DETERMINISTIC SECURITY STANDARD · AGSSH-STD-001 · v1.5.1

Air-Gapped Static Surface Hardening

A Property-Based, Risk-Proportional Conformance Standard for Client-Side Tools, Docs & Sites

v1.5.1 · August 2026 · Author **Fabrizio Salmi** · 57 rules · 9 families · Bronze / Silver / Gold profiles · containment-first · external signed attestation

§0 Normative Model, Profiles & Assurance

Versioning. This standard follows Semantic Versioning: rule IDs are stable and never reused; adding a rule or profile is a MINOR release, tightening or removing a MUST is a MAJOR. **This is v1.5.1**, incorporating the responses to an adversarial review: property-based conformance, containment-first verification, external signed attestation, governed waivers, and risk-proportional profiles.

Normative language

MUST, **MUST NOT**, **SHOULD**, **SHOULD NOT** and **MAY** follow RFC 2119 / RFC 8174. A MUST is mandatory and **can never be waived** (AG-GOV-01). A SHOULD is strongly recommended; a deviation is permitted only under the waiver governance of §10 — bounded, expiring, attributable. There are no silent skips.

Properties, not mechanisms

Each rule specifies a security *property* (e.g. “the page cannot be framed”, “no third-party egress”), satisfiable by any control in its equivalence set and recorded with an *assurance tier*. The standard mandates the property; the mechanism is a documented choice. This is what makes it vendor-neutral: a stateless GitHub Pages site (Tier D) and a header-setting CDN (Tier A) can both satisfy the anti-framing property — pure static hosting is not condemned, and putting a CDN in front is an explicit, recorded expansion of the trust chain, never an implicit mandate.

Tier	Enforcement	Example
A	Browser-enforced via HTTP response header	X-Frame-Options / header CSP frame-ancestors
B	Browser-enforced via <meta> (directives the meta form honors)	meta CSP script-src / connect-src
C	Runtime / script-enforced	frame-buster on a header-less host
D	Structural — the protected asset does not exist	statelessness removes clickjacking impact entirely

Containment over observation

Guarantees come from **capability boundaries enforced in production for every visitor** — a CSP that denies all third-party fetch-directives, an egress allowlist — not from observing the absence of bad behavior in a test. A time-delayed, environment-detecting or interaction-gated exfiltration attempt still strikes the production boundary. Verification therefore proves the boundary *blocks a known probe* (the egress canary, AG-GOV-06) and is corroborated by same-origin violation reporting from real users; it does not certify a negative. CSP is not a complete egress oracle — WebRTC and DNS-prefetch are residual channels that Gold profiles deny explicitly and the canary probes.

Conformance is binary — within a declared profile

Burden must match value, so the rules are partitioned into three cumulative profiles. A surface declares its **target profile** in `.airgap.yml`; the gate enforces exactly that profile's subset. A surface is **conformant at its target profile** iff every MUST in that subset passes and every SHOULD passes or carries a governed deviation. A single failed MUST = non-conformant; the weighted score (§12) only orders remediation.

Profile	For	Adds
Bronze — Baseline	any static surface (docs, small tools); an afternoon's work, no CI/signing	egress boundary, CSP lockdown + anti-framing property, core security headers, no secrets in output, no known-CVE deps, no pre-consent telemetry, self-hosted fonts, in-browser-only output
Silver — Hardened	tools handling user data or commercially deployed	supply-chain integrity, CI hardening, DNS trust anchors, TLS floor, remaining headers/privacy, the egress canary, waiver budget & auto-expiry
Gold — Sovereign	high-assurance / regulated / sovereign-cloud	release signing & provenance, Trusted Types everywhere, cross-origin isolation, external signed attestation, ephemeral hermetic build, segregation-of-duties waivers

Profiles are cumulative: Gold ⊃ Silver ⊃ Bronze. Each rule below is tagged with the lowest profile that requires it.

Governed waivers (not free text)

The deviation mechanism is the most-attacked part of any standard, so it is constrained by construction: MUSTs are non-waivable (AG-GOV-01); every waiver carries an enforced expiry after which the runner re-enforces the rule (AG-GOV-02); a per-surface debt ceiling caps total active waivers (AG-GOV-03); and at Gold each waiver is signed by an approver other than the change author (AG-GOV-04). The runner governs the *mechanics*; the *truthfulness* of a justification is a named human's accountable decision — this standard does not pretend a deterministic runner can judge intent.

Evidence & external verification

The authoritative conformance record is produced by an **independent runner observing the live deployed surface** (headers, egress, DNS and TLS are externally observable), signed and bound to the artifact digest (AG-GOV-05) — so a compromised build host cannot fabricate production reality. Build-host integrity is treated as a trust dependency defended by ephemeral hermetic builds (AG-GOV-07) and that external attestation; its full specification is out of scope (Appendix F), but it is not assumed away.

Anti-paralysis

Rigidity that exhausts teams gets bypassed, so the standard mandates the labor-saving automation alongside the control: a bot keeps pinned Action SHAs current (AG-CI-01) so no one edits 40-character hashes by hand; CSP is staged report-only before enforcing (AG-CSP-04); and a conformant starter (the `_headers`, `.airgap.yml` and CI gate) ships green by default, making conformance the path of least resistance rather than an obstacle.

Public, vendor-neutral standard; §10 governs conformance integrity and §11 maps controls to backends. Legal references in Appendix E are informative, not legal advice.

§1 Threat Model & Conformance Levels

Scope. Any deployed *static surface* — a client-side tool, a documentation site, or a marketing site — whose security posture rests on the promise that user data and document content never leave the browser, and ideally that the page performs no unsanctioned network egress at all. The rules are backend-agnostic; **§11** maps each

control onto what GitHub Pages, Cloudflare Pages, proxied Cloudflare, Fastly, Netlify and CloudFront can actually enforce.

Threat model. A passive network observer; a compromised or hostile third-party CDN serving the code that touches the document; a hostile embedder (clickjacking); a compromised build pipeline or dependency; data exfiltration through any third-party request; subdomain takeover; and regulatory exposure created by pre-consent third-party calls. Every rule composes into a deterministic conformance gate — the "deterministic labyrinth": scan → per-rule pass/fail → weighted score → fix queue → retained evidence.

Conformance levels

Level	Meaning	Defining control	Typical surface
L0 — Air-Gapped Strict	Client-side tool with no legitimate network need	`connect-src 'none'`, egress set ⊆ {self}	in-browser TLS / image / PDF utilities
L1 — Scoped Egress	Needs the network, but only to known origins	`connect-src` = explicit allowlist (never `*`)	agent talking to a local LLM or one API
L2 — Public Static	May include third parties, all non-essential gated	no pre-consent egress; trackers behind consent	product / docs / marketing site

§2 Egress & Air-Gap

AG-NET

AG-NET-01 Self-host every runtime dependency CRITICAL Bronze

MUST · L0 L1 L2 · static+dynamic

The third-party egress set of a surface is the UNION of every host it can reach: all CSP fetch-directive sources (script/style/img/font/media/connect/worker/frame/manifest/prefetch-src) plus every runtime sink (fetch, XHR, sendBeacon, EventSource, WebSocket, new Worker, importScripts, WebAssembly.instantiate*, dynamic import). For L0 that set MUST be a subset of {self}. The canonical breach is a browser OCR/codec engine fetched from a CDN on use — the document stays local, the engine does not.

CHECK

```
egress = union(all CSP fetch-dir hosts, all runtime request hosts)
assert egress ⊆ {self}      # not connect-src alone — the whole set
```

REMEDIATION

Vendor all assets same-origin; ship the .wasm, workers and data packs from your own origin.

