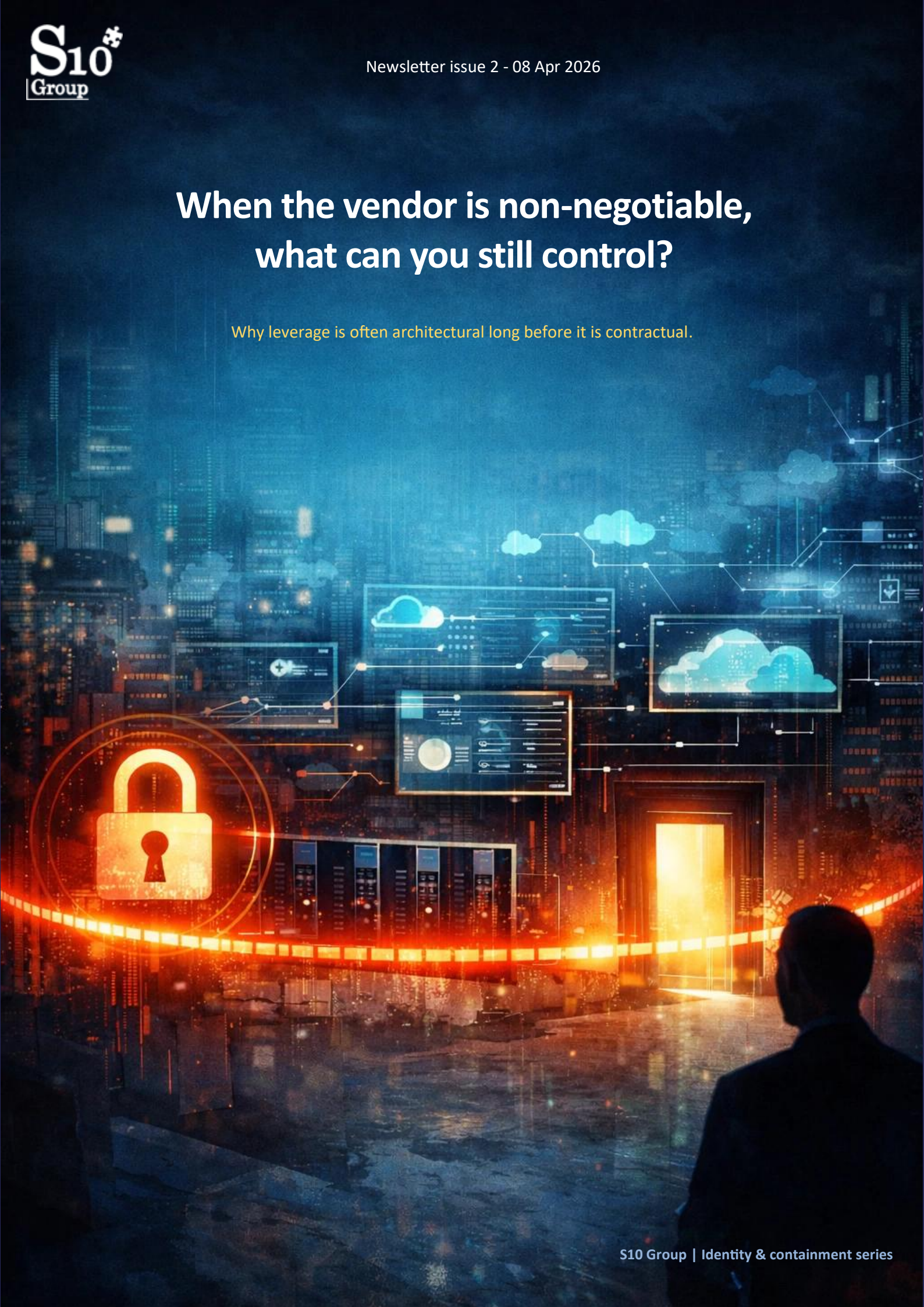
Newsletter issue 2 - 08 Apr 2026

When the vendor is non-negotiable, what can you still control?

Why leverage is often architectural long before it is contractual.



The dependency you cannot quickly exit

A live example of vendor dependency is unfolding in Dutch healthcare right now. This is why I wanted to write this newsletter issue. I keep seeing how quickly a supplier incident can stop being “about the vendor” and become a continuity question for everyone connected to it.

ChipSoft was hit by ransomware on April 7, 2026. Roughly 70% of Dutch hospitals use this supplier. After the incident, the issue did not remain confined to the supplier. Hospitals started making protective decisions almost immediately: OLVG temporarily stopped exchanging medical data with hospitals using ChipSoft, Rijnstate closed its VPN connection to the affected environment on advice from Z-CERT, and NOS reported that eleven hospitals took patient portals offline.

