WHITE PAPER · CYBERSECURITY ARCHITECTURE

Spend Where the Attacks Actually Start

A cost-effective defense-in-depth architecture for organizations with limited IT resources

Attacks vary enormously in what they do once inside, and far less in how they get in. Nearly all of them arrive through a message or a login. This is one architecture built on that convergence — eleven control categories in six layers, arranged so that the technique defeating any one of them doesn’t carry to the next.

BY **JACK NASH** 6 LAYERS · 11 CONTROL CATEGORIES ABOUT 17 MIN READ

THE ARGUMENT IN BRIEF

- Nearly every intrusion that reaches a small organization arrives the same way: **a message and a login**. That convergence is what makes a small architecture viable.
- Eleven control categories, arranged in six layers, cover that path end to end — each placed where it can see something the others structurally cannot.
- Which standards you build against is itself a decision. **Three roles need filling** — organize, prioritize, configure — and no single body does all three well.
- The layers are not backups of each other. **Trace a single attack through the stack** and each layer gets a different, non-interchangeable chance to end it.
- Apparent duplication should survive one question: **remove this control, and what real situation goes uncovered?** Three overlaps here answer it. A third endpoint agent usually cannot.
- Several goals have both an expensive route and a nearly free one. **Immutable snapshots, wider SSO, and edge filtering on hardware already owned** deliver most of the protection for close to nothing.

PART I **The Design Problem**

Introduction

The organizations I work with now are not Fortune 500 companies. They are parishes, nonprofits, clinics, schools, and small businesses — a handful of servers, no security staff, and a board that approves the IT budget once a year.

They face the same threat landscape as the enterprises I spent my career inside. The phishing campaign that reaches a forty-person nonprofit is the same campaign that reaches a forty-thousand-person manufacturer. The ransomware operator does not check the balance sheet first. What differs is not the threat. It is everything available to answer it.

The usual conclusion is that such an organization simply has to accept more risk. I don't think that follows. What it has to accept is a *smaller* architecture — and a smaller architecture can be a genuinely strong one, provided every control in it sits where the attacks actually arrive rather than where the product catalogue happens to be deepest.

This paper sets out one such architecture. Eleven control categories in six layers, described in vendor-neutral terms, with the reasoning for each position and for the places where two controls appear to do the same job and do not.

Attacks converge before they diverge

Intrusions vary enormously in what they do once inside. They vary far less in how they get in.

Strip the specifics from the incidents that reach organizations of this size and nearly all of them pass through one of two doorways. Someone receives a message and acts on it — opening an attachment, following a link, paying an invoice that was never owed. Or someone's credential is used by a person it does not belong to, phished from that same message or reused from a breach somewhere else.

Everything that follows — the lateral movement, the privilege escalation, the exfiltration, the encryption — is the second act. The first act is almost always a message or a login.

> **Attack methods diverge wildly after entry. They converge sharply at it. A small architecture can afford to be thorough exactly where that convergence happens.**

This is the same reasoning I have applied to network spend: find the layer where the failures actually occur, and concentrate there rather than distributing budget evenly across a diagram. A large enterprise can afford defense everywhere. A small one has to aim — and the aiming point is not a mystery.

Assume one control will fail

Defense in depth means multiple independent layers, each addressing a different stage of the attack lifecycle, so a threat that slips past one control meets another. The premise is not that the stack is impenetrable. The premise is the opposite.

> **Defense in depth doesn’t make an organization unbreachable. It makes a single mistake survivable.**

That framing matters when explaining the architecture to a board. Nobody is buying certainty, and any vendor promising it should be shown out. What is being bought is the difference between one person clicking one link on one Tuesday and that click ending the organization’s week.

It also sets the design rule that governs everything below. If each layer is assumed to fail, then no layer may depend on another having succeeded, and no two layers may fail for the same reason. A second control defeated by the same evasion as the first is not a second layer at all.

PART II **Choosing the Authorities**

Pick the authority before the product

Every architecture rests on some source of truth about what good looks like. The only question is whether that source was chosen deliberately or absorbed by default.