Tooling: chromedp request capture + CSP parser

AG-NET-02 Override library default CDN loaders CRITICAL Bronze

MUST · L0 L1 L2 · static

Tesseract.js (workerPath/corePath/langPath → jsDelivr + the tessdata host), jsPDF (the pdfobject viewer → cdnjs), pdf.js workers and ML model loaders all fetch from a CDN by default unless explicitly repointed. The default is invisible egress that CSP-less testing never reveals until it fires.

CHECK

```
grep dist: jsdelivr|unpkg|cdnjs|tessdata|projectnaptha => 0 hits
assert every loader path (workerPath/corePath/langPath/…) is same-origin
```

REMEDIATION

Pass explicit same-origin paths to every loader; treat a surviving default-CDN literal as a build failure.

Tooling: ripgrep over dist/

AG-NET-03 Same-origin workers; constrain worker-src HIGH Silver

MUST · L0 L1 L2 · static

A Blob worker doing importScripts('https://cdn…'), or a cross-origin module worker, re-introduces code that SRI cannot cover (there is no integrity for importScripts or module workers).

CHECK

```
every importScripts(/new Worker( arg resolves same-origin ;
CSP: worker-src 'self' ; child-src 'none'
```

REMEDIATION

Bundle workers locally; point any library workerPath at a same-origin file.

Tooling: AST/regex scan + CSP parser

AG-NET-04 Constrain connect-src CRITICAL Bronze

MUST · L0 L1 (conditional · L2) · static

The browser-enforced air-gap for XHR/fetch/WebSocket. connect-src 'none' makes that exfiltration channel impossible regardless of any bug or injection. Tools that genuinely need the network declare an explicit allowlist of exact origins — never '*', never a bare scheme.

CHECK

```
parse CSP connect-src:
  L0 => exactly 'none' ; L1 => literal-origin allowlist (no '*'/https:)
  L2 => must exclude every unconsented tracker host
```

REMEDIATION

Set connect-src 'none' for pure client-side tools; for an agent talking to a local LLM scope to that exact origin only.

Tooling: CSP parser

AG-NET-05 No third-party egress during load HIGH Silver

MUST · L0 (SHOULD · L1) · dynamic

The air-gap claim must be observable, not asserted. A headless load of the surface must contact NO third-party host: every responding origin is the surface's own. This is the necessary, directly-measurable half of 'works offline' — the network is observed during a normal load, not cut, so it proves no cross-origin dependency fired, not that a full offline smoke test passed.

CHECK

```
headless load; assert every responding host is same-origin
(zero third-party responses observed during the load)
```

REMEDIATION

Eliminate cross-origin subresources and runtime calls (AG-NET-01/02/04); serve every asset and data pack from your own origin.

Tooling: chromedp request/response capture

AG-NET-06 No third-party connection priming MEDIUM Silver

MUST · L0 L1 (SHOULD · L2) · static

preconnect/dns-prefetch/prefetch to third parties leak intent and warm tracker connections before the visitor interacts at all.

CHECK

```
all rel in {preconnect,dns-prefetch,preload,prefetch,modulepreload}
hosts ⊆ {self}
```

REMEDIATION

Remove off-origin resource hints; keep only self-targeted preloads.

Tooling: HTML parser

AG-NET-07 Service Worker hygiene HIGH Silver

MUST · when a service worker is registered · dynamic

A service worker is persistent code that intercepts every request and can cache and replay responses. A compromised or over-scoped SW is a durable foothold that survives page reloads.

CHECK

```
SW served same-origin ; narrowest scope ; fetch handlers contact
only self ; no remote importScripts ; cache stores only same-origin
```

REMEDIATION

Register the narrowest scope; never importScripts a remote URL; restrict the cache to same-origin responses.

Tooling: chromedp SW inspection

AG-NET-08 Violation/error reporting stays same-origin MEDIUM Silver

MUST · L0 L1 · static

CSP report-to / Reporting-Endpoints / NEL pointed at a third-party SaaS collector silently re-introduces egress and ships metadata about every visitor and violation off-origin — defeating the air-gap through the back door.

CHECK

```
Reporting-Endpoints / report-to / NEL targets ⊆ {self}  OR  absent
```

REMEDIATION

Collect reports at a same-origin endpoint, or disable reporting on air-gapped surfaces.

Tooling: header + CSP parser

AG-NET-09 External _blank links sever the opener; privacy surfaces sever the referrer

MEDIUM Bronze

MUST · L0 L1 L2 · static

Property: no <a target=_blank> exposes window.opener to the destination (reverse tabnabbing). The opener is severed by rel containing 'noopener' OR 'noreferrer' ('noreferrer' implies 'noopener' per the HTML spec). On privacy surfaces the referrer is also suppressed. Browsers now imply noopener for target=_blank by default — treat that as defence-in-depth, not as the control.

CHECK

```
for every <a target=_blank>: rel contains 'noopener' or 'noreferrer' ;
on privacy surfaces rel also contains 'noreferrer'
```

REMEDIATION

Add rel="noopener noreferrer" to external _blank links; noreferrer also blocks referrer leakage.

Tooling: HTML anchor scan

§3 Content Security Policy

AG-CSP

AG-CSP-01 Ship a CSP HIGH Bronze

MUST · L0 L1 L2 · static

A CSP is the single highest-leverage control on a static surface. Header-delivered is preferred; a <meta http-equiv> CSP is the fallback only for backends that cannot set response headers (GitHub Pages).

CHECK

```
response header content-security-policy present
OR <meta http-equiv='Content-Security-Policy'> present (record which)
```

REMEDIATION

Add the policy at the edge where possible; otherwise emit a meta CSP and accept the AG-CSP-03 limitation.

Tooling: header + HTML parser

AG-CSP-02 Deny by default; enumerate; forbid bypass primitives HIGH Bronze

MUST · L0 L1 L2 · static

A permissive CSP is theatre. The baseline denies by default, enumerates only what is used, and removes every known bypass primitive.

CHECK

```
default-src 'none'(or 'self'); object-src 'none'; base-uri 'none'|'self';
explicit script/style/img/font/connect/worker/frame/manifest-src;
form-action 'self'|'none'; NO 'unsafe-eval'; NO 'unsafe-inline' on
script-src; NO data:/blob: in script-src; NO wildcard or scheme-only;
'strict-dynamic' only with nonces
```

REMEDIATION

Hash or nonce inline scripts; delete unsafe-eval / unsafe-inline(script) / wildcards; add the missing fetch directives explicitly.

Tooling: CSP directive-predicate parser

AG-CSP-03 Clickjacking control must be header-delivered HIGH Bronze

MUST · L0 L1 L2 · static

frame-ancestors, report-uri/report-to and sandbox are IGNORED when a CSP arrives via <meta>. A meta-only surface therefore has NO clickjacking protection from frame-ancestors — it must come from an X-Frame-Options header. Where neither header can be set (pure GitHub Pages), the surface MUST be stateless and the residual risk recorded.

CHECK

```
if meta-CSP contains frame-ancestors => FAIL (ineffective)
require X-Frame-Options header  OR  declared stateless: true
```

REMEDIATION

Move clickjacking control to an XFO header (front with a CDN if the host can't set it), or keep the surface free of auth and state-changing actions.

Tooling: header + CSP parser

AG-CSP-04 Stage report-only before enforcing MEDIUM Silver

SHOULD · all (mandatory before first enforcing deploy) · manual

A CSP that breaks the surface gets disabled in a panic — worse than none. Report-only first turns breakage into telemetry.

CHECK

```
Content-Security-Policy-Report-Only canary in staging before the
enforcing policy ships ; violations reviewed
```

REMEDIATION

Deploy report-only, watch violations, then promote to enforcing.

Tooling: staging header + report log

AG-CSP-05 Force HTTPS subresources MEDIUM Silver

MUST · L0 L1 L2 · static

A single http:// subresource downgrades the whole page's integrity guarantee.

