CSURFACE

WHITEPAPER · DIGITAL SUPPLY CHAIN

Digital supply chain risk · inherited third-party exposure

Dozens of third parties run code inside your application — with no signal when one of them changes hands, is compromised or expires. Mapping the digital supply chain three levels deep turns that blind spot into a verifiable inventory.

CAPABILITY	AUDIENCE	EDITION
Digital Supply Chain	CISO · Risk · AppSec	2026

READING TIME	CLASSIFICATION	SOURCE
16 pages	Public	CSURFACE Platform

EXECUTIVE SUMMARY

Your application runs code that is not its own

The thesis of this whitepaper, in one page — what executives need to know before the technical detail.

SITES AFFECTED BY POLYFILL.IO

100k+

domains served the malicious script without knowing

VULNERABILITIES COMING FROM DEPENDENCIES

80%

in modern web apps, rather than from proprietary code

DEPTH OF CSURFACE MAPPING

3 levels

from the direct dependency to the subvendor of the subvendor

In June 2024, the domain **polyfill.io** — used by more than 100,000 sites to distribute JavaScript scripts — began injecting malicious code after a change of ownership. Most affected organizations were unaware of this dependency. There was no direct contractual link: Polyfill.io was a subvendor of a subvendor, integrated into the code years earlier by teams that were no longer with the organization.

The case exemplifies the norm. Modern applications execute, in real time, code belonging to dozens of third parties — **CDNs, analytics scripts, SaaS APIs, libraries with transitive dependencies**. Each of these elements is a door. And most of them appear in no vendor inventory.

This whitepaper analyzes the Polyfill.io case as an archetypal example, walks through **other public incidents** detectable externally (MageCart at British Airways, Ticketmaster via Inbenta, Sea Turtle, subdomain takeover), constructs **two critical scenarios** — session cookie theft via subdomain takeover and *second-grade takeover* of an expired domain — and presents the CSURFACE approach: the technical and continuous mapping of the digital supply chain up to **three levels deep**, with practical diagrams for web and DNS and readings dedicated to the manager, the engineer and the analyst.

The thesis. Every application executes third-party code. Each library, each CDN and each integrated API represents an inherited dependency — subject to change of ownership, compromise or expiration without notice. What is not continuously mapped can be neither managed nor defended.

What this whitepaper covers

- The Polyfill.io case and three public incidents of the same pattern.
- The technical fronts — scripts, DNS, headers and cookies.
- Two critical scenarios: cookie + subdomain takeover; second-grade takeover.
- The CSURFACE mapping across three levels and the reading by role.

THE POLYFILL.IO CASE · PART 01

A domain changed hands — and 100k sites did not notice

The anatomy of a public digital supply chain incident, from the ignored alert to the malicious code.

Whitepaper 2026
CSURFACE

What happened

Polyfill.io was an open-source CDN that distributed "polyfills" — fragments of JavaScript intended to provide modern functionality in older browsers. The service was free and achieved wide adoption in the web development community.

In early 2024, the domain was acquired by a new operator. The project's original creator, Andrew Betts, **publicly warned** that the domain had changed hands and was no longer trustworthy. The warning went unnoticed — because there was no monitoring.

In June 2024, malicious scripts began to be served from the domain: tracking, redirects to gambling sites and potential credential theft. Cloudflare and Google responded by blocking the domain in their services — but the damage had been accumulating for weeks.

Why the impact was so broad

1. **The dependency was unknown.** On many sites the script had been added years earlier, by someone who was no longer at the company. It was an invisible dependency.
2. **Ownership was not monitored.** There was no process tracking ownership changes of transitive vendors — precisely the event that triggered the attack.
3. **Permissive CSP.** Most sites operated without a restrictive Content-Security-Policy able to block the execution of an altered script.
4. **SCA did not cover public HTML/JS.** Composition analysis tools looked at package-manager dependencies, not scripts loaded at runtime by the browser.

The signal that was missing. The project's creator publicly communicated that the domain had changed ownership. The warning signal was available. The gap was the absence of a mechanism able to monitor ownership changes in transitive vendors and forward that notice to the security teams of the consuming organizations.

The pattern repeats. Polyfill.io illustrates a structural problem that recurs across many digital vendors. Every transitive dependency integrated without continuous monitoring represents an equivalent vector — subject to the same chain of events when control of its ownership changes hands.

CODE THAT IS NOT YOURS · PART 02

The digital supply chain has six fronts — and almost none is mapped

The general problem behind the Polyfill.io case: what third-party dependency in a modern application is made of.

