2026 ARCHITECTURE GUIDE

Island

The Perfect Packet: A Guide to Modern SASE Architecture

How to evaluate your current stack, close the gaps traditional proxies can't see, and build enforcement that scales

20K+

Enterprises purchased SASE solutions in 2025. Most have only partially deployed the capabilities they own.

62%

Leaders view SASE as critical to their overall security strategy and want to make sure the investment pays off.

100X

The scale at which an AI agent workforce will operate using employees' roles and access.

SASE promised to solve the security problem of the modern enterprise.

For a time, it did. And then AI entered the picture and changed everything.

Over 20,000 enterprises purchased SASE solutions in 2025, yet most have only partially deployed the capabilities they own. Licenses sit unused. Bypass lists quietly grow. Policies look complete on dashboards while gaps widen in practice.

This isn't a failure of execution. It's a sign that the problem has changed.

Work has moved. Risk has moved. And AI has accelerated both faster than any architecture could anticipate.

Employees paste sensitive data into prompts. Agents call external tools, access internal documentation, and move output downstream at machine speed. An organization's agent workforce will soon operate using employees' roles and access at 100x the scale.

While all of these events will eventually cross a network, it'll happen too late, after the moment of intent has passed.

The enforcement model most SASE vendors are built on was designed for a world where work flowed through corporate networks and applications lived in data centers. That was the right answer for that era. But AI has exposed a gap between where enforcement lives and where work actually happens now.

This guide is for practitioners who are responsible for making SASE work in the real world. It will help you understand where the architecture breaks down, why those breakdowns are structural rather than fixable, and what a more effective enforcement model looks like so you can evaluate your current stack honestly and make better decisions moving forward.

TABLE OF CONTENTS

What's inside

1

Where work *actually happens now*

There is a simple question at the center of every enterprise security decision: **where does work actually happen?**

A decade ago, the answer was clear. Employees worked on managed devices, inside corporate networks, accessing applications in data centers. Data moved across wires. The perimeter was a real thing you could draw on a whiteboard.

That world is gone.

Today, work happens in a browser tab, a SaaS platform, an AI tool, a desktop application on a personal laptop, a partner's unmanaged device connecting from a hotel in another country. Increasingly, it happens without a human in the loop at all as agentic AI workflows automate tasks, call external tools, and move data across systems at machine speed.

What that means for risk

When work moved into applications, risk moved with it.

- A freelancer copies customer data from a CRM and pastes it into a ChatGPT prompt.
- An analyst downloads a financial model and uploads it to personal cloud storage.
- A developer pushes proprietary code through an AI assistant to accelerate a sprint.

These are ordinary moments in the modern workday, and also the primary vectors through which sensitive data leaves the enterprise. None of these events look like a network incident. They don't generate anomalous traffic patterns or trigger firewall alerts. They occur at the presentation layer, inside the application, at the moment of user interaction, as work is being created and used.

The exposure isn't in transit. It's in the action.

Consider the scale of the gap. Modern encryption makes interception technically impossible for a growing share of traffic. And bypass lists required to keep critical applications functioning have quietly grown to cover most of the rest. Organizations end up with enforcement that looks complete on dashboards while gaps widen in practice.

That is not a configuration problem. It is an architectural one.

Why traditional SASE is falling behind. An architecture *built for a different era*

Traditional SASE standardizes on a single execution pattern: traffic is backhauled to a cloud proxy, broken and inspected, and forwarded to its destination. Enforcement depends entirely on routing sessions through centralized inspection points in distant points of presence (PoPs).

When that architecture was designed, it made sense. The problem isn't that it was wrong. It's that it was designed for a world that no longer exists.

The majority of web traffic runs over TLS 1.3, and HTTP/2, HTTP/3, and QUIC are now standard in every major browser. Certificate pinning prevents decryption for a growing list of applications. Post-quantum cryptographic implementations are already deployed in major browsers. These protocols were engineered to be fast and are structurally incompatible with legacy break-and-inspect architectures.

Every application added to a bypass list to preserve compatibility is an application outside the security perimeter. An architecture that requires exemptions to function is not enforcing policy, it’s avoiding it.

The detour tax

When the network becomes the primary enforcement point, every session pays a "detour tax." Traffic is forced through distant inspection paths even when the application is perfectly reachable directly.

1 Users feel it immediately.

CRM sessions lag. Video calls stutter. Ticketing systems stall. In bandwidth-constrained geographies, SaaS workflows become inconsistent or borderline unusable. Users find workarounds just to stay productive, and those workarounds quietly increase the attack surface.