CHECK

```
upgrade-insecure-requests present ; grep subresources 'http://' => 0
```

REMEDIATION

Add the directive; convert plain-HTTP references to HTTPS or same-origin.

Tooling: CSP parser + HTML scan

AG-CSP-06 **Trusted Types for DOM sinks** HIGH Silver

MUST · L0 L1 (SHOULD · L2) · static+dynamic

Trusted Types eliminate DOM-XSS by forbidding raw string assignment to injection sinks (innerHTML, script.src, eval). For a tool parsing untrusted input — certificates, uploaded files — this closes the highest-impact client-side bug class outright.

CHECK

```
CSP require-trusted-types-for 'script' ;
trusted-types <named-policy-list>  (no '*'; 'none' if no policy needed)
```

REMEDIATION

Enable Trusted Types; route any required HTML through one vetted, sanitizer-backed policy.

Tooling: CSP parser + runtime violation check

§4 Transport & Response Headers

AG-HDR

AG-HDR-01 **HSTS with a real max-age** HIGH Bronze

MUST · header-capable backends · static

max-age=0 DISABLES HSTS and disqualifies the domain from preload while still looking configured — the most common silent regression: the toggle is on, the value is zero.

CHECK

```
Strict-Transport-Security: max-age >= 31536000; includeSubDomains;
preload      (max-age < 31536000 => FAIL)
```

REMEDIATION

Set max-age to one year. On Cloudflare this lives under SSL/TLS → Edge Certificates → HSTS — check the value, not just the switch.

Tooling: header parser

AG-HDR-01a **HSTS scope is sticky — gate it** HIGH Silver

MUST · before enabling AG-HDR-01 · manual

includeSubDomains + preload apply to every subdomain and are hard to reverse. Enable them before a subdomain is permanently on HTTPS and you can lock yourself out of it.

CHECK

```
enumerate every subdomain => each returns valid HTTPS
(GitHub Pages: 'Enforce HTTPS' ON) ; record preload submission state
```

REMEDIATION

Verify all subdomains over HTTPS first, then add includeSubDomains/preload, then submit to the preload list.

Tooling: subdomain enumeration + curl

AG-HDR-02 **Disable MIME sniffing** HIGH Bronze

MUST · header-capable · static

Without nosniff a browser may reinterpret an asset's type and execute content never meant to run.

CHECK

```
X-Content-Type-Options: nosniff
```

REMEDIATION

Emit the header at the edge / in _headers.

Tooling: header parser

AG-HDR-03 X-Frame-Options HIGH Bronze

MUST · header-capable (pairs with AG-CSP-03) · static

The reliable, universally-honored clickjacking control; pairs with header CSP frame-ancestors.

CHECK

```
X-Frame-Options: DENY  (or SAMEORIGIN only if same-origin
framing is genuinely required)
```

REMEDIATION

DENY unless the surface is intentionally self-embedded.

Tooling: header parser

AG-HDR-04 Referrer-Policy MEDIUM Silver

MUST · header-capable · static

Stops the surface from leaking the visited URL to outbound links and resources.

CHECK

```
Referrer-Policy in { no-referrer,
  strict-origin-when-cross-origin, same-origin }
```

REMEDIATION

no-referrer for privacy tools; strict-origin-when-cross-origin for marketing.

Tooling: header parser

AG-HDR-05 Permissions-Policy MEDIUM Silver

MUST · header-capable · static

Affirmatively switches off powerful features the surface never uses, including the FLoC/Topics opt-out.

CHECK

```
Permissions-Policy disables unused features, e.g.: camera=(),
microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=()
```

REMEDIATION

Default-deny every feature; open only what the tool needs.

Tooling: header parser

AG-HDR-06 Cross-origin isolation for threaded WASM MEDIUM Gold

MUST · when SharedArrayBuffer / threaded WASM is used · conditional

Threaded/SIMD WASM (e.g. Tesseract with threads) needs SharedArrayBuffer, which requires cross-origin isolation; this also hardens against Spectre-class cross-origin reads.

CHECK

```
if SAB used => Cross-Origin-Opener-Policy: same-origin AND
Cross-Origin-Embedder-Policy: require-corp
(Cross-Origin-Resource-Policy: same-origin on served assets)
```

REMEDIATION

Set COOP+COEP at the edge; serve assets with CORP same-origin. Not needed for single-threaded WASM.

Tooling: header parser + runtime crossOriginIsolated check

AG-HDR-07 Cache discipline for fingerprinted assets LOW Silver

SHOULD · header-capable · static

Content-hashed assets are immutable by construction; HTML must never be long-cached or stale tools linger.

CHECK

```
hashed assets (/_astro/*, fingerprinted) => Cache-Control: public,
max-age=31536000, immutable ; HTML => no-cache / short
```

REMEDIATION

Long-cache hashed bundles, short-cache documents.

Tooling: header parser

AG-HDR-08 TLS floor HIGH Silver

MUST · header-capable · dynamic

A modern header set over a weak TLS channel is a contradiction. The transport must refuse legacy protocols and ciphers.

CHECK

```
min TLS 1.2 (prefer 1.3); no SSLv3/TLS1.0/1.1; no RC4/3DES/EXPORT;
OCSP stapling; valid chain
```

REMEDIATION

Set the edge/CDN TLS profile to modern; disable legacy protocols/ciphers; enable stapling.

Tooling: testssl.sh / sslyze

§5 DNS & Issuance Trust Anchors

AG-DNS

AG-DNS-01 CAA pins certificate issuance HIGH Silver

MUST · L0 L1 L2 · static

Without CAA, any public CA can be induced to issue a certificate for your domain. CAA restricts issuance to the CA(s) you actually use and provides an incident contact.

CHECK

```
dig CAA => issue/issuewild restricted to your ACME CA(s) ; iodef set
```

REMEDIATION

Publish CAA on the apex (and wildcards) naming your CA(s); add an iodef mailto.

Tooling: dig CAA / DNS API

AG-DNS-02 DNSSEC enabled MEDIUM Silver

SHOULD · L0 L1 L2 · static

DNSSEC prevents forged DNS answers from silently redirecting your domain.

CHECK

```
dig +dnssec shows AD ; DS record present at the registrar
```

REMEDIATION

Enable DNSSEC at the DNS provider and publish the DS at the registrar.

Tooling: dig +dnssec / delv

AG-DNS-03 No danglable records (subdomain takeover) HIGH Silver

MUST · L0 L1 L2 · static

A CNAME to a deprovisioned service — a removed Pages site, an unclaimed bucket — is a hijackable subdomain: a live foothold under your brand. Mixed estates (Pages subdomains + a proxied apex) are the usual culprit.

CHECK

```
every CNAME target resolves to a claimed/active resource ;
no NXDOMAIN / unclaimed-service takeover fingerprints
```

REMEDIATION

Remove or repoint stale records the moment a service is decommissioned.

Tooling: dnsReaper / subjack

§6 Supply Chain & Build Integrity

AG-SUP

AG-SUP-01 SRI on any unavoidable cross-origin subresource HIGH Silver

MUST · L0 L1 L2 · static

Eliminating cross-origin subresources (AG-NET-01) makes this moot — the strongest answer. Where one must remain, integrity pinning stops a compromised CDN swapping the file.

CHECK

```
every cross-origin <script>/<link rel=stylesheet> has integrity=
AND crossorigin=   ; else FAIL
```

REMEDIATION

Prefer self-hosting; otherwise add SRI hashes and re-pin on every version bump.

Tooling: HTML parser

AG-SUP-02 Pin dependencies by integrity hash HIGH Silver

MUST · L0 L1 L2 · static

A floating version is an unsigned cheque to whoever controls the registry or CDN. Pinning by content hash (not just semver) defeats tampering and typosquat republish.