Absorbed by default is the norm. A vendor’s datasheet says a control is essential. A peer organization runs a particular stack. A conference talk, a managed provider’s standard bundle, a phrase like “industry best practice” with no industry and no practice named behind it. None of these is worthless, and none of them is an authority. They are opinions with a commercial interest attached, and in the absence of a chosen standard they fill the vacuum by default.

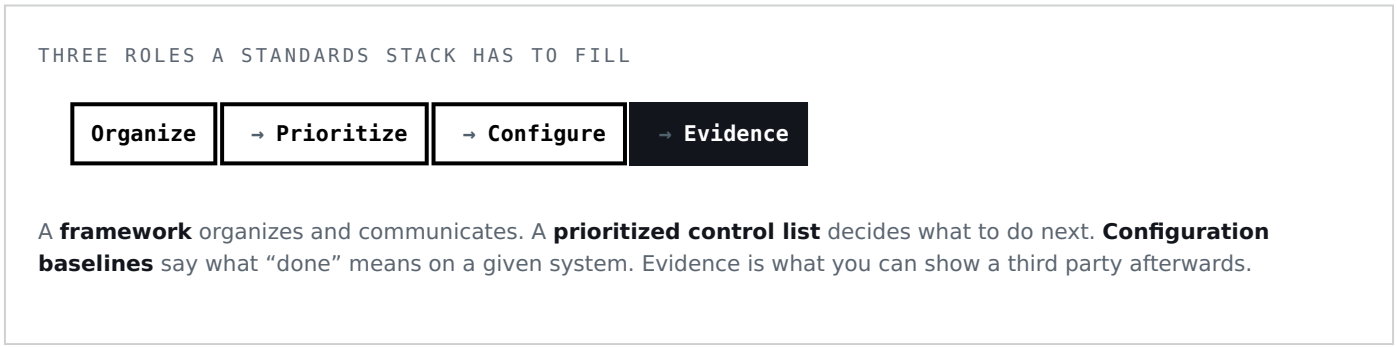
Choosing deliberately costs an afternoon and changes every conversation that follows. It converts “I think you should buy this” into “this is listed as baseline hygiene by a published standard and you don’t have it,” which is a different sentence with a different burden of proof. It also gives the organization something to hold the advisor to.

> **Whoever supplies your definition of adequate is setting your budget. It should not be a company that also sells the remedy.**

One standard cannot do three jobs

The most common error is not picking the wrong body. It is picking one body and expecting it to do everything, then concluding that standards are useless when it doesn’t.

Three distinct jobs need doing, and they are usually best served by different documents.



Organize. A framework gives structure and vocabulary: the categories of work, the shape of a complete program, and language that a board and an engineer can both use. Frameworks are deliberately non-prescriptive. That is a feature when you are explaining coverage to trustees and a frustration when you want to know what to do on Monday.

Prioritize. A control list turns the framework into an ordered set of actions. This is where the sequencing lives — the answer to *what first*, which is the question that actually stops small organizations, far more often than *what*.

Configure. Benchmarks say what a hardened system looks like setting by setting. Neither a framework nor a control list will tell you what a workstation should be configured to; a benchmark will, in detail, per platform.

Try to run a program on a framework alone and you get vocabulary with no roadmap. Run it on benchmarks alone and you harden individual machines beautifully while missing that there is no backup. The roles are complementary, not competing, and a body that is excellent at one is often deliberately silent on another.

Six questions to ask of a standards body

Given several candidates for each role, these are the tests I apply, roughly in order of how much they matter to an organization of this size.

- **Who funds it?** Consensus-developed, government, or nonprofit bodies have no product to sell you. A vendor’s “framework” may be technically sound and still be an instrument of positioning. Follow the funding before the content.
- **Is it maintained?** Look for dated revisions and a visible revision history. A standard last revised before the current threat landscape existed is a historical document. Both sources used here were revised in 2024.
- **Is it sized to the organization?** This is the test small organizations most often fail to apply, and it is the one that decides whether the program survives contact with reality. A catalogue written for federal systems will bury a five-person office. A standard that explicitly tiers itself by organizational capacity is doing you a favor.

