ESPROFILER

NEW IN THE SEAT

TOOL SPRAWL GUIDE

For security leaders new in post,
dealing with M&A,
or trying to get in front of
tool sprawl.

FIELD GUIDE

You have inherited a sprawling security stack.

You did not build it. You did not choose it. You are accountable for the budget on it anyway. This is the process I would give to anyone in that seat, and you can run every step of it without buying anything.

ABOUT THIS GUIDE

This is a field guide, not a white paper. It sets out, in order, how to get on top of a security stack you did not choose: build a single view of what the organisation owns, work out where your decisions actually fall, find out from the people using the tools what those tools are really doing, and then fix the operating model so the sprawl does not quietly build back up behind you.

It comes out of the same pattern repeating across consulting engagements and incident reviews. Every step is one you can run yourself, with nothing bought. Where ESPROFILER helps is set out at the end, fenced off and clearly signposted, so you can skip it.

WHO IT IS FOR

Security leaders in their first year in post, and anyone who has just inherited a stack through a merger or a reorganisation.

WHAT IT COVERS

Eight steps, from the first central view of what you own through to a standing renewal forum that keeps it honest.

HOW TO USE IT

Read it once end to end, then work the steps in order. Each one produces something you keep and carry into the next.

NAVIGATION

Contents

SUMMARY

The process at a glance

Eight steps. The first one is the hardest, and it is the most significant.

1. Change nothing. Spend the first weeks listening, not reorganising.
2. Build a whole-organisation view of every tool providing security capability: owner, cost, commercial end date, notice period.
3. Add a replacement time estimate to each tool to produce decision windows, then prioritise your attention by them.
4. Interview the orbit around each tool, not just the owner. Score sentiment, utilisation, coverage and reliance out of five.
5. Hold all of it in a capability inventory, which is not the same thing as a list of products.
6. Find structural overlap by category, then capability overlap using the Cyber Defense Matrix.
7. Stand up a renewal forum that runs on decision windows and issues invest, maintain or divest briefings.
8. Make the inventory the first gate for every new requirement the business raises.

CONTEXT

Why I am writing this

You did not build it. You did not choose it. You are accountable for the budget on it anyway.

That is the position most security leaders find themselves in during their first quarter, and it is an awkward one. You arrive with a clear idea of what good looks like and almost none of the money you need to act on it, because the money is already committed to decisions other people made years ago.

Something worth internalising early: freeing up existing budget is usually easier than persuading a board to hand you more. Conveniently, the work of freeing it up is the same work that makes you credible in front of that board. You cannot argue well for new investment until you can describe, in detail, what the current investment is actually delivering.

This guide is about how to do that. I represent ESPROFILER and this problem is what we build for, so I am not going to pretend neutrality about the destination. But the process below is the one I would give to anyone, and you can run every step of it without buying anything from us. I have put where we help at the end, clearly signposted, so you can skip it if you would rather.

CONTEXT

Who I am, and why I have a view on this

My background is a slightly unusual one. I started in cyber forensics and future strategy in the UK's counter terrorism unit, moved into the computer emergency response team at BT Global, and then spent years at Symantec helping its largest customers extract more value from security stacks they had already bought. Later I did the same work across the channel, spanning sectors and vendors, before founding ESPROFILER because I was tired of watching the same thing happen over and over.

Consulting gave me a vantage point that people inside organisations rarely get. I could sidestep the usual silos: a conversation with the CISO in the morning, a session with SOC analysts after lunch, then time with the engineers running the technology. When something needed to happen, I would be in the customer's datacentre physically racking security tin to make sure it did.

In almost every engagement and every incident I have worked, there was a sprawling list of security tools already in place. And in almost every post-incident review, someone at ground level would quietly pipe up: I knew this would happen, because of X. Nine times out of ten, X was a capability that had been bought and poorly deployed, or a feature set nobody had ever turned on. That is not acceptable to me, and it should not be acceptable to you.

When I first moved into consulting, the other consultants used to joke that their careers consisted of putting security capability into the same companies and then ripping it out three years later to replace it with something else. Factor in project change cost and the waste is staggering. The interesting question was always why. Usually, it came back to the same root: the organisation never extracted the value from what it had, frustration built, and the perception took hold that switching to something else would be better. Then the cycle repeated.