CHECK

```
lockfile with integrity hashes committed ; install uses npm ci /
--frozen-lockfile ; no '@latest' / unpinned '@vN' / floating CDN
```

REMEDIATION

Commit the lockfile; install frozen; pin any surviving CDN ref to an exact version + SRI.

Tooling: lockfile lint + ripgrep

AG-SUP-03 Reproducible build; pinned toolchain MEDIUM Silver

SHOULD · L0 L1 L2 · static

A reproducible build is auditable; a drifting toolchain is not.

CHECK

```
.nvmrc/engines present ; SSG pinned ; base image pinned by digest ;
build is deterministic
```

REMEDIATION

Pin Node, the SSG, and any container base image by digest.

Tooling: build-config lint

AG-SUP-04 No secrets in shipped output HIGH Bronze

MUST · L0 L1 L2 · static

The build artifact is public the instant it deploys; a baked token or private key is immediately compromised.

CHECK

```
secret scan of the BUILD DIR => 0 hits
```

REMEDIATION

Scan dist/ in CI; reject on any hit; rotate anything leaked.

Tooling: gitleaks / trufflehog over dist/

AG-SUP-05 No unintended source maps LOW Silver

SHOULD · L0 L1 L2 · static

Shipped .map files hand an attacker your original source and structure.

CHECK

```
no sourceMappingURL / .map served in production
(unless intentional and documented)
```

REMEDIATION

Disable production source-map emission, or host maps privately.

Tooling: ripgrep over dist/

AG-SUP-06 No known-vulnerable dependencies HIGH Bronze

MUST · L0 L1 L2 · static

Shipping a dependency with a known high/critical CVE hands the attacker a public exploit.

CHECK

```
vuln scan of resolved deps => 0 high/critical (or each triaged
with a recorded, time-bounded waiver)
```

REMEDIATION

Upgrade or replace; gate the build; waive only with justification + expiry.

Tooling: osv-scanner / trivy fs / npm audit --audit-level=high

AG-SUP-07 SBOM emitted and retained MEDIUM Gold

SHOULD · L0 L1 L2 · static

You cannot respond to the next supply-chain CVE for components you cannot enumerate.

CHECK

```
CycloneDX/SPDX SBOM produced per build and retained with the release
```

REMEDIATION

Generate an SBOM in CI and attach it to the artifact.

Tooling: syft / cdxgen

AG-SUP-08 Signed releases / build provenance MEDIUM Gold

SHOULD · L0 L1 L2 · static

Signatures and provenance let consumers verify the artifact is the one your pipeline built, unaltered.

CHECK

```
release artifacts signed (Sigstore/cosign or registry provenance);
verification documented
```

REMEDIATION

Sign in CI; publish provenance; document verification.

Tooling: cosign / npm provenance / SLSA

§7 Build Pipeline / CI Hardening

AG-CI-01 Pin third-party Actions/steps to a full commit SHA HIGH Silver

MUST · L0 L1 L2 · static

A mutable tag (@v4) lets the action's owner — or whoever compromises them — change the code you run with repo write access. A 40-hex commit SHA is immutable.

CHECK

```
every `uses:` referencing a third party pins a full 40-char
commit SHA (not a tag or branch)
```

REMEDIATION

Replace tags with the resolved SHA; let a bot bump SHAs under review.

Tooling: zizmor / actionlint / ripgrep 'uses:.@(?![0-9a-f]{40})'*

AG-CI-02 Least-privilege workflow tokens HIGH Silver

MUST · L0 L1 L2 · static

The default workflow token is often write-all; a single compromised step inherits it.

CHECK

```
top-level permissions: defaults to read ; write scoped to the
specific job that needs it
```

REMEDIATION

Add an explicit minimal permissions: block at workflow and job level.

Tooling: actionlint / policy check

AG-CI-03 No untrusted code in a privileged context CRITICAL Silver

MUST · L0 L1 L2 · static

A pull_request_target job (which carries secrets) that checks out and builds/executes the PR head runs attacker-supplied code with your credentials — a classic CI takeover.

CHECK

```
no pull_request_target job checks out head.ref AND runs
build/exec/install with access to secrets
```

REMEDIATION

Use pull_request (no secrets) for untrusted code, or split so the privileged job never executes PR code.

Tooling: zizmor / workflow analysis

AG-CI-04 Secrets are not exposed to forks or echoed HIGH Silver

MUST · L0 L1 L2 · static

Secrets reaching fork PRs, or printed to logs, leak to anyone who can open a PR or read a build.

CHECK

```
no secret passed to fork-triggered jobs ; no echo/printenv of
${{ secrets.* }} ; masking on
```

REMEDIATION

Gate secret-using jobs to trusted events; never print secrets; treat masking as defence-in-depth, not the control.

Tooling: workflow analysis + log scan

AG-CI-05 Protected branch with the conformance gate required MEDIUM Silver

SHOULD · L0 L1 L2 · manual

A standard that isn't a required status check is a suggestion. Make AGSSH a merge gate.

CHECK

```
default branch protected ; AGSSH conformance check required ;
no force-push ; review required
```

REMEDIATION

Enable branch protection and mark the gate a required status check.

Tooling: repo / org policy

AG-CI-06 Automated contributors run under scoped identity and the same gate MEDIUM Gold

SHOULD · L0 L1 L2 · manual

Bots and agents that open PRs across many repos are high-value targets and high-blast-radius writers; their commits deserve at least the scrutiny human ones get.

CHECK

```
automation uses a scoped token (least repos/permissions), signs
commits, and its PRs pass the AGSSH gate before merge
```

REMEDIATION

Scope the automation token; enable commit signing; route automated PRs through the same required gate.

Tooling: org policy / token scoping

§8 Privacy & Consent

AG-PRV

AG-PRV-01 Zero telemetry on air-gapped surfaces CRITICAL Bronze

MUST · L0 L1 (privacy tools) · dynamic

The correct number of analytics providers on a privacy tool is zero. Anything else contradicts the claim the tool is built on.

CHECK

```
zero analytics signatures AND zero third-party requests
(subsumed by AG-NET-01)
```

REMEDIATION

Remove analytics entirely; derive usage signal from edge logs you already control if you must.

Tooling: chromedp request capture

AG-PRV-02 Prior blocking of non-essential third parties CRITICAL Bronze

MUST · L2 (and anywhere a tracker exists) · dynamic

No analytics / ads / embeds / fonts may fire before explicit opt-in. EU law requires PRIOR blocking; exporting IPs to US analytics is a transfer-law exposure. Loading the tracker on page load and letting the visitor decline afterwards is the exact anti-pattern — the cookie is already set and the hit already sent.

CHECK

```
first visit, NO consent cookie => zero requests to analytics/ad
hosts AND zero non-essential cookies set
```

REMEDIATION

Consent Mode with analytics_storage/ad_storage = denied by default, granted only after explicit consent; wire the banner's accept/decline to actually toggle loading. Or drop third-party analytics for an EU-hosted, cookieless one.

Tooling: chromedp (no-consent first-visit capture)

AG-PRV-03 Self-host fonts HIGH Bronze

MUST · L0 L1 L2 · static

Dynamically embedding Google Fonts has been ruled a privacy violation: it ships the visitor's IP to a third party on every load.

CHECK

```
zero fonts.googleapis.com / fonts.gstatic.com references
```

REMEDIATION

Self-host WOFF2 files in @font-face with font-src 'self'.

Tooling: HTML/CSS scan

AG-PRV-04 Minimize storage & fingerprinting MEDIUM Silver

MUST · L0 L1 (SHOULD · L2) · dynamic

Storage and fingerprinting beyond functional need quietly erode the posture and the policy you publish.