- **Can you read it without buying it?** Freely published standards can be handed to a board member, a volunteer, or an incoming administrator. Paywalled ones cannot, and for organizations at this scale that difference decides whether anyone but the advisor ever reads it.
- **Does it map to the others?** Published crosswalks are what let one body of work answer several questionnaires — an insurer's, a grantmaker's, a client's. Mapping is the difference between one program and four.
- **Can you produce evidence against it?** Each requirement should be specific enough that you can point at something — a setting, a log, a signed policy, a test result — and say *here*.

> **A standard you cannot produce evidence against is not a standard. It is a preference with a citation.**

Separate what you adopted from what applies anyway

The tests above choose the standards you *adopt*. A second category exists that you do not choose at all, and conflating the two causes real trouble.

Obligations arrive from outside and apply whether or not anyone selected them: sector regulation covering health or payment data, contract and grant terms, state privacy statutes, cyber insurance application questions that become warranties once answered, and denominational, diocesan, or association policy. An organization does not decide whether these apply. It only decides whether it has found out.

So the sequence is: **discover the obligations first, then choose the voluntary standard that best covers them plus everything they miss.** Obligations are usually narrow — they address one data type or one system and are silent about the rest of the environment. A voluntary standard chosen well covers the whole organization and, through its crosswalks, most of the obligation as a by-product.

Worth stating plainly for the audience this paper is written for: an insurance questionnaire is not a security program, and answering it optimistically is a way of making a coverage problem into a claims problem. Treat it as an obligation to be evidenced, exactly like the rest.

What those tests select here

Applied to organizations with limited IT resources, the six questions land in the same place nearly every time.

NIST Cybersecurity Framework 2.0 fills the organizing role. Published by a government body with nothing to sell, revised in February 2024, and explicitly reframed in that revision to be usable by smaller organizations. It structures the program around six functions — Govern, Identify, Protect, Detect, Respond, Recover — and the Govern function added in 2.0 is precisely the part small organizations skip. It is free, and it is the vocabulary insurers and auditors already speak.

CIS Controls v8.1, Implementation Group 1 fills the prioritizing role, and does so better than anything else available to this audience. The full set runs to 153 safeguards; IG1 is the 56 of them defined as essential cyber hygiene for organizations with limited expertise and resources. That tiering is the sized-to-you test being answered explicitly by the authors rather than guessed at by the reader. Published by a nonprofit, revised June 2024, freely available, and mapped to CSF 2.0 and to most regulatory regimes.

CIS Benchmarks fill the configuration role, with consensus-developed hardening standards per operating system and application, at a level of specificity that makes evidence trivial: the setting either matches or it does not.