2 Reliability becomes a bottleneck.

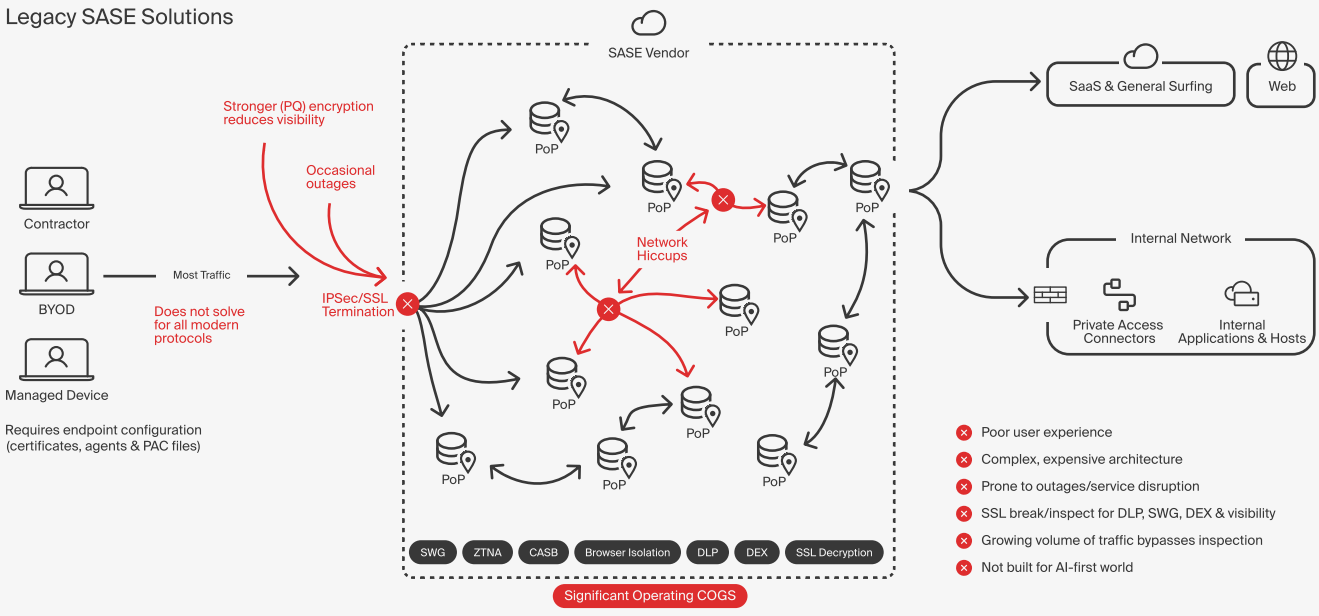
When a PoP degrades, a certificate chain breaks, or a decryption policy change propagates incorrectly, the blast radius is broad. A user in Toronto connected through a Detroit PoP may find their location misidentified, their compliance posture incorrectly evaluated, and their communications flagged for a border-crossing that never happened.

3 Operations grow harder, not easier.

Multiple service chains and overlapping policy engines create troubleshooting complexity that is difficult to contain. Rollouts stretch from weeks to months. Exception lists grow.

4 The shelfware problem is structural.

Licenses are purchased at scale. Deployments stall. Features are enabled but never tuned. Enforcement looks complete on dashboards while gaps remain in practice. This is a predictable consequence of architectures that require mandatory backhauling and extensive configuration before delivering meaningful coverage.



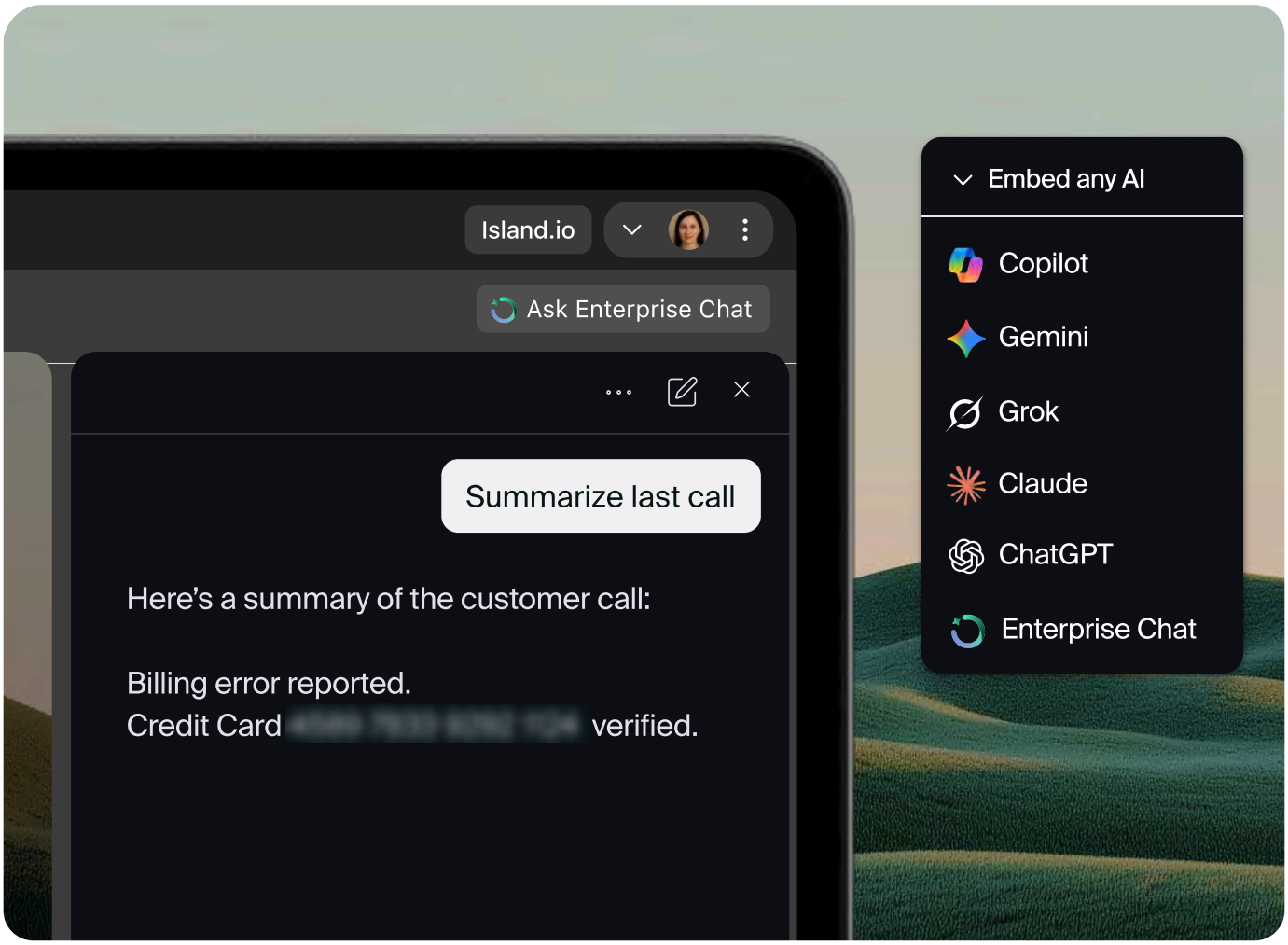
The visibility gap that can't be patched