CHECK

```
enumerate cookies + localStorage on load against an allowlist ;
nothing non-functional present
```

REMEDIATION

Keep only what the tool needs; disclose any remaining storage.

Tooling: chromedp storage inspection

AG-PRV-05 Cookie attributes where cookies exist HIGH Silver

MUST · when any cookie is set · dynamic

A cookie without Secure/HttpOnly/SameSite is interceptable, script-readable and CSRF-usable.

CHECK

```
every cookie => Secure ; HttpOnly unless a script must read it ;
SameSite=Lax|Strict ; __Host- prefix where applicable
```

REMEDIATION

Set the full attribute set; prefer __Host- prefixed cookies.

Tooling: chromedp cookie inspection

AG-PRV-06 **Third-party embeds isolated and consent-gated** MEDIUM Silver

MUST · L2 (forbidden · L0 L1) · dynamic

An embedded video/map/widget sets third-party cookies and runs third-party code; on a privacy surface it is egress and tracking.

CHECK

```
L0/L1 => zero third-party iframes ; L2 => embeds sandboxed,
consent-gated, privacy-mode variant or click-to-load facade
```

REMEDIATION

Use no-cookie/privacy embeds behind a facade and consent; sandbox the iframe.

Tooling: chromedp + DOM scan

§9 Output / Document Hygiene

AG-OUT

AG-OUT-01 **Neutralize output metadata** MEDIUM Silver

MUST · file-generating tools · dynamic

Generated files leak their toolchain: a PDF's Info dict AND its XMP packet carry Producer/Creator/timestamps (jsPDF embeds these by default); images carry EXIF/GPS/ICC/maker-notes.

CHECK

```
sample output => Info + XMP neutralized, creation timestamp
neutral ; image EXIF/GPS/ICC stripped
```

REMEDIATION

Set neutral document properties and clear XMP; strip image metadata on processing.

Tooling: exiftool / PDF metadata read

AG-OUT-02 **Deterministic output** LOW Gold

SHOULD · file-generating tools · dynamic

Identical inputs yielding byte-identical outputs is both anti-fingerprinting and a reproducibility property.

CHECK

```
two runs on the same input => identical bytes (modulo intended content)
```

REMEDIATION

Zero or fix timestamps and any other non-deterministic fields.

Tooling: sha256 diff of two runs

AG-OUT-03 **In-browser only, no server round-trip** CRITICAL Bronze

MUST · file-generating tools · dynamic

The 'nothing is uploaded' claim must hold at the data-flow level: output is produced and delivered locally.

CHECK

```
output delivered via Blob/object-URL ; no POST of user content
in the processing path (subsumed by AG-NET-01/04)
```

REMEDIATION

Generate and download client-side; remove any upload endpoint from the flow.

Tooling: chromedp request capture

§10 Governance & Conformance Integrity

AG-GOV-01 Waivers can never cover a MUST CRITICAL Bronze

MUST · all profiles · static

Conformance integrity needs a hard floor. Every CRITICAL/HIGH MUST is non-waivable by construction; the deviation mechanism exists only for SHOULD items, keeping the load-bearing controls outside any human-justification path.

CHECK

```
no entry in deviations[] references a rule whose obligation is MUST
=> any such entry FAILS the gate
```

REMEDIATION

Fix the MUST; never waive it. Re-scope to a lower profile only if the rule genuinely does not apply to the surface.

Tooling: manifest cross-check

AG-GOV-02 Waivers auto-expire; the runner enforces it HIGH Silver

MUST · when deviations exist · static

A waiver without enforced expiry becomes permanent debt. The runner compares expires against now; an expired or missing expiry re-enforces the rule immediately, so a waiver can only ever buy bounded time — a fake far-future date still decays on renewal.

CHECK

```
every deviation has expires <= now + max_window ; expired/absent
expiry => the rule is re-enforced (fail)
```

REMEDIATION

Set a real, short expiry; renew under review or remediate.

Tooling: runner clock check

AG-GOV-03 Deviation budget / debt ceiling HIGH Silver

MUST · L0 L1 L2 · static

Truthfulness cannot be machine-judged, but volume can be capped. A per-surface ceiling on the count and weight of active deviations makes waiving-to-green structurally impossible past the budget.

CHECK

```
sum(active deviation weights) <= budget.max_weight AND
count <= budget.max_count   => else FAIL
```

REMEDIATION

Pay down debt before adding more; the budget forces remediation over accumulation.

Tooling: runner budget check

AG-GOV-04 Segregation of duties on waivers HIGH Gold

MUST · when deviations exist (Gold) · static

A self-approved waiver is no control. Each deviation must be signed by an authorized approver distinct from the change author — turning a waiver into an attributable, accountable act rather than anonymous YAML.

CHECK

```
each deviation carries approver != author AND a signature
verifiable against an approver allowlist
```

REMEDIATION

Route waivers through a named approver; sign them; reject self-approval.

Tooling: cosign / signed-off-by + approver allowlist

AG-GOV-05 Conformance record is externally derived and signed HIGH Gold

MUST · L0 L1 L2 (Gold) · dynamic

A record produced inside a compromised build host can be forged. The authoritative record is produced by an independent runner observing the LIVE deployed surface — headers, egress, DNS and TLS are externally observable — and is signed and bound to the artifact digest, so build-host compromise cannot fabricate production reality.

CHECK

```
record generated against the live URL by an out-of-band runner ;
signed (in-toto / cosign) ; bound to artifact digest ; signature verifies
```

REMEDIATION

Run the gate against production from a separate trusted runner; sign the attestation; verify on consumption.

Tooling: external runner + cosign / in-toto attestation

AG-GOV-06 Active egress canary - prove the wall, do not observe its silence HIGH Silver

MUST · L0 L1 · dynamic

Observing 'no egress during a test' proves a negative and is defeated by env-detecting, time-delayed or interaction-gated exfiltration. Instead the harness injects a known exfil probe and asserts the policy BLOCKS it — a positive test of a capability boundary that holds in production for every visitor regardless of attacker timing. CSP is the boundary; the canary proves it is real.

CHECK

```
synthetic beacon to a non-allowlisted origin (fetch/img/beacon/ws)
is blocked by CSP ; a violation is recorded ; production same-origin
reporting is collecting
```

REMEDIATION

Lock every fetch-directive to self (AG-NET-01); add the blocked-probe test; collect violation reports same-origin in production.

Tooling: chromedp + injected probe + CSP report endpoint

AG-GOV-07 Build environment is ephemeral and hermetic HIGH Gold

MUST · L0 L1 L2 (Gold) · static

A persistent build agent is a persistent foothold that can tamper outputs after the gate. A fresh isolated runner per build, a base image pinned by digest, and short-lived OIDC credentials shrink the window in which the host can forge results — and pair with external verification (AG-GOV-05) so tampering is contradicted by the production scan.

CHECK

```
build runs on a fresh ephemeral runner ; base image pinned by
digest ; credentials are short-lived OIDC (no long-lived secrets on agent)
```

REMEDIATION

Use ephemeral CI runners, digest-pinned images and OIDC federation instead of stored secrets.

Tooling: CI config review + provenance

§11 Hosting-Backend Conformance Matrix

AG-HOST

The same rule is enforceable in different ways — or not at all — depending on whether the edge can set response headers. GitHub Pages is the constrained case: **no custom response headers**, so CSP exists only as `<meta>` (with the AG-CSP-03 limitation) and real HSTS/XFO/nosniff need a CDN in front. Everything below can enforce the full header set.

