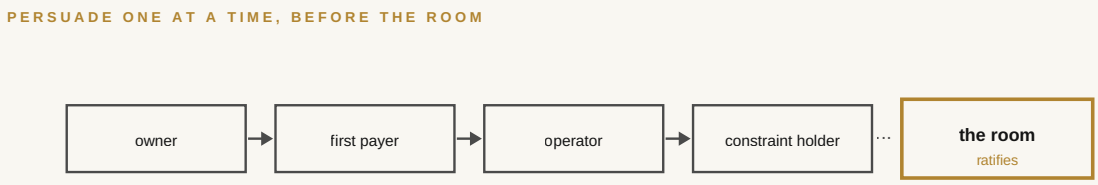

Figure 7. Persuade one at a time, before the room. The room becomes the last box, where the decision is ratified, not made.

Engineers often hear "work the room beforehand" and flinch, because it sounds like politics. The line is real, and it can be drawn precisely.

The line between preparing a decision and rigging one

Here is the test. If the private conversation would embarrass you to describe in the formal meeting, it is rigging. If describing it would make the meeting more honest, it is pre-alignment. Pre-alignment changes the proposal. Rigging changes the room.

Pre-alignment finds the missing constraint before it detonates in public. It learns that the platform lead's support model breaks and changes the proposal so it does not. It asks the reliability reviewer what would make the temporary rotation unsafe and builds in the ownership rule and end date she needs. By the time of the meeting, the real constraints are already in the proposal.

Rigging, the politics engineers are right to distrust, runs the other way: hiding an objection so it cannot be weighed, pressuring the one dissenter in private so they go quiet in public, lining up the most powerful person to overrule the operators who will carry the cost. It also happens before the meeting, which is why the two get confused. So keep one rule: every objection you hear in private either changes the proposal or walks into the meeting by name.

One tempting middle move fails the test: pre-aligning only with the people who already agree. The meeting gets friendlier but not safer, because the objection you avoided arrives late and armed with a grievance: the dissenter can see the process was built to route around them. See first the person most likely to change the proposal.

In written, async or RFC-driven organizations, pre-alignment happens in pre-reads and comment threads more than in one-on-ones, and the same tests apply to who gets the draft first.

Who gets the early conversation

The habit that makes pre-alignment work can also shut people out. Access to the early conversation is not evenly spread. The director you can catch after standup, the principal you already know, the people in your office and your time zone get a week to shape the proposal; the operator in another region, the junior engineer who will carry the pages, the contributor nobody thinks to invite meet it as a finished document. Run pre-alignment only through the people you can easily reach and you turn an unequal network into an unequal decision. The decision made before the meeting becomes a decision made without the people it lands on.

Four checks keep it on the right side of the line.

Access. Write down who saw the draft and when. Anyone whose work, pages or customers change because of the decision gets the material facts no later than the most senior approver does, in writing, with a comment deadline they can meet from another time zone. If the people who carry the cost hear last, you have converted the powerful first and left the affected to object against a coalition.

Reach. Go to the people who will not come to you. Ask the remote operator and the quiet reviewer the same question you asked the director, in writing: what fact, risk or constraint would make this unsafe to approve? Silence from someone who was never asked is not agreement.

Record. Every objection raised in private goes into the proposal, with the change it caused or the reason it caused none. Attribute it if the person wants the credit, leave the name off if they ask, but do not let it disappear into a one-on-one.

Dissent. Whatever is still unresolved walks into the meeting by name, in a short section of the proposal headed open objections, so the decider rules with the disagreement in view. Pre-alignment that removes dissent from the room instead of resolving it fails the disclosure test, however friendly the meeting is.