CONTEXT

Why the sprawl exists, and why it is not your fault

Tool sprawl is not a symptom of bad leadership. It is what happens when ordinary organisational forces run for a decade without a counterweight.

- **Mergers and acquisitions.** You inherit somebody else's stack alongside their people.
- **The change of guard.** Every leader before you bought to solve the problem in front of them at the time.
- **Multi-year subscriptions.** Renewal decisions sit with individual teams, on their own timelines, invisible to anyone else.
- **No capability-level front door.** When a new requirement lands, there is nowhere to ask "do we already own something that does this?" at a feature level rather than a product level.

The last one is getting worse, because security capability is quietly propagating into platforms that were never in the security budget. GitHub and GitLab now ship SAST, DAST and package security as part of the platform, covering ground that historically belonged to a Veracode or a Snyk. If your view of the stack stops at the boundary of your own cost centre, you will miss it entirely.

FOUNDATION

Before anything else: do not change anything

The temptation when you walk in is to reach for what worked last time. You know a vendor. You know a tool. You know the architecture that fixed this exact problem at your last organisation.

Resist it, at least for now.

Changing things in the first weeks sends a message you cannot take back: that the views of the people already here were not worth hearing. That is the fastest way to alienate the doers in your organisation, and the doers are the people whose cooperation you will need for everything that follows.

> Making the same change three months later, after those conversations, lands completely differently. You have demonstrated that you listened, and you can articulate the reasoning behind the decision. People will get behind a decision they can see the logic in, even when they disagree with it.

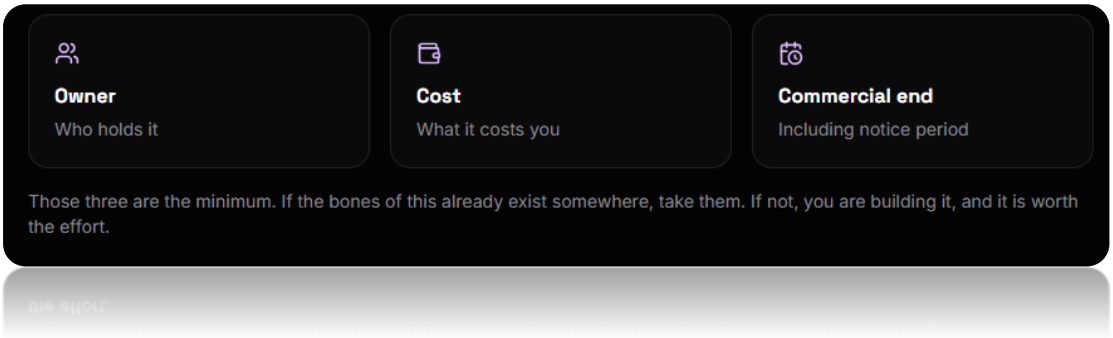
STEP ONE

Build the central view

Start with a single view of every tool providing security capability across your organisation.

For each one you want, at minimum:

- Who owns it
- What it costs
- When it commercially ends, including any notice period



The three fields that make the view usable. Everything else is an improvement on top of these.

The critical instruction here is scope. Build it as a whole-organisation view, not a "what is in my budget" view. I have seen it enough times that other departments hold capability you can leverage, sometimes capability that lets you free up budget of your own and redeploy it elsewhere. Mark clearly what falls inside your budget and what does not, but do not let the budget boundary define the edges of the exercise.