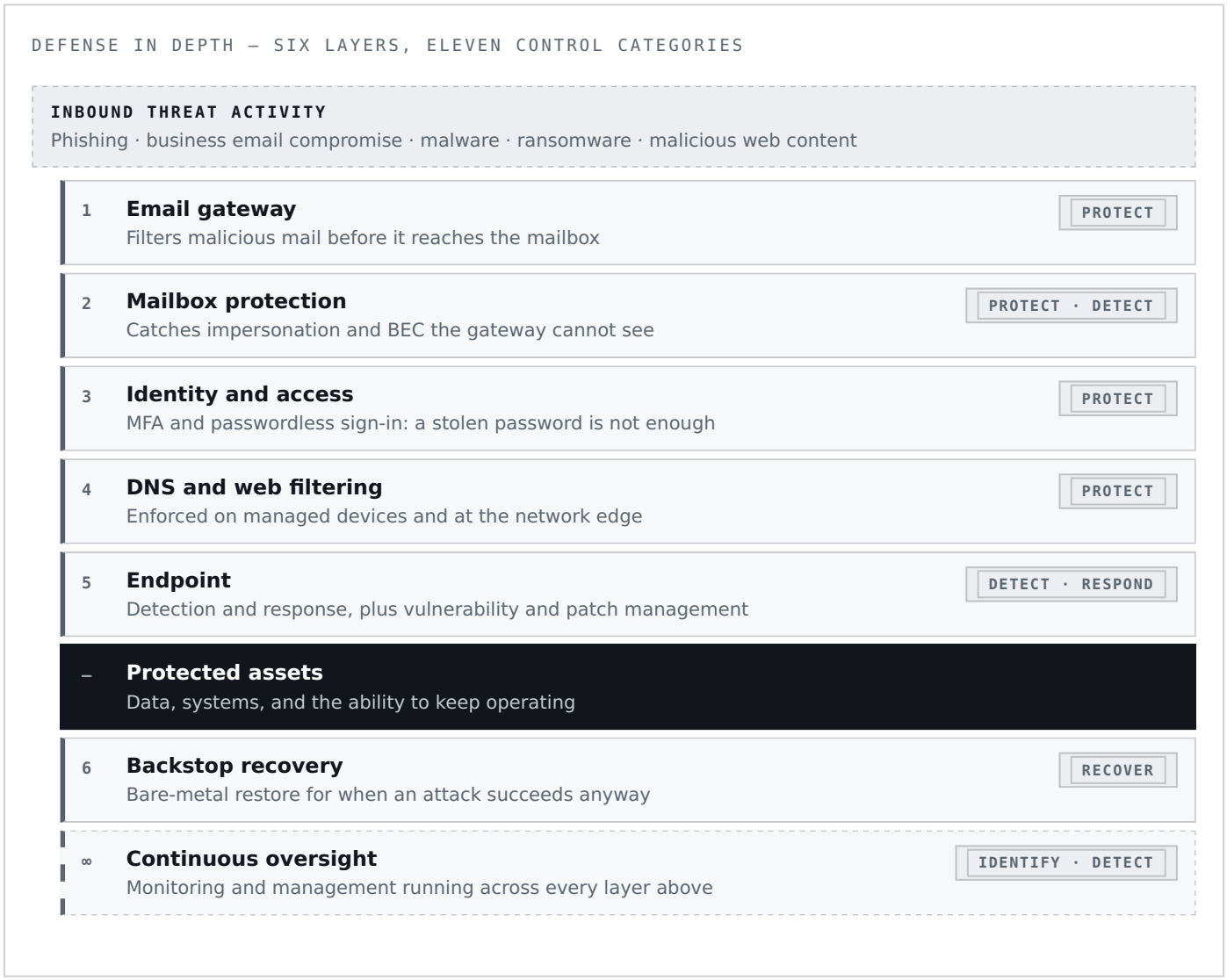
Other bodies are better answers to different questions, and it is worth knowing which. **ISO/IEC 27001** is the right choice when a customer contract or a certification requirement demands third-party attestation — it is certification-grade, and it carries purchase and audit costs that only make sense when someone is asking for the certificate. **NIST SP 800-171** is not chosen at all; it is flowed down by federal contracts. The Australian Signals Directorate's **Essential Eight** and the UK's **Cyber Essentials** are both excellent, similarly sized alternatives to IG1, and in their jurisdictions they carry recognition that CIS does not.

The point is not that this particular trio is uniquely correct. It is that the trio was chosen against stated tests, in three named roles, and can be defended or replaced on those grounds by whoever comes next.

PART III **The Architecture**

Six layers, eleven controls

The diagram below is the whole architecture. Each row is a layer, each layer addresses a different stage of the attack lifecycle, and the tag names the primary NIST CSF 2.0 function it serves.



Layers 1 and 2 — email, from two vantage points

Email is where the majority of these attacks begin, so it gets two layers rather than one. They are not the same control bought twice.

A **secure email gateway** evaluates messages on the way in — reputation, content, attachment and link analysis — and stops the bulk of malicious mail before delivery. **Post-delivery mailbox protection** evaluates messages already delivered, with visibility into internal sending patterns and established correspondent relationships.

The difference is not depth. It is vantage point, and it is the clearest illustration in this paper of why layering works at all.

Business email compromise typically carries no malicious payload: no attachment, no link, no malware, nothing a gateway is built to detect. It is a plausible message from a plausible sender asking for a wire transfer or a change of bank details. Catching it requires knowing that this sender does not normally make that request, in that tone, on a Friday afternoon. A gateway

inspecting a message in isolation cannot know that. It is not a weaker filter — it is standing in the wrong place to see the thing that makes the message dangerous.