Network inspection can surface destinations, connection metadata, and decrypted payloads when decryption is possible. What it cannot do is capture what a user actually did inside an application.

It can tell you a file was uploaded. It cannot tell you what was copied into an AI prompt, pasted into a web form, or moved between SaaS tenants before submission.

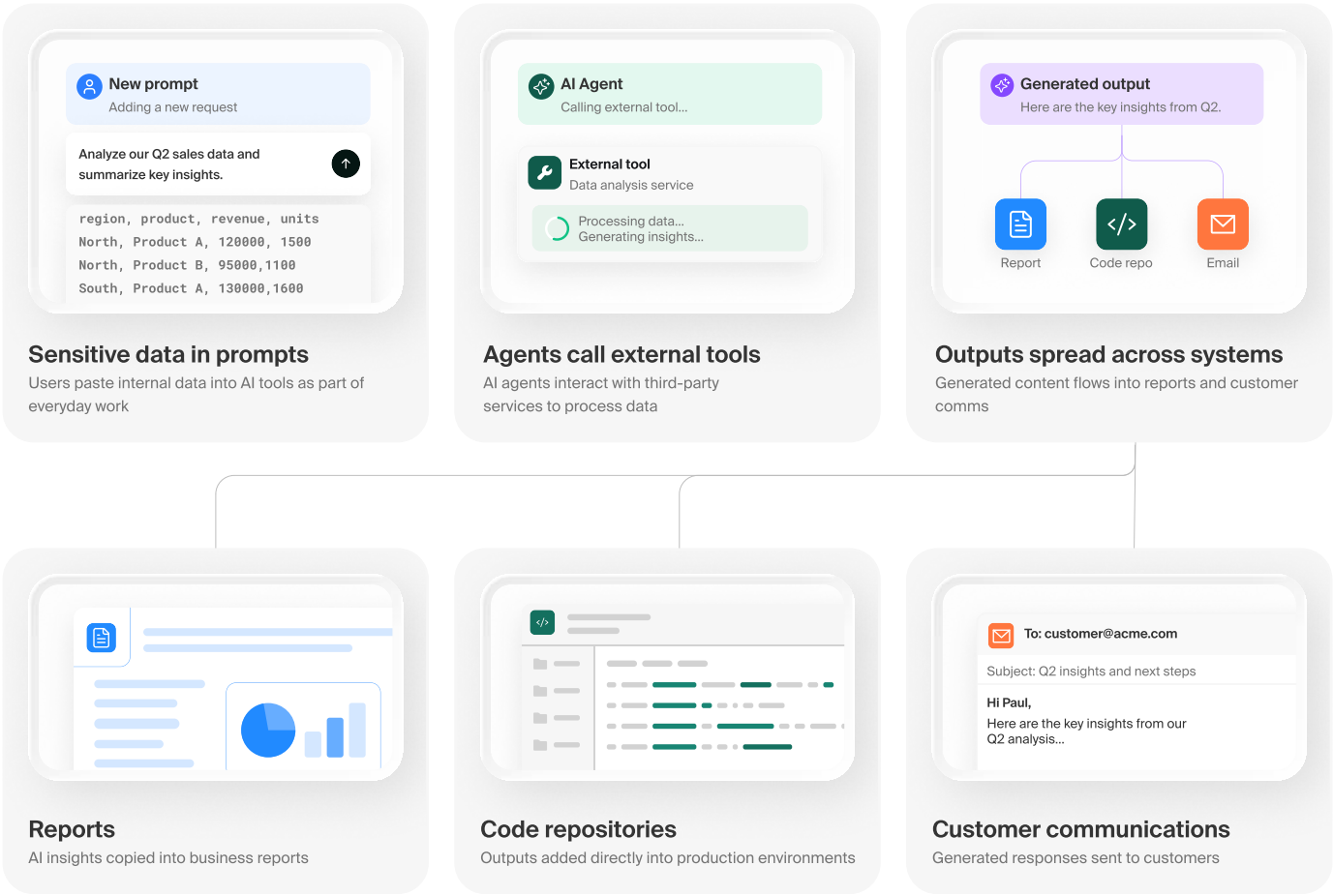
Persistent connections like WebSockets, which are common in modern SaaS and AI tools, carry continuous data streams that proxy inspection cannot reliably interpret, creating blind spots in exactly the workflows where visibility matters most.