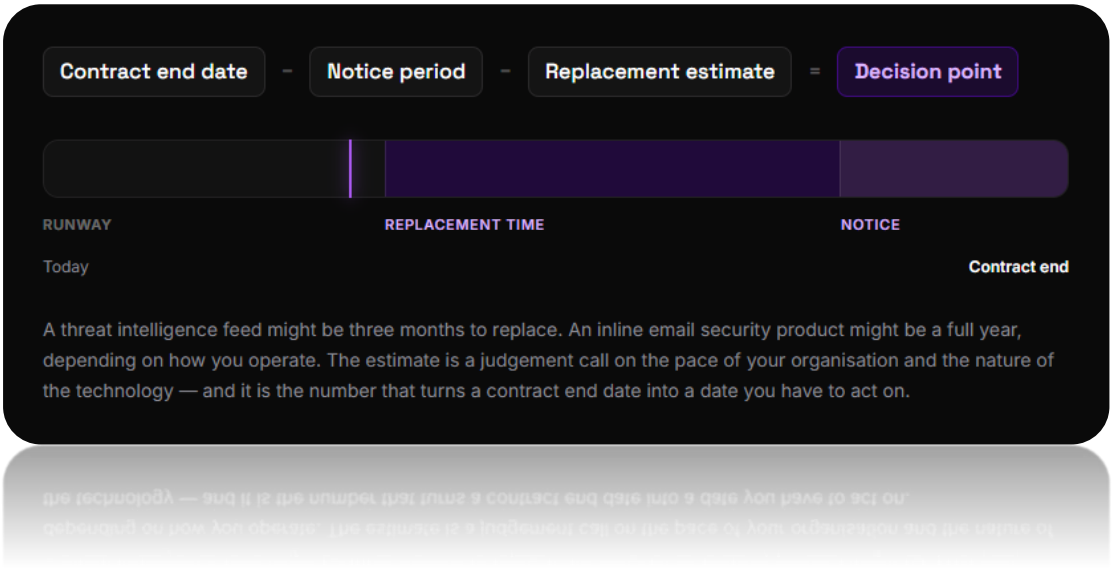
STEP TWO

Work out your decision windows

Now take every tool inside your budget and add an estimate: how long would it realistically take this organisation to replace it?

That number is a judgement call based on the pace of your organisation and the nature of the technology. A threat intelligence feed might be three months. An inline email security product might be a full year, depending on how you operate.

> **Contract end date, minus notice period, minus estimated replacement time, equals the point at which a decision has to be made.**



Runway, replacement time and notice period, laid against the contract end date.

Do that across the estate and you have something useful: a time-prioritised list of who to go and talk to first. Not everything at once, which is overwhelming and unachievable. Just the conversations that are about to matter.

STEP THREE

Talk to the orbit, not just the owner

Every tool has an orbit of people who interact with it directly and indirectly, and any single viewpoint is partial.

Take an email security product. The mail team cares about maintainability, uptime and delivery. Your SOC analysts care about log output and enrichment quality for investigations. Your security engineers care about rulesets and how much flexibility they have to deploy what they need. Ask only the owner and you get a third of the picture.

What to ask

Sentiment. What do they actually think of it? How is the usability? How is the vendor support? Do they believe better alternatives exist in the market, and if so, what makes those alternatives better?

Implementation. How is it deployed, and how is it being used day to day? Is anything about the implementation worrying them? Are you using all the feature sets available, and if not, why not?

Coverage. Where applicable, is it covering all the assets it should be?

Then the two questions that do the heavy lifting:

- If it were entirely your choice, would you keep it?
- If I removed it tomorrow, what would break, and what would you have to do without?

Tie the second one back to business functions and critical services wherever you can. These two matter because they put the person in your seat. You are no longer asking them to review a tool, you are asking them to make your decision, and people answer that differently.