Layer 3 — identity, the layer that pays for itself

Most successful attacks are not intrusions in any technical sense. Once a password is phished, reused, or guessed, the attacker inherits everything that account can reach and walks in through the front door holding a valid credential. Nothing is broken, so nothing alarms.

An **identity provider with multi-factor and passwordless authentication** severs the chain between a successful phishing email and a successful account takeover. The credential still gets stolen. It just stops being sufficient.

Of every control in this architecture, this one returns the most per dollar, and it is the one most commonly half-deployed — enabled for administrators, or for email, but not extended across the environment. Partial coverage protects the accounts an attacker was least likely to reach and leaves open the ones they were.

Layer 4 — filtering the device you don’t own

Not every device on the network belongs to the organization. Staff phones, volunteer laptops, vendor equipment, and guest devices connect constantly, and none of them can have security software installed on them.

DNS filtering is therefore enforced at two different points, and this is the cleanest example in the architecture of two controls that look redundant and are not.

Agent on managed devices
+ Runs on organization-owned computers
+ Travels with the device — home, hotel, public Wi-Fi
+ Enforced whether or not the device is on site
– *Cannot be installed on a visitor’s phone*

Filtering at the network edge
+ Runs on the firewall or router
+ Covers every device the moment it joins
+ Requires nothing installed on the device
– *Cannot leave the building*

An organization-owned laptop at an employee’s house is covered by the first and invisible to the second. A visitor’s personal phone on the guest network is covered by the second and untouchable by the first. Both are ordinary weekly occurrences. Remove either control and one of them goes unfiltered.

This layer also does quiet work at the far end of an attack. Malware that reaches a device still has to resolve a name to reach whatever is controlling it. Filtering at the resolver breaks that step without needing to recognize the malware itself, which is why it keeps earning its place even behind good endpoint protection.

Where edge filtering stops being an IT decision

For organizations that serve minors or the public — parishes, schools, youth programs, libraries, clinics — edge filtering carries a significance it does not carry in a commercial office.

Children and young people connect to guest Wi-Fi on personal devices the organization neither owns nor manages and cannot configure. Endpoint filtering software is structurally incapable of reaching those devices.

> **Without filtering at the edge, an organization offering Wi-Fi is providing unrestricted internet access to minors on its own premises.**

That is a safeguarding fact before it is a technical one, and it sits alongside the organization’s other safe-environment practices rather than in the category of optional IT expenditure. It is also precisely the kind of reasonable, documented step that safeguarding policies and insurers expect to find in place. Confirm the requirement that applies, and record that the control satisfies it.

Layer 5 — shrink the surface, contain the rest

Endpoints are where users open attachments, enter credentials, and run software, which makes them the most likely place for an attack to gain its first foothold. Three control categories share the layer and do three different jobs.

Endpoint detection and response detects and contains malicious behavior on the device, including activity that signature-based antivirus does not recognize — which matters because capable attacks increasingly use tools already present on the machine rather than malware that can be matched against a list.

Vulnerability and patch management identifies and closes known exploitable weaknesses, critically in third-party applications. That is where much of the real exposure lives, and where automatic updates usually do not reach.

Agent-based DNS filtering, described above, removes an entire category of destination from reach.

The first limits the blast radius. The second shrinks the attack surface before anything arrives. Neither substitutes for the other: patching prevents exploitation of a known flaw, and detection and response is what remains when the flaw is not yet known.

Oversight — you cannot protect what you cannot see

Without continuous visibility, outages and security incidents get discovered by users rather than by the people responsible for fixing them. Three elements cover this, and unlike the numbered layers it runs alongside all of them rather than sitting at one point in the lifecycle.

Vendor-neutral network monitoring provides inventory, alerting, and remote access across every device regardless of manufacturer — servers, workstations, printers, cameras, appliances, network hardware. **An infrastructure-specific management console** provides deep configuration, firmware, and client-level detail for the network equipment itself. **Dedicated management access** provides a support path that does not require borrowing a staff member's workstation.