Backend (edge)	Custom HTTP headers?	CSP	HSTS / XFO / nosniff	Canonical mechanism
GitHub Pages (Fastly)	**No**	meta only	not settable as headers	front with a CDN to gain headers; else meta-CSP + stateless surface
Cloudflare Pages	**Yes**	header	full	`_headers` file in the build output
Cloudflare (proxied origin)	**Yes**	header	full	Transform Rules + Managed Transforms; HSTS via SSL/TLS → Edge Certificates
Fastly (VCL / Compute)	**Yes**	header	full	`set resp.http.*` in vcl_deliver, or a Compute service
Netlify	**Yes**	header	full	`_headers` or `netlify.toml`
AWS S3 + CloudFront	**Yes**	header	full	CloudFront Response Headers Policy / Functions / Lambda@Edge

The DNS family (§5) is independent of the edge: CAA, DNSSEC and dangling-record checks apply at the DNS provider whatever serves the bytes. Mixed estates are common — a project's tool/docs subdomains on GitHub Pages (DNS-only) while the apex is proxied through Cloudflare; headers are settable on the apex and not on the Pages subdomains, and decommissioned Pages subdomains are the prime subdomain-takeover risk (AG-DNS-03). Apply fixes per surface, not per project. For clickjacking specifically, a header-less host reaches the anti-framing property structurally: a stateless surface (Tier D) or a frame-buster (Tier C) satisfies it, so a CDN is not required for conformance — adding one to gain headers is a recorded trust-chain expansion (a new TLS-terminating third party), weighed as such.

§12 The Deterministic Labyrinth — Verification Harness

AG-VER

Operationalize the standard as a fleet-level conformance runner. Each repo declares a **target profile** and its surfaces; the runner evaluates only that profile's applicable rules **against the live deployed surface**, emits a weighted score and a severity-ordered fix queue, signs a JSON evidence record bound to the artifact digest (AG-GOV-05), and is idempotent on `(surface.url, rule.id)`. Waivers pass through the governance of §10 — never covering a MUST, auto-expiring, and within the per-surface budget.

Per-repo declaration

```
# .airgap.yml  — one per repo. The fleet runner reads this to know what
# to check and at which strictness. The signed JSON output (AG-GOV-05) is
# the conformance record (evidence) and is retained per build.
target_profile: Bronze          # Bronze | Silver | Gold (cumulative)
level: L0                       # L0 strict | L1 scoped-egress | L2 marketing
surfaces:
  - url: https://tools.example.org/
    kind: client-tool           # client-tool | docs | site
    generates_files: true
    threaded_wasm: false
    has_service_worker: false
    stateless: true             # Tier-D: satisfies anti-framing on header-less hosts
backend: github-pages           # drives the host-capability matrix (§11)
dns:
  zone: example.org             # checked for CAA / DNSSEC / dangling CNAME
pipeline:
  enforce_ci_rules: true        # apply the AG-CI family to this repo's workflows
allow:
  connect: []                   # L1 only: explicit origins, never '*'
  storage: []                   # functional cookies / localStorage keys
  embeds:  []                   # L2 only: sanctioned third-party embed hosts
waivers:                        # governance for SHOULD deviations (§10)
  budget: { max_count: 3, max_weight: 6, max_window_days: 30 }
deviations: []                  # SHOULD-only; MUSTs are never waivable (AG-GOV-01)
```

```
  # - { rule: AG-SUP-07, reason: "SBOM rollout in progress",
  #     approver: alice, expires: 2026-08-01,        # != author; runner-enforced
  #     evidence: https://github.com/org/repo/issues/123 }
```

Canonical header block (header-capable backends)

```
# Cloudflare Pages / Netlify  ->  _headers   (header-capable backends)
/*
  Content-Security-Policy: default-src 'none'; script-src 'self'; style-src 'self'; img-src 'self'
data:; font-src 'self' data:; connect-src 'none'; worker-src 'self'; frame-src 'none'; child-src
'none'; manifest-src 'self'; form-action 'none'; frame-ancestors 'none'; base-uri 'none'; object-
src 'none'; require-trusted-types-for 'script'; trusted-types 'none'; upgrade-insecure-requests
  Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
  X-Frame-Options: DENY
  X-Content-Type-Options: nosniff
  Referrer-Policy: no-referrer
  Permissions-Policy: camera=(), microphone=(), geolocation=(), payment=(), usb=(),
accelerometer=(), gyroscope=(), magnetometer=(), interest-cohort=()
  Cross-Origin-Opener-Policy: same-origin
  Cross-Origin-Resource-Policy: same-origin

# threaded-WASM (SharedArrayBuffer) tools also set, on HTML + assets:
#   Cross-Origin-Embedder-Policy: require-corp
# if you create a Trusted Types policy (e.g. a sanitizer factory), name it:
#   ... require-trusted-types-for 'script'; trusted-types dompurify
# optional violation reporting MUST stay same-origin (AG-NET-08):
#   Reporting-Endpoints: csp="/_report"   + CSP directive  report-to csp
# Gold residual-channel hardening (the egress canary AG-GOV-06 probes these):
#   add  prefetch-src 'none'  and avoid <link rel=dns-prefetch>; WebRTC is not
#   fully gated by CSP, so disable it if the surface does not use it.
# L1 replaces connect-src 'none' with the exact origin(s). L2 loosens
#   script/style/img to what the page uses, keeps trackers behind consent,
#   and never weakens frame-ancestors / object-src / base-uri.
```

Conformance gate

```
# conformance gate (pseudo) — profile-scoped, prod-observed, signed
def gate(surface):
    rules = [r for r in MANIFEST
             if r.in_profile(surface.target_profile)         # Bronze/Silver/Gold
             and r.applies_at(surface.level, surface)]
    results = [run_external(r, surface.live_url) for r in rules]   # black-box, prod
    record  = sign(emit_json(results), artifact_digest)           # AG-GOV-05
    # waiver governance — AG-GOV-01..04
    valid = [w for w in surface.deviations
             if not w.rule.is_must                 # never waive a MUST (AG-GOV-01)
             and w.expires > now                   # auto-expire        (AG-GOV-02)
             and w.signed_by_approver(w)]          # SoD, Gold          (AG-GOV-04)
    assert within_budget(valid, surface.waivers.budget)           # AG-GOV-03
    must_fail = [r for r in results if r.is_must and not r.ok
                 and r.id not in {w.rule for w in valid}]
    ci_fail(must_fail) if must_fail else warn(unwaived_should_fails(results))
# idempotency key = (surface.url, rule.id). Run as a REQUIRED check on the
# protected branch (AG-CI-05), including on automated contributors' PRs (AG-CI-06).
```

60-second drop-in gate

```
# 60-second drop-in gate (top CRITICAL/HIGH on a live surface + its repo)
U=https://YOUR.SURFACE/ ; Z=YOUR.ZONE
# headers
curl -sI "$U" | grep -iE 'content-security|strict-transport|x-frame|x-content-type|referrer-
policy|permissions-policy|cross-origin'
```

```
# third-party egress / trackers / default-CDN loaders
curl -s "$U" | grep -oiE 'googletagmanager|gtag|G-[A-Z0-9]{8,}|fonts\.g(oogleapis|static)|
jsdelivr|unpkg|cdnjs|tessdata'
# meta-CSP frame-ancestors trap (ineffective)
curl -s "$U" | grep -oiE 'http-equiv="?content-security[^>]*frame-ancestors'
# DNS trust anchors
dig +short CAA "$Z" ; dig +dnssec +short "$Z" | grep -qi ad && echo "DNSSEC: AD"
# TLS floor
testssl.sh --quiet --protocols "$U"
# supply chain + pipeline (in the repo)
osv-scanner --lockfile=package-lock.json ; gitleaks detect --source dist/ --no-banner
grep -rEn 'uses:[^@]+@(?![0-9a-f]{40})' .github/workflows/    # unpinned Actions
# active egress canary (AG-GOV-06): the proof is a BLOCKED probe, not silence
#   in CI: page injects fetch('https://canary.invalid') and asserts CSP blocks it
# expect: HSTS max-age>=31536000 ; NO analytics/CDN/font hosts ; NO frame-ancestors
#   in a meta CSP ; CAA present ; modern TLS only ; 0 high-CVE deps ; 0 secrets ;
#   0 tag-pinned third-party Actions ; canary probe blocked
```