As encryption strengthens, even payload inspection becomes harder to sustain. TLS 1.3 reduces available metadata. Certificate pinning makes decryption impossible for a growing number of applications. Organizations are left with a choice that shouldn't exist: block traffic you can't decrypt, or accept the blind spot and allow it. Most choose the blind spot, and the bypass lists grow.



AI exposes the limit

AI did not create the architectural problem with traditional SASE, but it did make it impossible to ignore. Modern AI workflows are fundamentally non-linear:



These interactions happen entirely within the application context, at the presentation layer, before any new network connection is established. By the time traffic reaches the network, the moment of intent has already passed.

Agentic AI compounds this further. Tool calling, MCP access, agent-to-agent communication, and remote agent execution don't resemble human web traffic at all. A network proxy sees the session, but can't see what's happening inside it.

Traditional SASE vendors respond the only way their architecture permits: binary enforcement. Block AI or allow it. Organizations that block AI push usage into personal, unmanaged tools, fueling shadow AI and expanding the attack surface they were trying to reduce.

Organizations that allow AI without context accept data leakage, compliance exposure, and zero accountability.

Either way, the business loses.

Gartner expects AI security capabilities, including prompt inspection and application controls, to become a key factor in SASE platform decisions over the next two years.

Most current vendors stop at basic visibility or binary block controls. The architecture simply does not support anything more.

The architectural shift: *Enforcing at the last mile*

The architectural answer follows directly from the problem. If enforcement at the network layer cannot see what happens inside applications, the enforcement point must move closer to the work itself.

If risk lives at the presentation layer, in the actions users take inside applications, at the moment data is created, copied, uploaded, or sent, then enforcement has to live there too. Not in a distant PoP after the moment of intent has passed, but at the point of work, before the data ever leaves the endpoint.

What changes when you enforce at the last mile

Moving enforcement to the point of work changes three outcomes simultaneously:

Security gets stronger, not just different.

When policy is evaluated at the presentation layer, the visibility gap closes.

- Copy-paste actions, prompt content, tenant context, & output destinations all become visible
- Enforcement is based on what users actually do, not inferred from packet metadata
- Zero Trust is applied at the application layer per user, per session, per action

And because modern protocols are supported natively, organizations don't need bypass lists to keep critical applications functioning. *Every* session is governed.

User experience improves by design.

Most sessions go direct. There is no mandatory detour through a distant proxy, no forced TLS inspection degrading application performance, no latency from routing traffic through inspection points. Applications behave the way they were designed to behave. Users stop finding workarounds because they don't need them.

When security stops creating friction, adoption of sanctioned tools increases and IT spends less time managing exceptions.

IT operations simplified at the foundation.

One policy engine. One audit trail. No service chaining across separate consoles. No overlapping rule sets between SWG, CASB, and ZTNA producing inconsistent outcomes. Policy is defined once and enforced consistently. Deployments that used to take months are compressed to days.

This is the trifecta that traditional SASE promised but rarely delivered: superior security, a better user experience, and operations that scale without compounding complexity.

3	1	0
Concrete outcomes	Policy engine & audit trail	Bypass lists required

AI governance means *saying yes without losing control*

The question for most security teams is no longer whether employees will use AI. They already are. The question is whether you can govern it without blocking it.

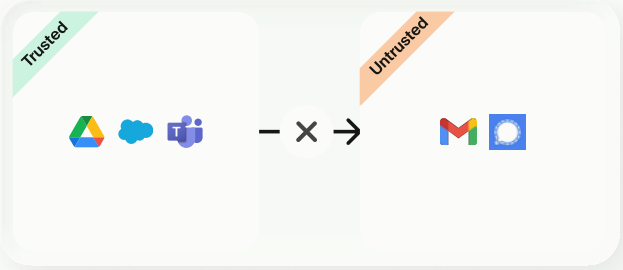
Effective AI governance at the presentation layer means seeing what network-only enforcement can't:

- What users and agents send into AI providers
- What content is being pasted or uploaded
- Where data is allowed to flow
- What AI agents are doing within an application

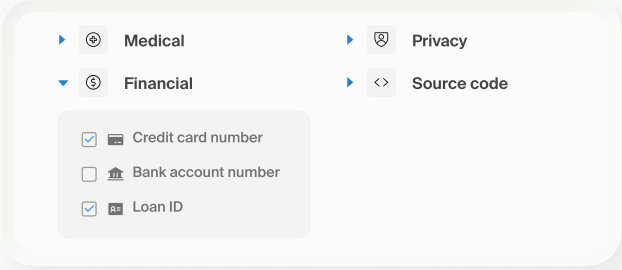
At the network layer, governance extends to agentic traffic, tool calls, MCP interactions, and agent-to-agent communications that never originate from a human session. Data lineage tracks how data moves across browsers, SaaS, desktop, AI tools, and device channels, showing where information originated and what happened along the way.

Four controls make this work in practice:

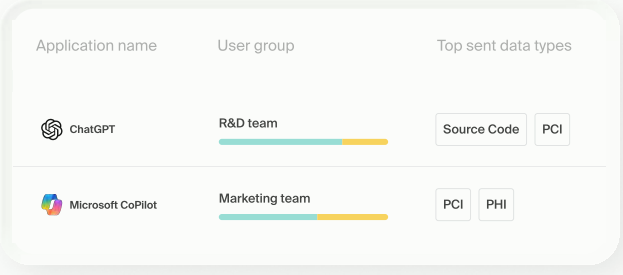
1 **Data boundaries** define which AI tools, tenants, and workflows are approved for organizational use. They enable work within defined boundaries without granting blanket permission and allow users to access personal AI tools within their personal context, without organizational data crossing into unsanctioned territory.