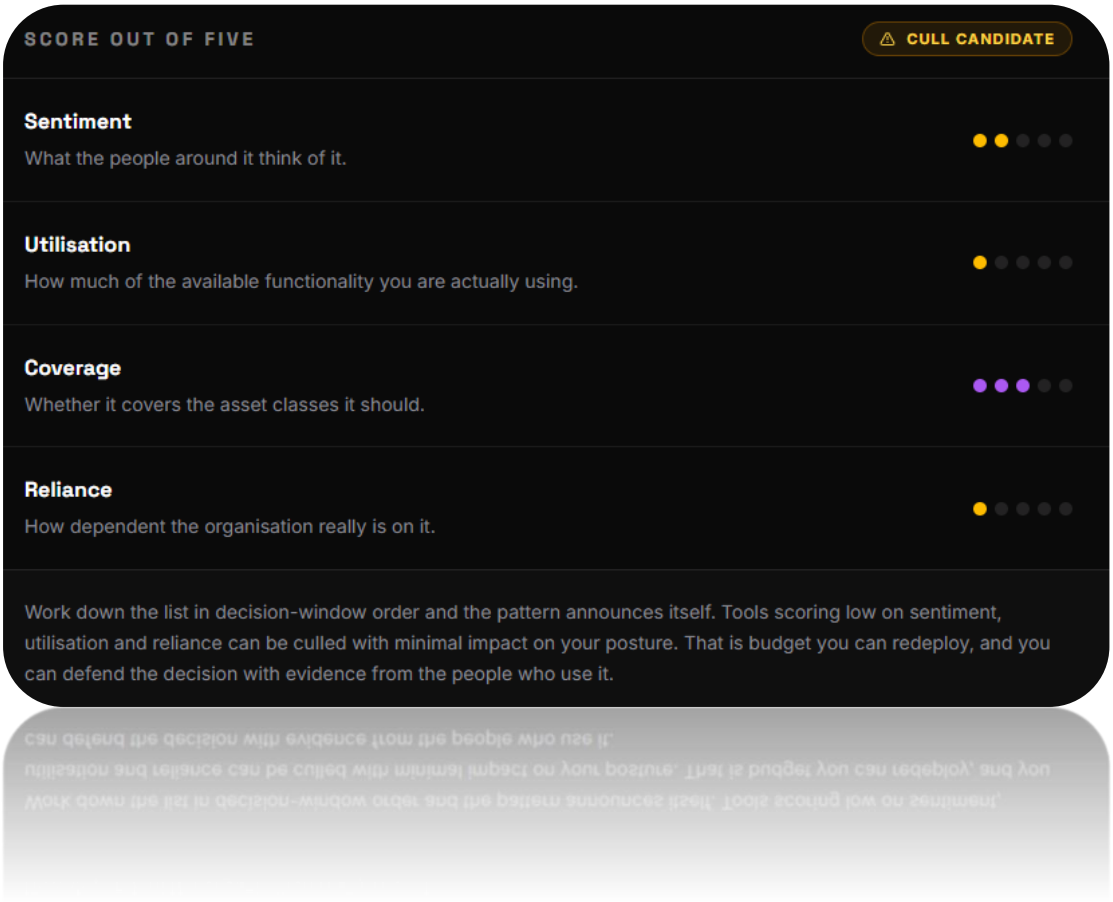
> Expect a chasm between what you thought you had and what you actually have. This is the most consistent finding across every one of these conversations I have had, and it is where most of the value in the exercise sits. It is also, precisely, what surfaces in the post-incident review you do not want to be sitting in, when someone finally says out loud that they knew this would happen.

Score what you hear

Capture high-level notes for future reference, then score each tool out of five on four dimensions:

- **Sentiment.** What the people around it think of it.
- **Utilisation.** Your view on how much of the available functionality your team is actually using.
- **Coverage.** Whether it is covering the asset classes it should be.
- **Reliance.** How dependent the organisation really is on it, drawn from those last two questions.

As you work down the list in decision-window order, the pattern announces itself. Tools scoring low on sentiment, utilisation and reliance can be culled with minimal impact on your security posture. That is budget you can redeploy, and you can defend the decision with evidence from the people who use it.



Four scores, one line each. Low across sentiment, utilisation and reliance is a cull candidate.

A practical note on unfamiliar tools

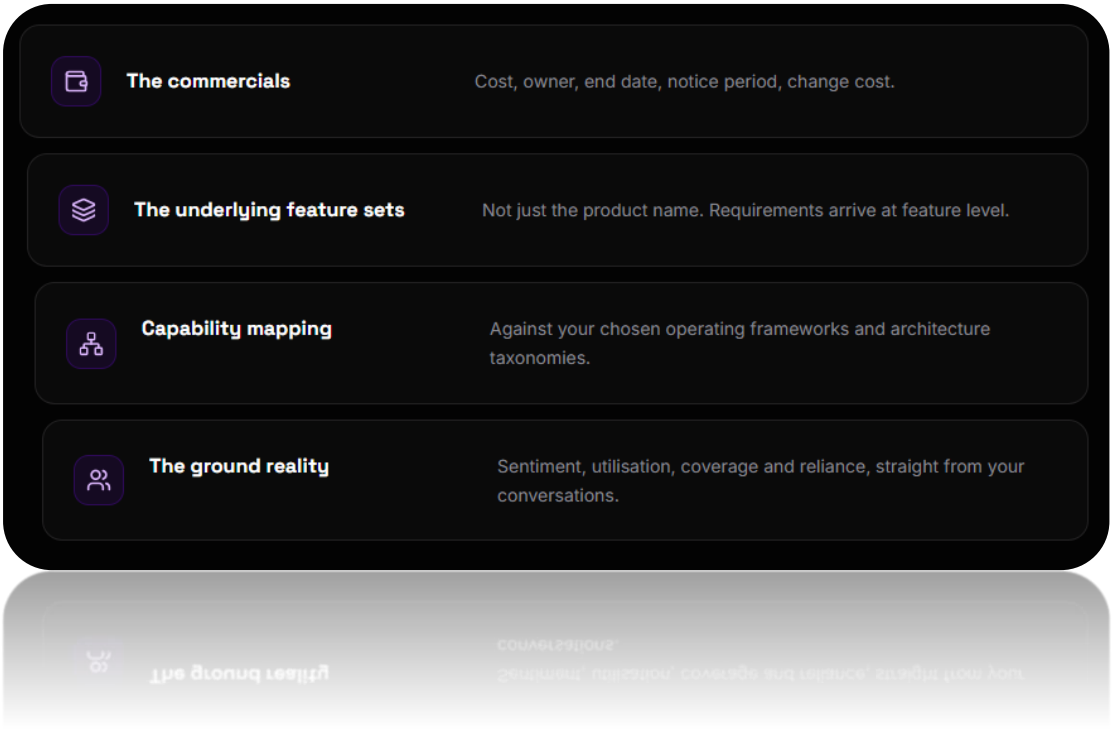
Every organisation I have worked in has had tools in it I had never seen before, and they took time to go away and understand. We track over 27,000 products globally that provide security capability, and we publish a free, condensed, no-nonsense view of what each one is and does at capability.exchange. Familiarise yourself with a tool before you go and interview people about it.