The first two overlap in capability and differ in scope. One sees the entire network but treats each manufacturer's equipment generically; the other sees a single vendor's infrastructure in full detail. Running only the first loses the depth needed to diagnose the network equipment. Running only the second loses sight of everything else — which, in practice, is where the forgotten camera and the internet-exposed printer live.

One configuration is worth naming, because it is a common expedient that quietly couples two things that should stay apart: **management tooling hosted as a virtual machine on the backup appliance.** It works. It also puts the support tooling on the same device that holds the backups, so a failure of that appliance removes both the backup copy and the remote access needed to respond to it. A small dedicated management host separates them for the price of a mini PC.

Layer 6 — the layer that assumes the others failed

Every preventive control can be defeated by a sufficiently determined or sufficiently lucky attacker. One question decides whether that becomes a disruption or a disaster: can the organization restore without paying?

Image-based backup with bare-metal restore allows a complete system to be rebuilt rather than recovered file by file. That distinction matters more than it sounds. After a ransomware event, file-level recovery means reinstalling operating systems, reconfiguring applications, and reconstructing settings across every affected machine while the organization is not operating. Image-level recovery means restoring the machine and moving to the next one.

Two properties decide whether this layer actually functions under attack, and both are configuration rather than purchase. Backups must be **immutable** for a defined retention period, so that an account with full administrative rights cannot delete them — destroying the backups

first is standard practice, not an advanced technique. And backup credentials must be **separate** from everyday network and domain logins, so a compromised workstation cannot reach the repository at all.

> **Backups are the control that turns a ransom demand into a scheduling problem.**

PART IV **How the Layers Work Together**

One message, five chances

The layers are easiest to justify individually and easiest to understand collectively. So take a single ordinary attack — a phishing message carrying a credential-harvesting link, the most common intrusion these organizations face — and follow it through the architecture.

A CREDENTIAL-PHISHING ATTEMPT, LAYER BY LAYER

Every stage after the first can end the attack on its own — and each does so for a **different reason**, which is what makes them layers rather than duplicates.

The gateway may recognize the sending infrastructure, the link, or the payload, and never deliver it. Most volume dies here, and that is the layer’s job: reduce what everything above it has to consider.

Mailbox protection may catch what the gateway passed, because it can see that this sender has never before written in these terms, or that the display name matches an executive while the address does not.

DNS filtering may refuse to resolve the destination when the user clicks anyway — and users do click. This stage is indifferent to how convincing the message was, which is exactly why it works: it never evaluates the message at all.

Identity is where the attack usually ends even when everything above it has failed. The user reaches a convincing fake login page and enters a correct password. Without a second factor, the account is gone. With one, the attacker is holding a credential that no longer opens anything.

The endpoint catches what follows if the attacker converts access into execution — the tooling, the persistence, the lateral movement — and patch management reduces what that stage has to work with.

Recovery assumes all of it failed and asks only whether the organization can rebuild without paying.

Notice what makes the chain robust. The gateway can be fooled by a message with no payload. Mailbox protection can be fooled by a first-contact sender. DNS filtering can be fooled by a domain registered an hour ago. Identity can be fooled by a user who approves a prompt they did not initiate. Every one of those evasions is specific to its own layer, and none of them carries to the next.

THE DESIGN RULE

Two controls are two layers only if the technique that defeats one does not defeat the other.

Everything else is the same layer, purchased twice.

Apply the removal test

Any architecture containing controls that sound similar will eventually face the same question, usually at budget time.

“Why are we paying for both?”

The budget-season question every layered stack eventually gets

It is a fair question and it deserves a real answer rather than a brochure. The test is deliberately one a non-technical board member can apply without help.

> **Remove the control. Name the specific situation that now goes uncovered. If nothing can be named, it isn’t a layer — it’s a line item.**

Real means real: not a threat category, but a situation that occurs in this organization, in a normal week, that the remaining controls do not address. The three apparent overlaps in this architecture each answer it.

Two email controls. Remove mailbox protection and the uncovered situation is a payment-fraud message from a compromised vendor account, carrying no attachment, no link, and no malware, arriving from a legitimate sender the organization has corresponded with for years.

Two DNS enforcement points. Remove edge filtering and the uncovered situation is a visitor on a personal phone on the guest network. Remove the agent and it is an employee laptop on home broadband.