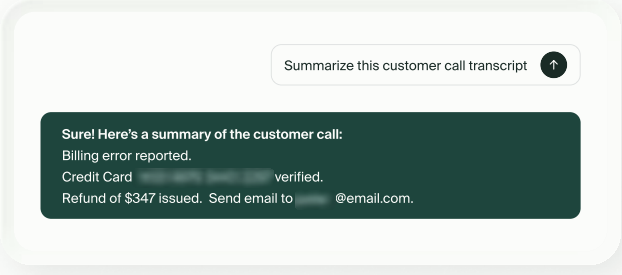
2 **Content-aware detection** inspects what users send to AI in real time (pasted text, uploads, structured and unstructured data) based on the content itself, not just the destination. A user approved to use ChatGPT for general research can be blocked from pasting regulated customer data into the same session, without disrupting the workflow.



3 **AI extension visibility and control** give IT real-time risk scoring across 200,000+ extensions, surfacing what's accessed, what's sent, and where output goes. It means shadow AI stays visible, measurable, and under control and organizations con't have to default to blanket blocking of AI extensions.



4 **Last-mile enforcement** applies policy instantly, before data leaves the device. Sensitive content can be restricted or redacted at the point of submission. AI output can be confined to approved workflows. Every prompt interaction can be logged for audit, providing accountability without surveillance.



The result is AI access that works within policy, without block pages, without shadow AI, and without forcing a choice between productivity and protection.

5

Checklist: Evaluate your current SASE stack

Before evaluating new solutions or renegotiating with an existing vendor, it's worth taking an honest look at what your current architecture is actually delivering. These questions are designed to surface the gaps dashboards tend to hide.

Checklist:

VISIBILITY

- [ ] What percentage of our traffic is actually being inspected today?
- [ ] Do we have visibility into what users copy, paste, upload, or submit inside SaaS applications? Or only at the network level?
- [ ] Can we see what employees are sending into AI tools in real time?
- [ ] If an employee pasted regulated data into a public AI tool right now, would we know?
- [x] Can we govern agentic AI workflows, tool calls, and MCP interactions or are those invisible to our current stack?

PERFORMANCE

- [ ] Are users reporting latency or application issues that trace back to proxy routing?
- [ ] How many bypass list exemptions have we added in the last 12 months?
- [ ] Do remote employees in bandwidth-constrained regions experience performance issues?

DEPLOYMENT

- [ ] What percentage of our purchased SASE capabilities are fully deployed and actively enforcing policy?
- [ ] How long did our last major rollout take from purchase to meaningful coverage?
- [ ] Are there workflows or device types that sit entirely outside our current enforcement model?

AI & AGENTIC RISK

- [ ] Do we have controls for what employees send to public AI tools or are we relying on allow/block at the domain level?
- [ ] If an employee is blocked from a sanctioned AI tool, do we have visibility into whether they've moved to a personal or unmanaged alternative instead?
- [ ] Can we inspect and govern AI-generated output? What the model returns, not just what the user submits before it moves into downstream workflows or is shared externally?

OPERATIONS

- [ ] How many separate consoles and policy engines are we managing?
- [ ] When an incident occurs, do we have a single audit trail? Or are we piecing together logs from multiple systems?
- [ ] How long does it take to onboard a new location or add a new device type to coverage?

If you've identified some gaps, that's common. The sections that follow explain the architectural shift and how organizations are closing them incrementally.

A different architectural answer: How Island approaches SASE

The approach comes from the gaps the checklist likely surfaced. If enforcement at the network layer cannot see what happens inside applications, the enforcement point has to move closer to the work itself.

Island's approach to this is built on what it calls the **Perfect Packet:** for every session, the most efficient and secure path is chosen based on policy and not a default route. In most cases, there is no traffic steering to Island's cloud as enforcement runs locally in the browser, and the session goes direct to its destination.

When steering to the cloud is needed, it travels through the most capable infrastructure available. The path is chosen per session based on policy. Most sessions go direct. Cloud inspection is invoked when policy requires it.

This is delivered through the Island Enterprise Platform: one unified control plane spanning browser, endpoint, and network. Every component shares the same policy engine and produces a single audit trail. There is no service chaining, no policy re-evaluation at different points of presence, and no need to route traffic to a separate service for inspection.

The insight behind Island's approach is straightforward: the browser is already where most enterprise work happens. Rather than backhauling that activity through a centralized inspection point, Island makes the browser itself the enforcement edge: applying policy inline, at the moment of interaction, with full visibility into what the user is actually doing.

And because work doesn't stop at the browser, enforcement doesn't either. The same policy model extends across the entire workspace to desktop applications, non-web protocols, and device-level traffic. Island evaluates identity, device posture, network and geolocation application context, and user activities at the moment of interaction, locally and in real time.

> One identity. One posture evaluation. One control plane. Regardless of where the work happens.