STEP FOUR

The capability inventory

Everything above needs to live somewhere, and that somewhere is a capability inventory.

This should not be a static list of tools. A list of tools cannot answer the questions you are going to be asked. The inventory needs to combine:



Commercials, feature sets, capability mapping and ground reality, held in one place.

Think of it as your capital deployment map: what you have, what it costs, what it is really doing, and when you next have to make a decision about it. It also becomes the first gate for any new capability requirement coming out of the business.

> *Keep it current and it does two jobs at once. It is your defensibility insurance when someone asks why the money was spent this way, and it is the evidence base you carry with you when you go and ask for more.*

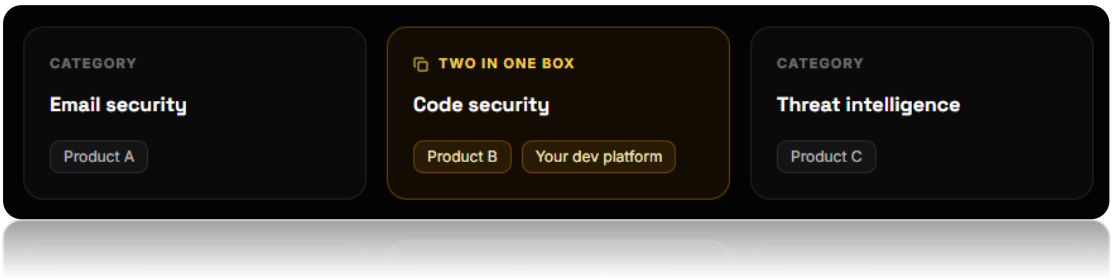
STEP FIVE

Find the structural overlap

The decision-window exercise gives you a manageable hit list and builds relationships along the way. In my experience most people appreciate being asked about their problems and where they would like to see improvement, and that has been the single biggest value driver for me across my consulting career.

But ground-level conversations have a blind spot. Teams work within the orbit of the tools they need, which means they are usually unaware of overlap sitting outside that orbit, in another team or another department entirely.

Start structurally, because it is the easier of the two overlap problems. Map each product to the categories it sits in. If your organisation has an architecture taxonomy, that is a good starting point. If it does not, you can use the framework section on capability.exchange/frameworks and use our ESP Products Taxonomy for free.



Any category holding more than one product is your second hit list.

Always consider a primary and a secondary mapping, because products rarely sit neatly in one box. Then look for any category holding more than one product. That is your second hit list, and you go and repeat the ground conversations against it. What comes out is a view of where structural overlap exists and what could be divested or consolidated.