Some decisions should not be pre-aligned beyond sharing information. When the outcome's legitimacy depends on open deliberation, such as a change to on-call expectations that affects people's working hours, a policy that applies to a group with less power than you, a safety call, or a choice between values the room disagrees about, the persuading has to happen where everyone can hear it. Send the material early and to everyone at once, take questions in writing, and then stop. The work moves into the room. There is a reason

beyond fairness: in group discussion, people mostly talk about what everyone already knew, and facts held by one person often never surface unless the process asks for them. An early, equal pre-read is how those facts reach the decision at all.

Find the objection that would land last

For every name on the invite, ask whose late objection would legitimately change this decision. Most opinions belong in the review itself, where they cost nothing to hear. For the rest, the question is when their objection arrives. Live in the review, it is an ambush. Raised a week earlier in a one-on-one, the same objection is a design input. If any real cost in the proposal has no name attached, you have found the gap that will open in the meeting, in time to close it first.

The sentences that do the work in a one-on-one

A pre-alignment conversation is not a pitch. Present in someone's one-on-one and you have held a small version of the meeting you were avoiding.

Name what you are doing, because the openness is the point: "Before this goes to the review, I want to make sure the decision is ready. I am talking to the teams that adopt it, the people who operate it, and the person who signs off. I am not trying to settle this privately. I am trying to bring the real constraints into the review instead of discovering them there."

Then ask for the objection directly: "What fact, risk, or constraint would make this unsafe to approve?" In my experience people asked "are you aligned?" often say yes while carrying a blocking objection, because yes is polite and no requires a reason; research on employee silence finds the same reluctance to raise concerns upward. This question makes the objection the useful answer. When the platform lead says the support model breaks, you found it on Monday, two days before the review.

Follow with: "If that stays unresolved, should Wednesday be a discovery session, a review, or an actual approval?" Now the person tells you what kind of meeting is honest. A review that downgrades itself to "we will decide next week once the ownership rule and end date are designed" is pre-alignment preventing a false yes.

For the respected skeptic whose doubt carries weight, do not try to convert them in private; route their dissent into the meeting cleanly. "You clearly have reservations. I would rather you bring them Wednesday than sit on them. What is the strongest version of your objection, so I can make sure the proposal answers it?" That invites the disagreement and, by design, keeps it from arriving as an ambush.

The artifact: a pre-alignment plan

The object that holds this is small, and it is for you: a pre-alignment plan, appendix template 5. One row per person who could change the decision: who they are, why their objection would count, the evidence they need to say yes, the objection you expect, and the change you will make to the proposal because of it. Watch that last column. A row with an expected objection and no change next to it means you have predicted their unhappiness without pre-aligning them.

The plan's real job is to expose readiness. If three rows show unresolved constraints that would change the decision, Wednesday is not an approval, whatever the calendar says: fix them first, or ask the decision owner to relabel the meeting as discovery. The failure I see most often here is an approval booked for a decision that was never ready, and a room blamed for noticing. The plan fits on one screen and costs an afternoon of conversations, less than one failed review.

When a private yes is not a real yes

People sometimes give a yes in private that they will not defend in the meeting, because agreeing with you alone costs nothing and backing you in front of peers costs something. When all you hear is "yeah, that makes sense," ask: "Can I count on you to say that Wednesday, or is there something that would make it hard for you to?" The answer tells you whether you have a supporter or a spectator.

Then, before the review, send each person a short note restating what they agreed to and the change you made because of them. "You flagged the support model; the proposal now keeps your engineers off pages for code they did not write until the runbooks are handed over. That work for you?" The written restatement lets a soft yes firm up or wobble in private, which is where you want the wobble. Some will still wobble on Wednesday; you will at least know which ones, and why.

IN THE FIELD

The move. Shift the objections forward: take each real one into a one-on-one before the meeting, so the review weighs it instead of meeting it.

The artifact. Pre-alignment plan, appendix template 5. Reach for it before any decision that needs a group to approve it.

The signal. The review ratifies instead of discovering: the costs arrived a week earlier as design inputs, no objection lands as an ambush, and nobody who carries the cost heard about it last.