And that, to me, is the real signal: when a critical dependency becomes unsafe, the issue is no longer only what happened at the supplier. It is what customer organisations can still restrict, detach, or degrade around that dependency without losing control of operations.

That is exactly why this question of dependency matters.
There is an uncomfortable truth in many third-party risk discussions: once a vendor becomes unsafe — through compromise, credential abuse, hidden dependencies, or operational failure — the issue is no longer only whether the supplier is at fault, or even whether the service matters. It is what the organisation can still restrict, detach, degrade, or otherwise control around that dependency once it has itself become part of the risk. (Link to NOS) (Link to OLVG)

Some vendors are only suppliers on paper. In practice, they are part of how the organisation runs. And when one of those dependencies becomes unsafe, the real question is no longer whether the contract was well drafted. It is whether the organisation still has any real room to move. (Link to Bitsight)

A procurement decision that aged badly

The moment usually does not begin in the incident room. It begins much earlier.

A service is renewed because it is deeply embedded, the migration cost is high, the business depends on it, and the vendor is considered “strategic”. Extra controls may be written into the contract: notification clauses, remediation timelines, audit rights, service expectations. All of that is sensible. BitSight’s vendor-contract guidance explicitly recommends specific breach-notification windows and remediation timelines rather than vague language, and ongoing monitoring across the relationship life cycle. (Link to Bitsight)

But when the pressure comes, a hard truth appears:
contractual rights are not the same as operational leverage.

If a critical platform sits inside identity, payments, customer operations, claims handling, care delivery, or another core process, you often cannot simply “turn

- One Lesson

Leverage is architecture.
When you cannot quickly exit the dependency, control comes from boundaries, access design, and kill-path options.

it off" without creating a second problem of your own. That is why DORA-style thinking has pushed exit strategy, resilience testing, and dependency visibility into the centre of operational resilience rather than leaving them as procurement afterthoughts. (Link to European Payments Council)

Why this matters now

The operating environment is becoming harder to govern, not easier.

Recent reporting around ChipSoft shows how quickly a supplier incident can become a care continuity issue for customers. More broadly, vendor risk guidance continues to stress two truths that leaders underestimate: contracts need explicit notification and remediation expectations, and third-party risk has to be monitored across the entire relationship lifecycle. But when trust breaks, monitoring alone is not enough. What matters then is whether the organisation has already designed practical room to narrow access, pause integrations, and contain exposure before operations start to unravel. (Link to Help Net Security) (Link to NOS)

That changes the leadership question.

The issue is no longer only whether the vendor is compliant, or whether the security review was completed at onboarding.

It is whether the dependency can be contained when the dependency itself becomes the risk.

The decision boundary that matters

At that point, the conversation needs to move from vendor management to governability.

If the vendor is critical and non-negotiable in the short term, what is the actual lever?

Not a penalty clause.
Not an angry escalation call.
Not a contract line saying the provider "must cooperate".

The real lever is usually architectural:

- what the vendor can reach
- what identities and tokens flow through the dependency
- what data paths exist
- what can be degraded or isolated without collapsing operations
- what fallback mode exists if trust in the vendor becomes uncertain
- what exit trigger has been defined before the crisis rather than during it (Link to enisa.europa.eu)

That is the core shift in thinking.

- Board Question

If a critical vendor became unsafe tomorrow, what could we still restrict, detach, or degrade fast enough to keep control of the business?

In moments like this, the instinct is often to ask **who** got it wrong: which supplier failed, which team approved the dependency, which contract did not protect us strongly enough.

The more useful question is **why** the organisation was left with so little room to move once trust in that dependency started to fail.

That is usually where the real lesson sits — not in blame, but in the design choices, dependency paths, and control boundaries that made the problem harder to govern. (Link to Atlassian) (Link to cognidox.com)

What incidents like ChipSoft actually expose

Incidents like the ChipSoft case are often described in operational terms: a supplier was hit, portals were paused, data exchange was restricted, connections were closed. But the harder truth is usually not only that something failed. It is that customer organisations suddenly have very little room to govern the dependency once trust starts to break.

It is not that someone missed a checkbox or that the contract had no value.
It is that the organisation had not designed enough room to contain, detach, restrict, or degrade the dependency once trust started to fail.

A line worth remembering is this:

Contracts buy obligations. Architecture buys options.