Two monitoring platforms. Remove the vendor-neutral one and the uncovered situation is a failing NAS, an offline camera, or a printer nobody knew was internet-exposed. Remove the infrastructure console and it is diagnosing a switching or firmware fault blind.

Each is specific, weekly, and real. Compare that to the honest result of running most security stacks through the same test, where the answer for the third endpoint agent is often a shrug and the word “layered.” Overlap in capability is fine and frequently unavoidable. Overlap in *coverage*, with nothing nameable on the other side of removal, is something else.

Several of these goals have a cheaper route

This architecture is deliberately modest in cost, and a meaningful share of its protection comes from configuration rather than purchase. Four examples, each delivering most of an expensive control’s benefit for close to nothing.

- **Immutable snapshots rather than replicated offsite copies.** The goal is a copy an attacker holding administrative credentials cannot encrypt or delete. Offsite replication achieves that and costs a subscription. Locked snapshots on the appliance already in place achieve most of it, defeat the same ransomware tactic, and typically cost nothing. Physical separation still matters for fire and theft — rotated media to a separate building covers that end without a recurring bill.
- **Wider single sign-on rather than more password management.** Connecting additional applications to the identity provider already deployed reduces the number of standalone passwords needing management at all, and extends multi-factor protection further into the environment. This is the rare option that makes the problem smaller instead of managing it better.
- **Self-hosted credential storage.** Open-source vault software runs on hardware most organizations already have. The vault stays on the premises with no subscription and no third-party custody — and the goal was never the vault, it was that no password exists only in one person’s memory or browser.
- **Edge filtering on the firewall already owned.** The safeguarding control described earlier is, in most environments, a configuration change on existing equipment rather than a new product.

The pattern generalizes. State the security goal without naming a product or a delivery model, and there is frequently a route to most of it running through equipment already in the rack.

What no product in the stack provides

The eleven categories above are the technical architecture, and they are not the whole architecture. Five safeguards complete alignment with CIS Controls v8.1 (IG1) and the NIST CSF 2.0 Govern and Recover functions, and they are the ones most often missing.

- **Security awareness training** — regular phishing simulation and user training, addressing the layer no product can patch.
- **Written incident response plan** — a documented plan naming who is contacted, in what order, and what the first hour looks like.
- **Documented restore testing** — periodic verification that backups actually restore, on a defined schedule, with results recorded.
- **Asset inventory and access review** — a current record of devices and accounts, with periodic review of who has access to what.
- **Secure configuration baselines** — systems hardened against a published benchmark rather than left at installation defaults.

Look at what they have in common.

> **None of the five is a product. All five are process — which is precisely why they get deferred, and why no vendor is calling to sell them.**

They cannot be purchased and installed in an afternoon. They require someone’s attention on a recurring basis. They produce no dashboard. And collectively they are worth more than the next tier of product spend in most of the environments I see.

Two of them are load-bearing for the architecture rather than merely advisable. Restore testing is what converts Layer 6 from an assumption into a capability; an untested backup is a belief. And awareness training acts on the layer every one of these attacks is aimed at, which is the person reading the message.

What the architecture adds up to

Eleven control categories. Six layers plus continuous oversight. Nothing exotic, nothing bought to duplicate anything else, and several positions filled by configuring equipment the organization already owns.

What makes it hold is not the count. It is that each layer sits where it can see something the others structurally cannot, and that the technique defeating any one of them does not carry to the

next. That property is what a stack of overlapping products bought on separate occasions almost never has — and it is available to an organization with no security staff and a modest budget, because it comes from arrangement rather than from spend.

Aim at the convergence point. Layer so that the failures do not correlate. Then assume the whole thing fails anyway, and keep a tested way back.

Put the layers where the attacks arrive.

Make each one see what the others cannot.

A note on scope. This paper describes control categories rather than specific products; any competent implementation of a category satisfies the architectural requirement, and the choice among them is a matter of budget, existing platform, and operational fit. Frameworks referenced are the NIST Cybersecurity Framework 2.0, CIS Controls v8.1 (Implementation Group 1), and the CIS Benchmarks.