Appendix A Rule Index

ID	Rule	Severity	Profile	Obligation
AG-NET-01	Self-host every runtime dependency	CRITICAL	Bronze	MUST
AG-NET-02	Override library default CDN loaders	CRITICAL	Bronze	MUST
AG-NET-03	Same-origin workers; constrain worker-src	HIGH	Silver	MUST
AG-NET-04	Constrain connect-src	CRITICAL	Bronze	MUST
AG-NET-05	No third-party egress during load	HIGH	Silver	MUST
AG-NET-06	No third-party connection priming	MEDIUM	Silver	MUST
AG-NET-07	Service Worker hygiene	HIGH	Silver	MUST
AG-NET-08	Violation/error reporting stays same-origin	MEDIUM	Silver	MUST
AG-NET-09	External _blank links sever the opener; privacy surfaces sever the referrer	MEDIUM	Bronze	MUST
AG-CSP-01	Ship a CSP	HIGH	Bronze	MUST
AG-CSP-02	Deny by default; enumerate; forbid bypass primitives	HIGH	Bronze	MUST
AG-CSP-03	Clickjacking control must be header-delivered	HIGH	Bronze	MUST
AG-CSP-04	Stage report-only before enforcing	MEDIUM	Silver	SHOULD
AG-CSP-05	Force HTTPS subresources	MEDIUM	Silver	MUST
AG-CSP-06	Trusted Types for DOM sinks	HIGH	Silver	MUST
AG-HDR-01	HSTS with a real max-age	HIGH	Bronze	MUST
AG-HDR-01a	HSTS scope is sticky — gate it	HIGH	Silver	MUST
AG-HDR-02	Disable MIME sniffing	HIGH	Bronze	MUST
AG-HDR-03	X-Frame-Options	HIGH	Bronze	MUST
AG-HDR-04	Referrer-Policy	MEDIUM	Silver	MUST
AG-HDR-05	Permissions-Policy	MEDIUM	Silver	MUST

ID	Rule	Severity	Profile	Obligation
AG-HDR-06	Cross-origin isolation for threaded WASM	MEDIUM	Gold	MUST
AG-HDR-07	Cache discipline for fingerprinted assets	LOW	Silver	SHOULD
AG-HDR-08	TLS floor	HIGH	Silver	MUST
AG-DNS-01	CAA pins certificate issuance	HIGH	Silver	MUST
AG-DNS-02	DNSSEC enabled	MEDIUM	Silver	SHOULD
AG-DNS-03	No danglable records (subdomain takeover)	HIGH	Silver	MUST
AG-SUP-01	SRI on any unavoidable cross-origin subresource	HIGH	Silver	MUST
AG-SUP-02	Pin dependencies by integrity hash	HIGH	Silver	MUST
AG-SUP-03	Reproducible build; pinned toolchain	MEDIUM	Silver	SHOULD
AG-SUP-04	No secrets in shipped output	HIGH	Bronze	MUST
AG-SUP-05	No unintended source maps	LOW	Silver	SHOULD
AG-SUP-06	No known-vulnerable dependencies	HIGH	Bronze	MUST
AG-SUP-07	SBOM emitted and retained	MEDIUM	Gold	SHOULD
AG-SUP-08	Signed releases / build provenance	MEDIUM	Gold	SHOULD
AG-CI-01	Pin third-party Actions/steps to a full commit SHA	HIGH	Silver	MUST
AG-CI-02	Least-privilege workflow tokens	HIGH	Silver	MUST
AG-CI-03	No untrusted code in a privileged context	CRITICAL	Silver	MUST
AG-CI-04	Secrets are not exposed to forks or echoed	HIGH	Silver	MUST
AG-CI-05	Protected branch with the conformance gate required	MEDIUM	Silver	SHOULD
AG-CI-06	Automated contributors run under scoped identity and the same gate	MEDIUM	Gold	SHOULD
AG-PRV-01	Zero telemetry on air-gapped surfaces	CRITICAL	Bronze	MUST
AG-PRV-02	Prior blocking of non-essential third parties	CRITICAL	Bronze	MUST
AG-PRV-03	Self-host fonts	HIGH	Bronze	MUST
AG-PRV-04	Minimize storage & fingerprinting	MEDIUM	Silver	MUST
AG-PRV-05	Cookie attributes where cookies exist	HIGH	Silver	MUST
AG-PRV-06	Third-party embeds isolated and consent-gated	MEDIUM	Silver	MUST
AG-OUT-01	Neutralize output metadata	MEDIUM	Silver	MUST
AG-OUT-02	Deterministic output	LOW	Gold	SHOULD
AG-OUT-03	In-browser only, no server round-trip	CRITICAL	Bronze	MUST
AG-GOV-01	Waivers can never cover a MUST	CRITICAL	Bronze	MUST
AG-GOV-02	Waivers auto-expire; the runner enforces it	HIGH	Silver	MUST
AG-GOV-03	Deviation budget / debt ceiling	HIGH	Silver	MUST

ID	Rule	Severity	Profile	Obligation
AG-GOV-04	Segregation of duties on waivers	HIGH	Gold	MUST
AG-GOV-05	Conformance record is externally derived and signed	HIGH	Gold	MUST
AG-GOV-06	Active egress canary - prove the wall, do not observe its silence	HIGH	Silver	MUST
AG-GOV-07	Build environment is ephemeral and hermetic	HIGH	Gold	MUST