How Island does SASE

Others

- Poor user experience
- Complex, expensive architecture
- Prone to outage/ service disruption
- SSL break/inspect for DLP, SWG, DEX, visibility
- Growing volume of traffic bypasses inspection
- Not build for AI-first world

Island

- Traffic goes direct - no friction
- Less backhauling, less SSL inspection
- 3 hyperscalers, 2 network stacks.
- Most traffic does not require SSL inspection
- Expanded visibility by being on the endpoint
- AI workflows are integrated in the user experience

The full Island SASE capability set includes:

Data Protection

Last-mile DLP in the browser and at the endpoint, without decrypt-everything architectures.

Island Private Access

Zero trust network access (ZTNA) to private applications without the needs for agents, VPNs , or exposed networks.

Cloud Access Security Broker (CASB)

Visibility and control inside SaaS environments through native APIs, monitoring files, permissions, and configurations without rerouting traffic.

Secure Web Gateway

Precision web security without default backhaul or forced TLS inspection.

AI Protection

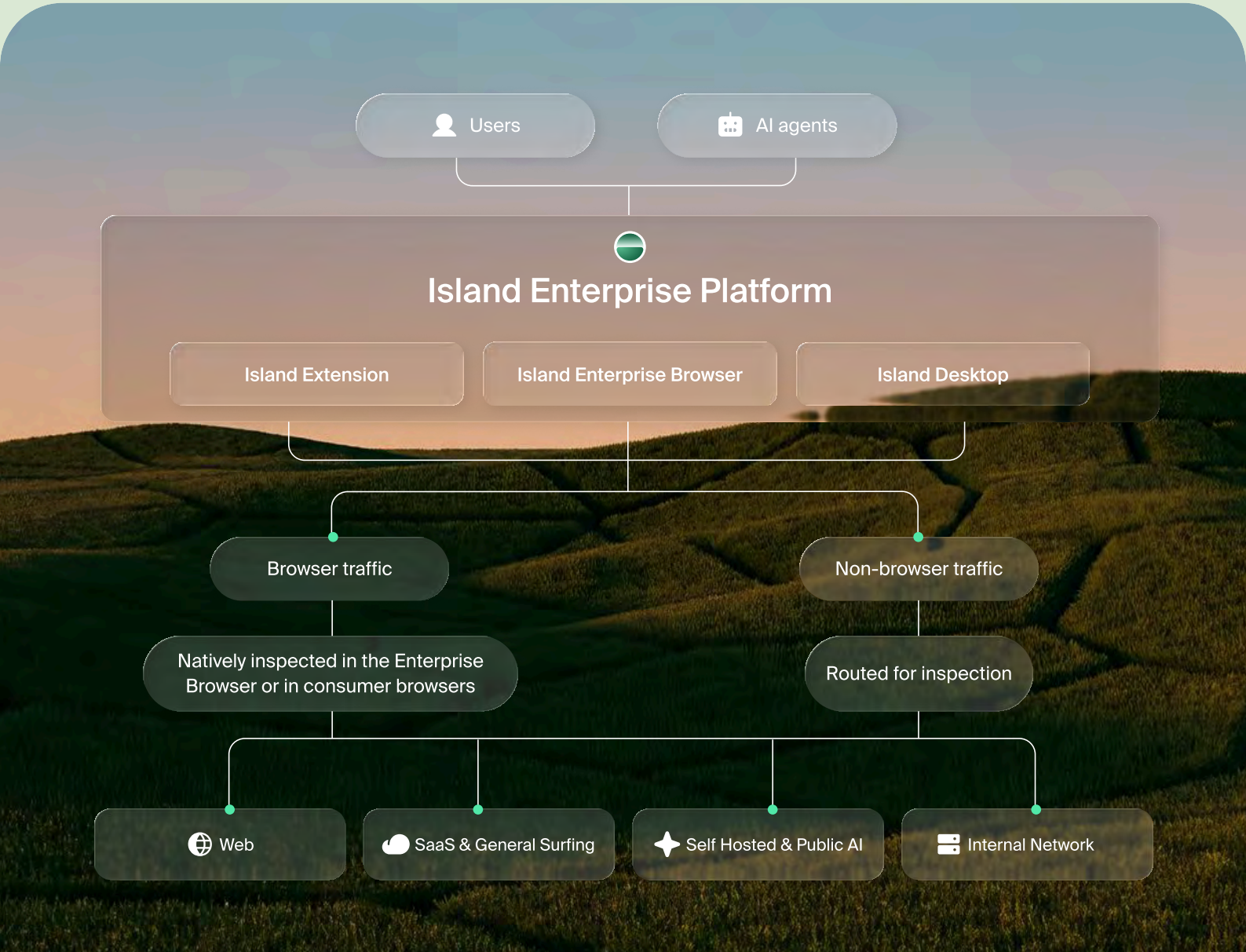
Governs prompts, tools, MCPs, uploads, and generated output across users, devices, and agents, with a full audit trail of every AI session.

Remote Browser Isolation

Cloud-rendered protection invoked only for high-risk destinations.

Digital Experience (DEX) Monitoring

Real-time insight into application performance, device health, and employee experience.



How these capabilities are delivered

There are three enforcement surfaces and one global network, depending on where the work happens.

1

The Enterprise Browser is the primary enforcement point for web and SaaS activity. Policy is applied inline at the moment of interaction. Built on Chromium, it supports modern protocols natively, including IPv6, TLS 1.3, HTTP/3 and post-quantum cryptography, without needing to disable security and performance features.