The level. A senior engineer pre-aligns the reviewers of their own proposal; a staff engineer seeks out the dissenter most likely to change a cross-team proposal and changes it before the review. The principal's version is structural: cross-team approvals do not get booked until every real objection has a name and a response.

The boundary. Some decisions need to be made in the open to be legitimate, and pre-aligning them privately reads as working the room. When the fairness of the process is the point, do the work in the meeting, not before it.

The line. "What fact, risk, or constraint would make this unsafe to approve?"

APPENDIX, ARTIFACT 5 OF 16, THE TOOL FOR CHAPTER 8

Pre-alignment plan

What it is: a plan for the conversations that happen before a review, so the meeting ratifies a decision instead of discovering it. It names the real purpose of the meeting and the objection each owner would raise if surprised in the room.

When to use it: chapter 8. Use it before any approval or cross-team decision where a late objection could derail a proposal that is technically sound.

When not to use it: when the decision is small and reversible enough to settle in the room; pre-aligning it only adds overhead.

Minimum viable version: the people whose late objection would change the decision, and the one question you will ask each of them.

Failure mode: pre-aligning only with allies. The plan is honest only if the likeliest dissenter is on it, the people who carry the cost see the draft no later than the approver does, and every unresolved objection goes into the meeting by name (chapter 8).

Completed example. Context: a platform team wants approval for a shared service template.

FIELD	ANSWER
Decision to be made (as a choice)	Whether to approve the shared service template for wave-one adoption now, with exception rules and a support model, or defer it a cycle.
Decision owner	Engineering director.
Meeting purpose: discovery, review, or approval	Review, not approval, until the support and security constraints below are resolved.
Who has seen the draft, and when	Director, product team lead, operations lead and security reviewer, all on Monday, with written comments open until Wednesday noon.
Open objections carried into the meeting	Whether wave-one timing displaces a committed customer release. Raised by the product team lead in private; it goes into the room by name.

STAKEHOLDER	PRODUCT TEAM LEAD
Why they matter	Must adopt in wave one
Evidence they need	Migration effort and support model
Likely objection	This displaces customer work
What changes before the review	Move the team to wave two or fund one sprint

STAKEHOLDER	OPERATIONS LEAD
Why they matter	Operates failures after rollout
Evidence they need	Recovery path and runbook
Likely objection	Operations inherits unknown behavior
What changes before the review	Add operations sign-off and a runbook

STAKEHOLDER	SECURITY REVIEWER
Why they matter	Defines non-negotiable controls
Evidence they need	Template handles secrets and audit
Likely objection	Exception path is too vague
What changes before the review	Add a temporary exception with an owner, a reason, an expiry date and a migration plan

PART FOUR

Secure

make the outcome hold

CHAPTER 15

Escalation without status warfare

The message goes out at 4:40 on a Friday, and it works, which is the problem.

Months ago the payment provider announced it will retire the API version the mobile app depends on. After the cutoff, the provider can stop processing those calls, and checkout fails for anyone still on a build that uses the old SDK. The client work is done. The server side is not: the new SDK needs a token endpoint and webhook changes owned by the payments platform team, which has spent five weeks on an incident follow-up its director considers more urgent. And the fix reaches users only after a build clears App Store review, which nobody here can schedule, then rolls out and gets installed.

The message names the blocker in the first line and copies two directors and the launch sponsor. Within the hour, people who had let the ticket sit are replying with dates.

It also cost him something no dashboard shows. The lead whose team owned the endpoint feels ambushed, her manager concludes that mobile ran straight to the executives, and over the next twenty replies the thread turns into an argument about who dropped the ball. The blocker moved. The working relationship he will need for the next three launches got more expensive.

It is a pattern I have seen often: a strong engineer wins the moment and loses the position. The escalation itself was justified; the problem was how it was sent.