STEP SIX

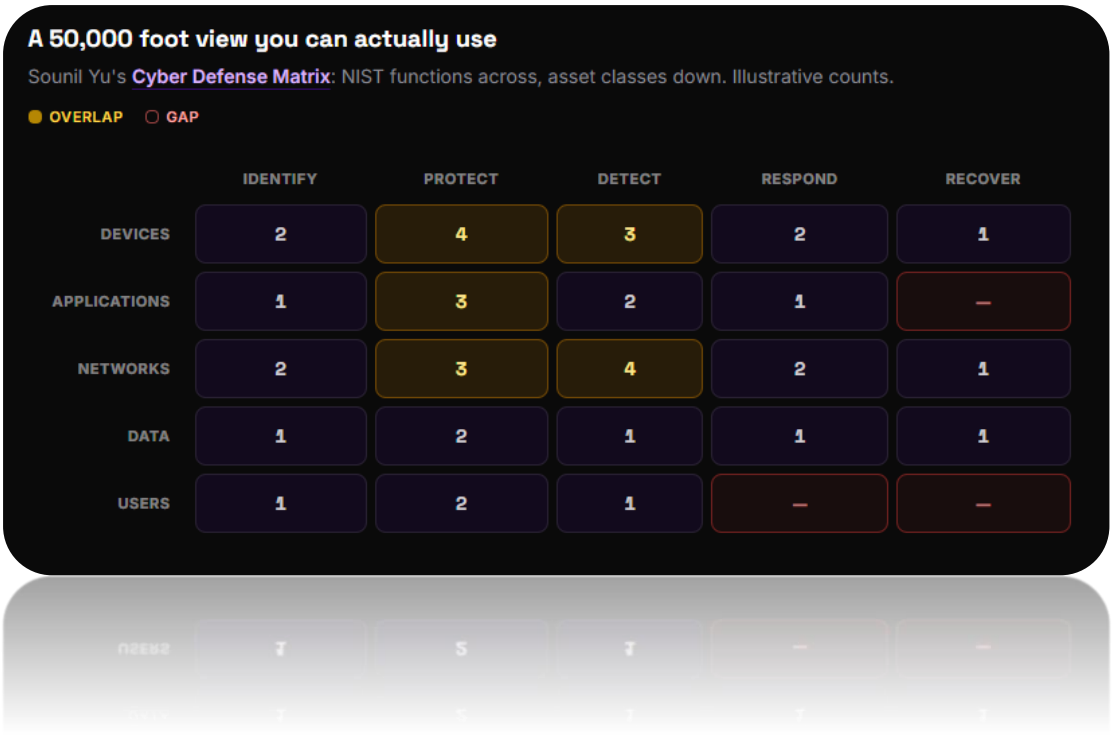
Find the capability overlap

Capability overlap is trickier. The security market does not help here, and I often joke that it resembles the supplements aisle, with everything claiming remarkable benefits and nothing making it easy to see where the real overlaps and gaps are.

Frameworks like MITRE D3FEND give you a defensive capability view aligned to attacker methodology in ATT&CK. We support it, but starting there manually, or even with AI assistance, will overwhelm you quickly.

A better starting point is Sounil Yu's Cyber Defense Matrix. It uses the NIST functions as its pillars and subdivides by asset class, which gives you a 50,000 foot view that is actually usable. Align your stack to it and you can start answering questions like "how many tools do I have protecting employee devices?"

AI can help a great deal with this mapping. What matters is not the tooling you use to do it, it is that you are approaching overlap from a capability angle rather than a category angle. Map each tool with a primary relationship, meaning the first thing it does, and secondary relationships, meaning anything else it strongly supports. I would recommend buying Sounil's book, because the same exercise is as good at exposing gaps as it is at exposing overlap.



The Cyber Defense Matrix: NIST functions across, asset classes down. Overlap and gaps in the same view.

Keep the limitation in mind. This view is high level, and plenty of detail lives in the weeds where the framework cannot see it. Being able to work across multiple frameworks and taxonomies at once is exactly why we built that into our platform. For the purposes of this guide, the matrix plus your ground-level outreach will cover both the structural and capability angles well enough to give you solid footing.

STEP SEVEN

Fix the operating model so it does not happen again

A question I always ask is how the organisation got here in the first place.

The answers are usually nuanced. Mergers and acquisitions. Successive changes of guard buying point solutions for problems that incumbent platforms eventually solved anyway. But the most common answer, and the least discussed, is simply the operating model: how renewals get decided and who decides them. Close behind it is the absence of a maintained, central understanding of what capability the organisation actually holds.

The worst model

A fully decentralised one, where teams and product owners have the only say on their own renewals.

They should absolutely have a say. But it cannot be the only say, for every reason set out in this guide. A product owner does not have visibility of capability elsewhere in the organisation, and does not have sight of the licensing agreements that could be leveraged for better commercials.