Example: An employee attempts to download a file from Salesforce and move it to a personal account. Island detects the action at the moment it occurs and blocks the transfer before the data leaves the session.

2

The Island Extension extends the same enforcement model to users on existing consumer browsers, with no full browser migration required. For organizations adopting incrementally, this is the natural entry point.

Example: An organization rolls out the Island Extension to 5,000 users on existing Chrome browsers in a single afternoon. No network changes. No agent deployment. Policy enforcement starts immediately.

3

Island Desktop extends the policy model beyond the browser to desktop applications, non-web protocols, and device-level traffic. Built on WireGuard for high throughput and full protocol support, no additional agents are needed.

Example: A finance user opens a desktop trading application that connects to internal data. Island Desktop grants access based on user identity and device posture, while blocking outbound connections from that application to unsanctioned destinations. One audit trail covers the entire session.

4

The Global Multi-Cloud Network provides resilience, failover, and centralized visibility across all paths. Island's network is purpose-built across Google Cloud, Microsoft Azure, and AWS, with over 100 points of presence globally. A dual-network architecture reduces shared failure domains and eliminates single-cloud dependencies.

Example: When a regional cloud provider experiences an outage, Island's dual-network architecture automatically reroutes traffic across its remaining infrastructure. The disruption stays invisible to the people doing the work. IT sees the event in the audit trail.

Beyond web traffic: *firewall & network enforcement*

Island includes firewall and network enforcement for all outbound traffic. Basic firewall controls are delivered at the endpoint today, with advanced cloud firewall capabilities, intrusion prevention, and deep inspection available as the platform expands. Unlike traditional SASE, firewall services support the architecture rather than define it.

7

Getting there: Phased deployment *without rip-and-replace*

The most predictable SASE failure mode isn't a security breach. It's a deployment that is never completed.

Licenses are purchased. Rollouts begin. Environments change faster than the deployment plan can accommodate. Exception lists grow. Policies drift. By the time the project reaches "full deployment," the organization has moved on, and the investment is partially realized.

Island's deployment model is designed to avoid this. Each phase delivers standalone value. There is no waiting for full deployment to see results. Security improves incrementally, and because coverage follows adoption rather than the other way around, the shelfware problem disappears by design.

As one Island solution engineer puts it:

"I'll set up just one hour with a customer and we'll have a tenant created, a manager created, Island downloaded, and we're protecting Office 365. With traditional SASE, you spend the first week and a half troubleshooting what the agent is breaking on the endpoint. With Island: Can you install a browser? Great. Then you can install Island."

1 *Consumer browser-based enforcement*

Deploy the Island Extension on existing browsers to establish immediate policy control and visibility. No network reconfiguration. No agent conflicts. Governance starts on day one.

2 *Enterprise Browser for higher-risk workflows*

Deploy the Island Enterprise Browser for workflows that require deeper controls, richer context, and stronger governance. Rollout is measured in days, not months. Enforcement happens directly in the workspace.

3 *Device-wide coverage with Island Desktop*

Extend the same policy beyond the browser to private applications, desktop applications, legacy protocols, and sensitive device workflows. One identity, one posture evaluation, and one policy fabric across browser and device.

4 *Network steering only when required*

For environments that cannot run browser or endpoint enforcement, such as legacy devices, branch locations, IoT and OT infrastructure, apply selective DNS steering or IPsec tunnels. Layer it in. No rip-and-replace.

Five use cases, *five risk areas closed*

The following use cases each close a specific risk introduced earlier in this guide. They are drawn from real deployment patterns and show what the architectural shift looks like when it meets actual workflows.

VPN Replacement

The risk

Users connecting to private applications through VPNs gain broad network access, not application access. Lateral movement isn't an accidental risk with VPNs, it's a consequence of how they work.

In practice

A regional sales manager needs access to an internal CRM and a finance reporting tool. With a VPN, she joins the network and everything on it. With Island Private Access, she connects directly to those two applications, based on her identity and device posture, with no network exposure and no lateral movement risk. The applications perform at native speed. IT gets a full audit trail.

What changes

Per-application, per-session Zero Trust access replaces network-level trust. Access is precise, auditable, and no longer contingent on joining the network.

Real-world example

University of the Pacific ↪

Data Protection in Use

The risk

Traditional DLP operates at the file level or the network level, catching data after it moves, or missing it entirely when it moves through actions rather than file transfers.

In practice

A finance analyst opens a regulated spreadsheet, copies a section of customer financial data, and attempts to paste it into a personal Google Doc. With traditional SASE, that action is invisible. With Island, the copy event is governed at the moment it occurs. The paste is blocked based on content sensitivity and destination context. The analyst receives a clear, non-disruptive explanation. The data stays where it belongs.

What changes

Enforcement moves from file transfers to user actions. Copy, paste, upload, download, print, and screenshot are all governed based on content, context, and identity, before the data moves.

Real-world example

Eftsure ↪

Secure Internet and SaaS Access

The risk

Web and SaaS policies enforced through distant proxies introduce latency, break modern application behavior, and create bypass lists that quietly reduce coverage.

In practice

A distributed engineering team relies on GitHub, Jira, and Figma for daily work. With traditional SASE, TLS interception degrades performance and periodically breaks integrations, forcing IT to add exception lists. With Island, SWG enforcement happens inline in the browser before content renders. Browser traffic goes direct: no proxy detour, no forced TLS inspection. Tenant-aware controls distinguish corporate from personal SaaS sessions in real time.