Escalation is not the problem

Escalation is moving a decision to the owner who can make it. When a risk crosses team boundaries and the people at your level cannot trade the priorities that would resolve it, someone with broader authority has to choose, and putting that choice cleanly in front of them is ordinary senior work.

Status warfare uses visibility as a weapon. From outside it looks almost identical, but its purpose is to make an audience see who failed, and it does lasting damage because it works in the short term. One question separates the two: is this message asking one owner for one decision, or asking an audience for a verdict on a person?

The engineer in the cold open got there the ordinary way: he waited too long, then escalated too hot.

Too late, then too loud

In the teams I have worked with, technical leaders more often escalate late and hot than early and calm, and the lateness produces the heat.

First comes the patient stretch: a soft reminder, a private nudge, and the cutoff eating the weeks he needed for review and rollout. By the time he escalates, the message carries five weeks of resentment. "This has been sitting with platform for five weeks" is an accusation in the shape of a status update, and everyone copied can feel it.

The waiting felt like professionalism. Every week of it moved the risk downstream, onto the release, the customers on the old build and the on-call engineer on cutoff morning, none of whom got a vote.

So escalate earlier, smaller and cleaner: on a trigger named in advance, about one decision, in a form the owner can decide without reconstructing the conflict.

Find the decision before you find the recipient

Escalation is Surface performed under pressure. The common failure is asking "who is senior enough to make this move?" before answering "what decision am I asking anyone to make?"

Seniority is not ownership. A capacity trade-off between two teams belongs to whoever owns both priorities, here the launch sponsor, not the VP three levels up. Route a decision to someone who cannot make it and you have created noise.

Surface the other side's risk too, because skipping it is what turns escalation into warfare. The platform team is protecting an incident follow-up that exists because something already broke once. Erase that to spotlight your risk and you leave them no choice but to defend themselves in public. Name both and the temperature drops: checkout on current builds is at risk if the token endpoint is not in staging this month, the incident follow-up is at risk if an engineer is pulled off it, and the decision is which risk the organization will carry. Nobody has to have failed.

The vendor changes one thing: nobody inside the company controls the provider's cutoff. The real deadline is the last day you can submit a build and still, allowing for a slow review or a rejection, reach enough of your users before the cutoff. That day comes weeks earlier.

The brief that carries the choice

The escalation brief, appendix template 13, travels to people who were not in the conversation and forces a choice instead of an opinion. The field most people leave out is prior attempts: what has been tried and why the current group cannot close the decision on its own. Without it, the escalation reads as first contact with a bigger audience.

Compare the two as spoken. The weak one: "Looping in leadership because the payments migration has been blocked by platform for five weeks and the provider cutoff is coming." A villain and no decision. The strong one: "We have a priority conflict between the payment SDK migration and the platform team's incident follow-up. The provider retires the API version our app uses on the fourteenth of next month, and that date is not ours to move. To leave room for review, a possible rejection and a phased rollout, the token endpoint has to be in staging by the twenty-first of this month. The decision is the sponsor's: whether platform assigns one engineer to the endpoint this sprint and slips the follow-up by two weeks, or the sponsor accepts that older builds lose payment at cutoff and we ship an upgrade prompt, with in-app payment disabled on the old builds. The two leads reviewed this on Tuesday and neither can trade the other's priority; both managers have been consulted. My recommendation is the first option, because the second pushes a payment failure onto customers who did nothing wrong. Please choose by Wednesday noon so the release plan uses the real scope."

Attention buys minutes, a decision buys the outcome

Sometimes you have no clean decision to ask for, only a wish for the thing to move. Then ask for attention directly, in a small room: I want this cutoff on your radar before it becomes a release problem, no decision needed yet. What poisons the well is dressing a plea for attention as an escalation, because an audience with nothing to decide will decide who is at fault instead. The test before you type a name into the recipient line: can I state, in one sentence, the decision I am asking one person to make? If not, you may still need to raise the issue, but you are not escalating yet.