This is also why exit strategy deserves more executive attention than it usually gets. Recent DORA-focused guidance treats it as part of resilience, not merely procurement hygiene: define the triggers, the fallback arrangements, the data protections, and the transition steps before pressure arrives. Because when a critical dependency becomes unsafe, the real issue is not whether the contract allows exit. It is whether the organisation has any credible way to reduce reliance without losing control of operations. (Link to panorays)

In that sense, “exit” is often too narrow a word; what matters first is controlled detachability.

The hidden problem is not only the vendor

There is also a second layer that makes this harder: fourth-party opacity.

Your organisation may know the named vendor. It often knows much less about the vendor’s own dependency chain: subcontractors, cloud services, identity providers, software components, MSP relationships, support channels, and hidden trust paths. Public cyber guidance continues to warn that incidents at one supplier can cascade into many customer organisations precisely because those dependencies are not visible enough — or not governable enough — when they suddenly matter most. (Link to Canadian Centre for Cyber Security)

That is why some vendor failures spread so far.

- Pressure Point

The most dangerous dependency is often not the one with the biggest contract. It is the one you cannot quickly replace and cannot clearly contain.

The problem is not only that a supplier was compromised.
The problem is that the supplier sat on a trust path no one could narrow quickly enough.

What leaders should ask next time

Three board questions are more useful here than a longer checklist.

1. **Which vendors are critical in practice, not only in category?**
   Not who has a large contract. Who sits inside a process we cannot easily replace, isolate, or degrade.
2. **If trust in that vendor broke tomorrow, what could we safely restrict first?**
   Access, tokens, data flows, integrations, privileges, shared services, customer-facing features.
   In healthcare, that may also mean patient portals, external data exchange, VPN paths, and connected workflows that are still operational but no longer fully trustworthy.
3. **What is our real exit or fallback position if the dependency becomes unsafe?**
   Not the theoretical one. The one the business could actually live with under pressure.

These questions matter because they shift the conversation from reassurance to options. And in a live incident, options matter more than policy language.

One pressure-test worth running

A useful capability check here is not a generic vendor review.

Take one genuinely critical vendor and run a 45-minute containment-boundary validation around a simple scenario:

Assume the vendor is still technically available, but no longer fully trustworthy.

Then test, in sequence:

- what identities or sessions would need to be invalidated
- what integrations could be narrowed or paused first
- what data flows could be contained without freezing the whole business
- what minimum operational mode is acceptable
- who can authorise those moves
- what conditions would trigger a partial detach, a full detach, or a controlled fallback

The value is not proving that the vendor contract exists.

The value is discovering whether the organisation has any practical lever when the service cannot simply be terminated.

– What Contracts Change

Contracts can buy reporting duties, remediation timelines, and audit rights.
They do not, on their own, create room to act when the dependency is already inside your critical path.

The real lesson

This is why I do not think critical-vendor resilience is mainly a procurement issue. It is an architectural one.

Contracts still matter. Audit rights matter. Incident reporting clauses matter. Remediation obligations matter. But in the moment when a critical dependency becomes unsafe, those things do not by themselves create room to act. Containment boundaries, access design, token discipline, dependency mapping, and kill-path options do. (Link to Bitsight)
And that is what makes the issue so uncomfortable.

The organisation may discover that the dependency was reviewed, approved, renewed, and monitored — and still find that, when pressure hits, it has very little immediate leverage.

So, the question I would leave with is not whether your critical vendors meet policy. It is this:
If one of your non-negotiable vendors became unsafe tomorrow, what could you still control by design?

That is exactly where S10 Group's platform fits.

Not by replacing the vendor. Not by rewriting the contract after the fact. And not by adding another dashboard.

But by helping organisations create operational control when trust in a critical dependency becomes uncertain: reduce exposure, interrupt malicious behaviour, contain spread, and preserve enough room to keep core operations governable while the dependency is being assessed, restricted, or re-routed.

In other words, the value is not only in knowing that a critical vendor has become a risk. It is in having a practical way to respond without losing control of the environment around it.

Because when the dependency is non-negotiable, leverage does not come first from legal language. It comes from what the organisation can still isolate, narrow, and protect by design.

For readers who want to go further:
I can share more detail on how **S10 Group's platform** helps create control around a critical dependency when trust in a vendor can no longer be taken for granted.

And for teams that want to pressure-test their own setup:
I can run a **free remote resilience assessment** in your own sandbox environment, with your existing security controls enabled, to show how your environment behaves under pressure and **what difference that added control makes in practice**.
If any of these would be relevant for your team, feel free to contact me.
sgemert@s10group.com

- What Containment Changes

Containment gives leadership a way to narrow exposure around a vendor that cannot simply be switched off.
It turns dependency risk into something that can still be governed.