The best model: a renewal forum built on decision windows

Take the same calculation from step two: commercial end date, plus notice period, plus estimated replacement time. That gives you a line in the sand. Apply a grace period in front of it, and you have a decision window.

The forum runs on those windows. Cadence follows volume, so the number of products and the number of decisions falling due each month will tell you whether you meet monthly or quarterly.

WHO SHOULD BE IN THE ROOM

- The product owner
- Architecture
- Someone holding portfolio decisions, meaning a view of the wider commercials and strategy. This role is not always formally defined, but it usually exists in large organisations.

The point is an audience that collectively understands what the business needs, where the architecture is heading, and what the wider budget and commercial picture looks like.

WHAT THE FORUM WORKS THROUGH

First, has overlap appeared? You are dealing with a dynamic set of products, most of them adding features and entire new offerings every quarter. The justification for buying this tool may have been completely sound at the time and may have solved a real requirement. But other tools in your stack may since have developed the same capability. A current inventory is essential here, because it is the reference point.

Second, go back to the ground. Reach out again to the stakeholders in the tool's orbit, following the same lines of enquiry as step three. Is it still meeting requirements? Is vendor support still up to scratch? Are they happy with the roadmap? The decision window is an opportunity to put your ear back to the ground and avoid renewing something the team has quietly stopped believing in.

Do not stop at the product owner. Which tool it is determines who you need to hear from. For a code security product, you want at least one conversation with an end developer about whether it is causing friction. For an email product, the owner's view on delivery and the SOC's view on log and hunt quality are two entirely different assessments.

Use this step to rate the maturity and implementation quality of the tool as well. Done at the cadence of your decision windows, this gradually builds a picture of overall stack maturity and risk.

Third, look outward. With an updated view of requirements and ground feedback, ask what else exists in the market. Start with the vendors flagged as strategic in your capability inventory: have they developed something that could serve as a replacement, and would it suit the requirements and the architecture direction? Then extend to a wider market search regardless of strategic vendor status, because that is how you stay in tune with better alternatives.

The output: invest, maintain, divest

Each decision window should produce a briefing with a recommended status.

- **Invest.** The business should put more into this tool: broader rollout, additional feature sets, additional modules. The recommendation comes from the ground, from architecture and from wider business requirements.
- **Maintain.** The business is happy with the current solution as it stands.
- **Divest.** The product should be removed. If something is replacing it, name what.

Divest carries an obligation. Technology change has a cost, and it needs to be defined and factored in, because a tool can be flagged for divestment while the budget and resource to actually make the change are not available. Track a change cost estimate against every tool, based on the estimated replacement time plus any CAPEX where hardware is involved.

> One last point on divestment, and it sounds obvious right up until it happens to you. Once the decision is made and the project is under way, stay on top of the notice period and keep procurement looped in. There is nothing more frustrating than completing the migration and then discovering you missed the notice window and are locked in for another year.

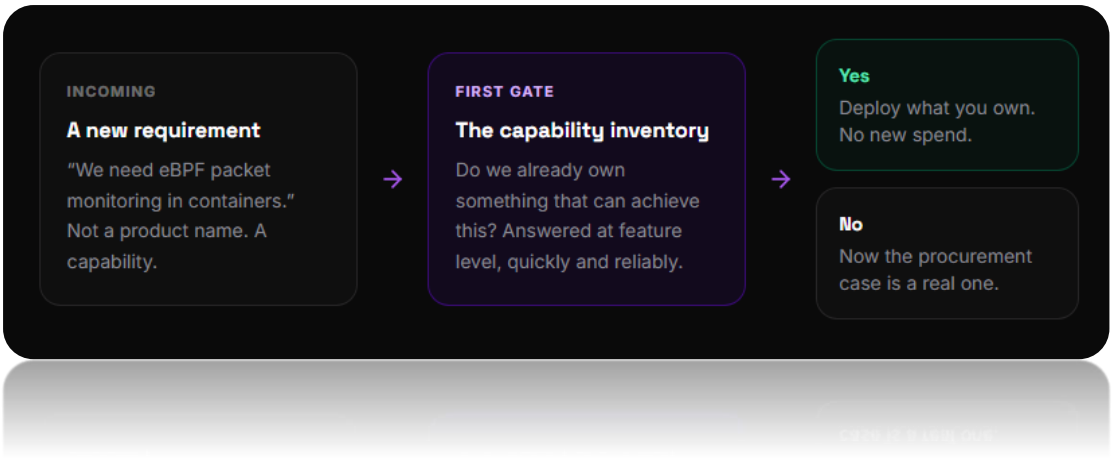
The purpose of the forum is to keep your stack and your budget pointed at the highest security value available. It gives you a defensible process with an audit trail running directly back to the people on the ground. The forum owns the capability inventory, and the inventory is your investment map.