The threshold you name in advance

With a vendor cutoff it writes itself backward from the date nobody can move: the cutoff, minus review time with room for one rejection, minus rollout and the time it takes enough users to update, minus the time the endpoint needs in staging. When the gap cannot close at your level, the threshold becomes a ladder: the other lead first, then a brief to the sponsor, who decides, with both managers consulted before it goes. Said to the other lead weeks earlier, in calm weather: "The provider's date is fixed, so I worked backward from it. If the endpoint is not owned and scheduled by Monday, we lose the window to ship a build before cutoff, and I will take a brief to the sponsor to decide, with both managers consulted. I wanted you to hear it from me first." (figure 11)

Figure 11. Escalation is a ladder, not a hierarchy. Each step carries the named threshold that triggers the next.

"If this keeps slipping I will have to escalate" is a threat with a soft voice, and it can never fire cleanly because nobody knows when it has been crossed.

The follow-up checks whether the risk moved and whether the relationship survived. If the other lead felt surprised anyway, repair it directly. If your escalations only work when they carry public pressure, the pattern is broken even when each message succeeds.

When raising it is punished

Everything above assumes a decision path exists and that using it is safe. Chapter seven covers the decision nobody can legitimately make. This section covers the other failure: there is an owner, and raising the risk with them costs you.

A staff engineer finds that a release planned for the last week of the quarter ships a login change with a known weakness: sessions stay valid after a password reset. The security review has rated it high. The director who owns the release has a customer demo tied to the date and has said in planning that he does not want another security fire drill. She writes a clean brief: the issue, the evidence from the security review, two options (fix the sessions and slip four days, or ship on the date with the new login flow switched off), the risk owner, a deadline. The reply calls her alarmist. The next week she is off the release sync, a peer mentions that the director called her not a team player in a calibration discussion, and the release plan has not changed.

Transparent escalation stopped being safe when the person who decides became the person punishing the question. A sharper brief to the same address will not fix that; the moves in this book assume a good faith that is no longer on the table. What the organization can reasonably expect from her has narrowed too. It can expect her to have raised the risk once, clearly, in writing, to the right owner, and she has. It cannot expect her to keep spending her standing on it alone, or to absorb the risk herself by fixing the sessions over a weekend so nobody has to decide.

Her moves now are about the record and the right channel. She keeps a dated, factual account of what she raised, to whom, what came back, and what changed for her afterward, kept within the company's rules on confidential material, because someone else will need to rely on it. She checks whether the risk has a path that does not run through the director: a high-rated security finding usually has its own route, and whoever owns that route can decide whether the release ships with it. She tells her own manager, separately and factually, both things: "I raised a high-rated security risk on the login release, and since then I have been taken off the release sync. I need both looked at, separately." Where the treatment looks like retaliation for raising a risk, HR, an ethics or compliance line, or the company's whistleblowing process is the channel built for it. Which one fits, and what protection it offers, depends on the company and the jurisdiction; that is a question for those channels or an employment lawyer, not for this book. Allies help when they stand on the facts: the security reviewer who rated the finding, a peer who saw the same evidence. Asking people to take sides against a director is a different request, and usually a worse one.

Some answers are refusals. She can decline to sign the launch readiness as the person who accepts the risk, because accepting it was never hers to do. And sometimes, after the formal channels have run, the honest conclusion is that this organization punishes people who tell it inconvenient things, and the right move is to leave. That is not the method failing in her hands. It is the method's limit. Surface, Shift and Secure work inside an organization that, however clumsily, wants to know where its risks are. They cannot repair one that has decided it does not, and a better brief is no substitute for a safer place to work.

IN THE FIELD

The move. Secure the escalation before you need it: name the threshold while things are calm, then escalate earlier, smaller and cleaner when it fires.

The artifact. Escalation brief, appendix template 13. Reach for it when a cross-team risk is stuck and no one at your level can trade the priorities that would resolve it.