Appendix B Machine-Readable Manifest

Emitted from the same rule source as this document — spec text, tooling and manifest cannot drift. Feed it to the gate in §12.

```
rules:
  - id: AG-NET-01
    severity: CRITICAL
    check: static+dynamic
    tool: chromedp request capture + CSP parser
  - id: AG-NET-02
    severity: CRITICAL
    check: static
    tool: ripgrep over dist/
  - id: AG-NET-03
    severity: HIGH
    check: static
    tool: AST/regex scan + CSP parser
  - id: AG-NET-04
    severity: CRITICAL
    check: static
    tool: CSP parser
  - id: AG-NET-05
    severity: HIGH
    check: dynamic
    tool: chromedp request/response capture
  - id: AG-NET-06
    severity: MEDIUM
    check: static
    tool: HTML parser
  - id: AG-NET-07
    severity: HIGH
    check: dynamic
    tool: chromedp SW inspection
  - id: AG-NET-08
    severity: MEDIUM
    check: static
    tool: header + CSP parser
  - id: AG-NET-09
    severity: MEDIUM
    check: static
    tool: HTML anchor scan
  - id: AG-CSP-01
    severity: HIGH
    check: static
    tool: header + HTML parser
  - id: AG-CSP-02
    severity: HIGH
    check: static
    tool: CSP directive-predicate parser
  - id: AG-CSP-03
    severity: HIGH
    check: static
    tool: header + CSP parser
```

```
  - id: AG-CSP-04
    severity: MEDIUM
    check: manual
    tool: staging header + report log
  - id: AG-CSP-05
    severity: MEDIUM
    check: static
    tool: CSP parser + HTML scan
  - id: AG-CSP-06
    severity: HIGH
    check: static+dynamic
    tool: CSP parser + runtime violation check
  - id: AG-HDR-01
    severity: HIGH
    check: static
    tool: header parser
  - id: AG-HDR-01a
    severity: HIGH
    check: manual
    tool: subdomain enumeration + curl
  - id: AG-HDR-02
    severity: HIGH
    check: static
    tool: header parser
  - id: AG-HDR-03
    severity: HIGH
    check: static
    tool: header parser
  - id: AG-HDR-04
    severity: MEDIUM
    check: static
    tool: header parser
  - id: AG-HDR-05
    severity: MEDIUM
    check: static
    tool: header parser
  - id: AG-HDR-06
    severity: MEDIUM
    check: conditional
    tool: header parser + runtime crossOriginIsolated check
  - id: AG-HDR-07
    severity: LOW
    check: static
    tool: header parser
  - id: AG-HDR-08
    severity: HIGH
    check: dynamic
    tool: testssl.sh / sslyze
  - id: AG-DNS-01
    severity: HIGH
    check: static
    tool: dig CAA / DNS API
  - id: AG-DNS-02
    severity: MEDIUM
    check: static
    tool: dig +dnssec / delv
  - id: AG-DNS-03
    severity: HIGH
    check: static
    tool: dnsReaper / subjack
  - id: AG-SUP-01
    severity: HIGH
    check: static
    tool: HTML parser
  - id: AG-SUP-02
```

```
    severity: HIGH
    check: static
    tool: lockfile lint + ripgrep
  - id: AG-SUP-03
    severity: MEDIUM
    check: static
    tool: build-config lint
  - id: AG-SUP-04
    severity: HIGH
    check: static
    tool: gitleaks / trufflehog over dist/
  - id: AG-SUP-05
    severity: LOW
    check: static
    tool: ripgrep over dist/
  - id: AG-SUP-06
    severity: HIGH
    check: static
    tool: osv-scanner / trivy fs / npm audit --audit-level=high
  - id: AG-SUP-07
    severity: MEDIUM
    check: static
    tool: syft / cdxgen
  - id: AG-SUP-08
    severity: MEDIUM
    check: static
    tool: cosign / npm provenance / SLSA
  - id: AG-CI-01
    severity: HIGH
    check: static
    tool: zizmor / actionlint / ripgrep 'uses:.*@(?![0-9a-f]{40})'
  - id: AG-CI-02
    severity: HIGH
    check: static
    tool: actionlint / policy check
  - id: AG-CI-03
    severity: CRITICAL
    check: static
    tool: zizmor / workflow analysis
  - id: AG-CI-04
    severity: HIGH
    check: static
    tool: workflow analysis + log scan
  - id: AG-CI-05
    severity: MEDIUM
    check: manual
    tool: repo / org policy
  - id: AG-CI-06
    severity: MEDIUM
    check: manual
    tool: org policy / token scoping
  - id: AG-PRV-01
    severity: CRITICAL
    check: dynamic
    tool: chromedp request capture
  - id: AG-PRV-02
    severity: CRITICAL
    check: dynamic
    tool: chromedp (no-consent first-visit capture)
  - id: AG-PRV-03
    severity: HIGH
    check: static
    tool: HTML/CSS scan
  - id: AG-PRV-04
    severity: MEDIUM
```

```
    check: dynamic
    tool: chromedp storage inspection
  - id: AG-PRV-05
    severity: HIGH
    check: dynamic
    tool: chromedp cookie inspection
  - id: AG-PRV-06
    severity: MEDIUM
    check: dynamic
    tool: chromedp + DOM scan
  - id: AG-OUT-01
    severity: MEDIUM
    check: dynamic
    tool: exiftool / PDF metadata read
  - id: AG-OUT-02
    severity: LOW
    check: dynamic
    tool: sha256 diff of two runs
  - id: AG-OUT-03
    severity: CRITICAL
    check: dynamic
    tool: chromedp request capture
  - id: AG-GOV-01
    severity: CRITICAL
    check: static
    tool: manifest cross-check
  - id: AG-GOV-02
    severity: HIGH
    check: static
    tool: runner clock check
  - id: AG-GOV-03
    severity: HIGH
    check: static
    tool: runner budget check
  - id: AG-GOV-04
    severity: HIGH
    check: static
    tool: cosign / signed-off-by + approver allowlist
  - id: AG-GOV-05
    severity: HIGH
    check: dynamic
    tool: external runner + cosign / in-toto attestation
  - id: AG-GOV-06
    severity: HIGH
    check: dynamic
    tool: chromedp + injected probe + CSP report endpoint
  - id: AG-GOV-07
    severity: HIGH
    check: static
    tool: CI config review + provenance
```

Appendix C Severity → Gating Policy

Severity	Weight	Build effect on failure
CRITICAL	8	block
HIGH	4	block
MEDIUM	2	advisory — record a deviation to waive
LOW	1	advisory — record a deviation to waive

Appendix D Tooling Reference

by check family

Each check family maps to a deterministic, scriptable open-source tool so the standard can be implemented as automation rather than inspection.

Check family	Rules	Recommended tool(s)
Response headers	AG-HDR-01..05,07 · AG-NET-08	`curl -I` + a header parser
CSP directive predicates	AG-CSP-01..03,05,06 · AG-NET-03,04	a deterministic CSP parser
Runtime egress / offline / SW / cookies / embeds / output	AG-NET-01,05,07 · AG-PRV-01,02,04,05,06 · AG-OUT-01..03	chromedp (headless Chromium; request & storage capture)
Static bundle inspection	AG-NET-02 · AG-SUP-02,05	ripgrep over `dist/`
Secrets in build output	AG-SUP-04	gitleaks / trufflehog
Vulnerable dependencies	AG-SUP-06	osv-scanner / trivy / npm audit
SBOM	AG-SUP-07	syft / cdxgen
Signing & provenance	AG-SUP-08	cosign / SLSA generators
CI workflow hardening	AG-CI-01..04	zizmor / actionlint
DNS trust & takeover	AG-DNS-01..03	dig / dnsReaper / subjack
TLS floor	AG-HDR-08	testssl.sh / sslyze
Output metadata	AG-OUT-01	exiftool / a PDF metadata reader

Appendix E Informative References

Pointers only; consult primary sources. Legal items describe well-known positions and are not legal advice.

Area	Reference
Keyword interpretation	RFC 2119; RFC 8174
HSTS	RFC 6797; HSTS preload inclusion criteria (max-age ≥ 1 year, includeSubDomains, preload)
Disclosure / well-known	RFC 9116 (security.txt); RFC 8615 (well-known URIs)
CSP	W3C CSP Level 3 — incl. that frame-ancestors, report-uri/report-to and sandbox are ignored when delivered via <meta>
DOM-XSS hardening	W3C Trusted Types
Cross-origin isolation	COOP, COEP and the SharedArrayBuffer prerequisite
Supply chain	SLSA framework; CycloneDX / SPDX (SBOM); OSV; OpenSSF Scorecard
Prior consent (EU)	ePrivacy Directive 2002/58/EC, Art. 5(3)
Data transfers (EU)	CJEU C-311/18 (“Schrems II”, 2020)
Analytics transfers	Italian Garante — 2022 decision on unlawful Google Analytics transfers
Embedded fonts	LG München I — 2022 ruling on Google Fonts embedding and IP transfer

Appendix F Trust Dependencies & Non-Goals (Out of Scope to Specify)

The following are out of scope to *specify* here — they belong to adjacent standards — but where the threat model depends on them they are treated as **trust dependencies with compensating controls**, not as assumptions:

Build-host / runner integrity — not assumed hardened. Defended by ephemeral hermetic builds (AG-GOV-07) and by deriving the authoritative conformance record externally from the live surface and signing it (AG-GOV-05), so a compromised build host cannot forge production reality. Its full hardening specification belongs to a separate standard.
Server-side / application security of any dynamic backend (authz logic, injection in server code)
Email-domain anti-abuse (SPF / DKIM / DMARC)
Account, session and identity security beyond the static surface
Accessibility (WCAG) and SEO — important, but a separate concern