STEP EIGHT

Make the inventory the front door for new requirements

New requirements will keep arriving. Regulation changes, the business environment shifts, the threat picture moves, and your stack has to evolve with all of it.

At the risk of sounding like a broken record, the first port of call for any new requirement should be the capability inventory. This is where "not just a list of products" earns its keep, because requirements surface at feature and capability level, not product level. Someone will ask for eBPF packet monitoring in containers, not for a product name.



The inventory as first gate. Either you deploy what you own, or the procurement case is a real one.

Embed the inventory into your new requirement process as the first step, so that "do we already have something that can achieve this?" gets a quick and reliable answer.

I have seen a lot of organisations, global ones especially, go out and purchase capability they already owned. One had a large-scale Splunk deployment being used for application performance monitoring, and the security team went and bought something else entirely, purely because they were unaware the deployment and the commercial relationship existed. Writing that down makes it sound ridiculous, and I wish I could say it was a conscious decision.

Make the inventory accessible, and given the direction of travel, make it programmatically accessible: API, MCP, something that can be wired into automated checks. Procurement teams can then sanity check a purchase themselves without needing intimate knowledge of every tool in the estate.

ESPROFILER

Where we come in

I have written this so you can pick it up and run with it regardless of how you choose to build it. But I would not be doing my job if I did not tell you how we help, because all of this consumes the one resource none of us get back.

The outreach. For a stack of 80 tools, running the ground-level conversations at a twice-yearly cadence works out somewhere in the region of 1,440 to 2,600 hours. That is 0.8 to 1.5 FTE, and it only buys you one person per tool. We automate it with asynchronous agent-based interviews, which lets us go wider and deeper across the orbit of each tool, with no meetings required and very little friction on the team. It also means you can run it more often than the renewal cycle, so you are tracking maturity, risk and opportunity continuously rather than once a year.

Maintaining the catalogue. Most organisations I see are still tracking things like Bluecoat ProxySG, which has been acquired twice since (Symantec, then Broadcom), renamed, and shipped countless new capabilities. We have had customers arrive having spent four months of architecture time building out what they have and aligning it to a framework, only for it to be out of date on arrival. We manage the catalogue of over 27,000 products, indexing new startups and tracking M&A, headcount and follower metrics so we stay ahead of the changes. We target a 30-day update cadence on every product our customers track, and we can align your stack to multiple frameworks at the same time: NIST, MITRE, custom architecture taxonomies. For that same customer, we demonstrated it on the call in four minutes.

Our platform and data are fully open and extensible, so it can serve as the entry point for capability checks before any requirement proceeds, and it is exposed through our co-pilot so you can simply ask.

If you are new in post and thinking it is ironic to be sold another platform to solve tool sprawl, that is not lost on us. It is why we offer the **Security Reality Baseline** as a fixed time, fixed cost engagement: a fast way to get on top of the stack when you have just taken the hot seat. It builds the picture of what you have, the reality on the ground, the risks, the opportunities and where consolidation is available. If you decide to move forward with us afterwards, we credit the engagement back against your platform subscription.

We also offer a free tool optimisation map to help you identify structural and capability overlap once you have the picture of what you own.

> Either way, built by us or built by you, the forum and the centralised capability view are the best model I have seen for keeping a security stack honest, keeping sprawl in check, and giving a leader a view of budget deployment that will stand up to board scrutiny.

WHAT NOW?

Ready to see your stack as it really is?

Tell us where you are with your stack. We will bring the evidence, do the heavy lifting, and leave you with decisions you can defend.

Tool Optimization Map Free. Structural and capability overlap, once you know what you own.

Security Reality Baseline Fixed time, fixed cost. What you own, what is really deployed, what you can cut, keep or renegotiate.

Security Consolidation Baseline Merger, acquisition or a consolidation mandate. Which moves are consolidation-safe, evidenced.