The signal. After a week, an owner decided, the teams stopped debating status, the dependency plan changed, and the other lead did not feel ambushed.

The level. A senior engineer raises the blocker early with evidence and no villain; a staff engineer turns it into a brief that names both teams' risks and asks the one owner who can trade them, on a threshold announced in advance. Principal work sits upstream: vendor deprecation notices get an owner and a backward-planned date when they arrive, so the late, hot escalation stops being needed.

The boundary. This assumes a decision path that exists and is safe to use. When no one owns the decision, chapter seven applies. When raising a risk is punished, stop sending better briefs to the same address: keep a dated record, use the risk's own route and your manager, take retaliation to the channel built for it, and decline to accept a risk that was never yours.

The line. "If the endpoint is not owned and scheduled by Monday, I will take a brief to the sponsor to decide, with both managers consulted. I wanted you to hear it from me first."

APPENDIX, ARTIFACT 13 OF 16, THE TOOL FOR CHAPTER 15

Escalation brief

What it is: a brief that escalates a decision rather than a frustration, naming the choice, the right risk owner, the evidence, the prior attempts, and real options. It gives the recipient something they can decide, within their authority, by a deadline tied to the risk.

When to use it: chapter 15. Use it when a cross-team blocker needs a decision from someone outside the working group and private coordination has stalled. It is never the first contact, and never a way to shame another team.

When not to use it: in an organization that punishes honest escalation. Protect yourself first: document, and seek formal support.

Minimum viable version: the decision needed, two options, the risk owner, and the date after which not deciding is itself a decision.

Failure mode: a frustration dressed as a decision. If the brief has no options, it is a complaint.

Completed example. Context: checkout migration is blocked by identity integration capacity, and both teams have valid priorities.

FIELD	ANSWER
Issue (one sentence)	Checkout migration readiness depends on identity integration support this week.
Decision needed (as a choice)	Whether identity funds support, checkout reduces launch scope, or the sponsor accepts readiness risk.
Risk owner	The launch sponsor.
Evidence	The integration test cannot run without identity test credentials and a review slot. Identity's own risk is real too: the engineer who would help is on its committed login-reliability work, which slips if they are moved.
Prior attempts (factual)	Leads reviewed options Tuesday. Identity cannot commit without a manager priority trade-off.
Option 1	Reassign identity capacity this week.
Option 2	Reduce checkout scope to paths that do not require new identity behavior.
Option 3	Keep scope, with the sponsor accepting the migration readiness risk by name.
Recommendation	Option 2 unless identity capacity is explicitly reassigned by Wednesday.
Requested action and deadline	The sponsor chooses an option by Wednesday noon, before the readiness review; both managers are consulted.

BEFORE THE ROOM

What the full book covers

- 20 chapters in five parts: The gap, Surface, Shift, Secure, Make it a system.
- Every chapter ends with an In the field block: the move, the artifact, the signal, the level, the boundary, and the line to say.
- 16 artifacts in the appendix, each with when to use it, when not to, a minimum viable version, its failure mode, and a completed example. 12 figures.
- Three sections where the method fails on purpose: an objection misread as an incentive, a decision nobody can legitimately make, and an organization that punishes the person who raises a risk.
- 25 notes on the sources, and an operational index of 22 problems, each with its chapter, artifact and the line to say.

The Field Kit

The Field Kit includes editable versions of the artifacts used in this book, blank templates, completed examples, the field card and the thirty-day repair workbook.

The team edition

The team edition licenses the book and the Field Kit for up to 10 people inside one organization.

Before the Room

A field guide to leading outcomes you have no authority to command

20 chapters · 16 artifacts · 12 figures · PDF and EPUB

BUY THE FULL VERSION

mikesalari.gumroad.com/l/before-the-room

salari.dev/books/before-the-room

Staff mobile engineer · 500+ technical interviews from the hiring side