What changes

Backhaul becomes optional. SaaS performance is restored. Enforcement becomes more precise.

AI Governance

The risk

AI tools are already embedded in daily work. Blocking them increases shadow IT. Allowing them without context creates data leakage, compliance exposure, and zero accountability.

In practice

A software team deploys an AI coding agent that calls external APIs, accesses internal documentation, and pushes output into a development pipeline. With traditional SASE, these interactions are invisible. There's no human web session to inspect and no standard network flow to analyze. With Island, the agent's interactions are governed at the presentation layer. Tool calls are evaluated against policy. Access to internal repositories is scoped by identity and context. Data cannot be routed to unsanctioned destinations, and every action is logged. The agent operates within approved boundaries without IT having to choose between enabling it and securing it.

What changes

The organization stops choosing between AI productivity and data protection. Employees get access to the tools that make them faster. IT gets the visibility and control that makes that access defensible.

BYOD and Unmanaged Devices

The risk

Employees and partners working from unmanaged devices sit outside traditional enforcement models, either excluded from access entirely or granted access without meaningful control.

In practice

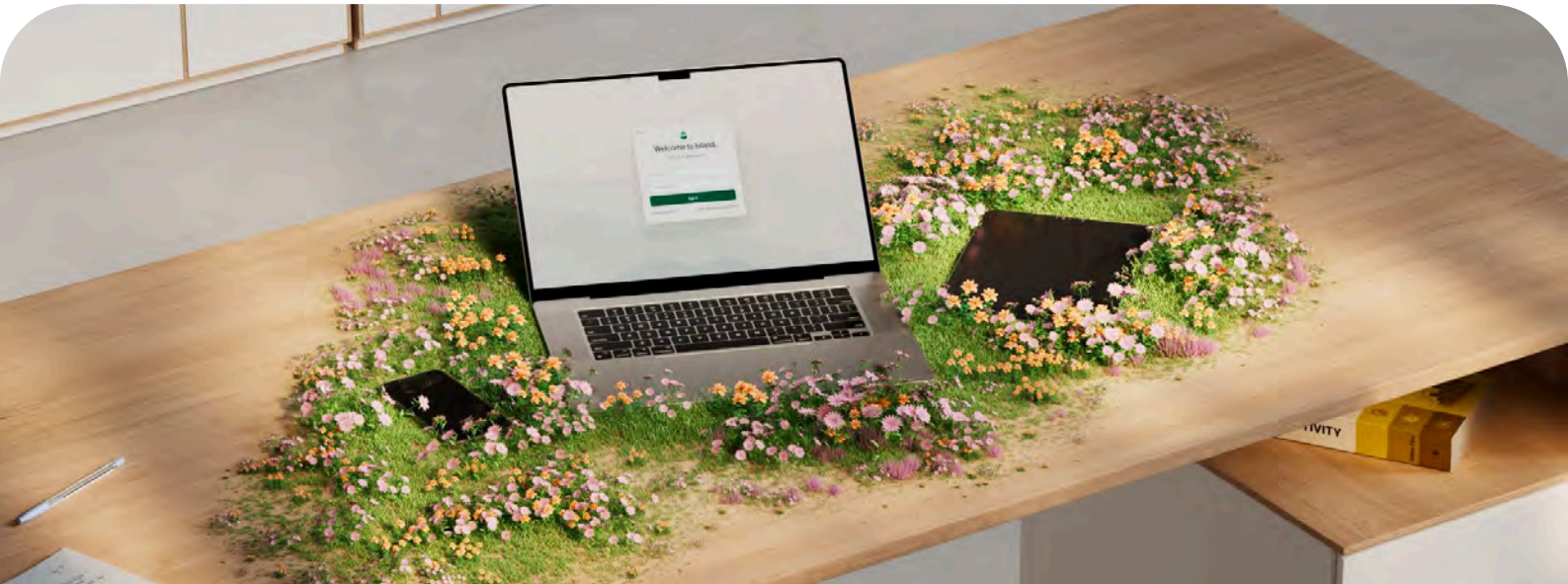
A partner organization needs access to a shared project portal and a document collaboration tool. Their devices are unmanaged. IT will never control them. With Island, access is granted through the Enterprise Browser with full policy enforcement, no MDM, no heavy agents, no broad network access. Data stays within approved workflows. The partner onboards in hours, not weeks. When the engagement ends, access is revoked instantly.

What changes

Secure access no longer requires device ownership. Client onboarding moves four times faster. No infrastructure to provision, no lengthy testing cycles.

Real-world example

TaskUs ↪



CONCLUSION

SASE was built on a sound premise: security and networking should converge into a unified, cloud-delivered architecture that scales with the modern enterprise. That premise remains valid.

What has changed is where enforcement needs to live to make that premise real.

The network is where work travels, occasionally, selectively, and increasingly over protocols that resist interception by design.

Modern work happens inside applications, at the moment a user copies data, submits a prompt, uploads a file, or moves output into a downstream tool. That is where identity, context, and intent are clearest. And that is where enforcement has to be.

Organizations that rely on network-centric enforcement to govern AI will find themselves choosing between productivity and protection, repeatedly, at scale, without a good answer. Organizations that enforce policy at the point of interaction will find AI governance is not a constraint on how people work. It is the condition that makes it safe to work faster.

The ground has shifted. Enforcement has to follow.