87%
of modern web applications operate with 50 or more direct dependencies

80%
of a web app's vulnerabilities come from dependencies, not from proprietary code

94 days
is the average time to discover compromised credentials in the vendor chain

What the dependency is composed of

Modern applications execute, in real time, code belonging to dozens of third parties — and reference, in DNS, dozens of other external domains. This risk is distributed across **six technical fronts** that an ASM platform detects from the external perimeter, with no agent and no internal instrumentation. Of these six fronts, DNS trust, subdomain takeover, WHOIS, certificates and vendor credentials belong to the platform's full external surface (ASM/Discovery); the Supply Chain module focuses on the chain of embedded components — scripts, APIs and libraries — mapped up to three levels deep.

Each front follows its own pattern of adoption and of failure. Treating them with a single control — questionnaire, SCA or an isolated CSP — is what produces the invisibility that Polyfill.io made public.

The cost of invisibility. Attacks exploiting edge devices and VPNs managed by third parties grew roughly eightfold compared with the previous year. Approximately **30% of 2025 breaches** involved a third party — and the average time to discovery of compromised credentials in the vendor chain is 94 days.

Sources: Snyk State of Open Source 2024; OWASP; Verizon DBIR 2025; IBM Cost of a Data Breach 2025 — public reports.

The six risk categories of the digital supply chain

CRITICAL **1 · Third-party JavaScript in the browser**
Tag managers, analytics scripts, A/B testing, polyfills, pixels and widgets — any code executed in the client's browser. The category of Polyfill.io and the MageCart family.

CRITICAL **2 · DNS trust relationships**
SPF records (and their includes), external MX and NS, CNAMEs pointing to third-party domains, DKIM records — the authority chain of your primary domain.

WARNING **3 · Referenced subdomains and cloud resources**
CNAMEs pointing to cloud services (AWS S3, Azure App Service, Heroku, GitHub Pages, Fastly) — the classic subdomain takeover surface when the origin resource is deallocated.

WARNING **4 · APIs and endpoints called at runtime**
SaaS APIs, webhooks, tracking pixels, payment endpoints, transactional email integrations — every external call the application executes in real time.

WARNING **5 · HTTP headers and protections**
CSP, SRI, CORS, HSTS, cookie flags (Secure, HttpOnly, SameSite) — the technical posture that decides the outcome when one of the third parties above is compromised.

CRITICAL **6 · Domains in a degraded state**
Near expiration, recent WHOIS change, operator transfer, certificate in atypical renewal — early signs of takeover or second-grade in the chain.

OTHER REAL CASES · PART 03 · SCRIPTS AND CDN

Polyfill.io was not the first

Three earlier public incidents that followed the same pattern — a third party embedded in the digital supply chain, compromised without notice to the consuming organization.

The pattern behind the cases. In both incidents that follow, the attack entered through a component the target organization had integrated years earlier, with no subsequent monitoring mechanism. The vector differs — a direct script on the payment page or the compromise of an entire widget vendor — but the structure is the same, and the external detection is equivalent: headless rendering of the page identifies any change in the content delivered by the third party.

British Airways · MageCart (2018)

Between 21 August and 5 September 2018, attackers injected **22 lines of JavaScript** into the payment page of British Airways' website and application. The script captured card number, name, address and CVV in real time and sent them to a server controlled by third parties. The injection came from the digital supply chain: a vendor JavaScript component already integrated into the site was modified to include the skimmer, with no visible change to the internal team.

The attack lasted approximately **15 days** before being detected. About **380,000 transactions** were compromised.

Impact. A **£20 million** fine from the ICO under GDPR (2020) — initially proposed at £183 million. Civil compensation to customers, lasting reputational damage. The regulator identified the absence of technical monitoring of third-party JavaScript as the company's central failing.

Ticketmaster · via Inbenta (2018)

In **June 2018**, Ticketmaster disclosed that payment pages on its sites had been serving a malicious script for approximately nine months. The script was not embedded directly in Ticketmaster's code: it was served by the customer support chat widget from **Inbenta**, a vendor integrated into the payment pages.

Inbenta confirmed a compromise of its infrastructure. The JavaScript it hosted — loaded in real time by the browser from the vendor's legitimate origin — was altered to include a skimmer. Approximately **40,000 customers** had data exposed.

Impact. A **£1.25 million** fine from the ICO. A vector distinct from the BA case: what was compromised was an **entire application vendor**, not Ticketmaster's code. The JavaScript loaded by the browser continued to come from the vendor's legitimate origin — with all the inherited chain trust.

How CSURFACE detects this category

The scripts and CDN category is the most externally visible — and the best covered by **headless rendering**. The platform loads each of the organization's pages as a real browser, records all executed JavaScript of external origin, computes the hash of each file received and compares it against the baseline.

For each identified third-party script, CSURFACE maintains: the origin (domain and CDN), the current WHOIS, the hash of the delivered content and the history of changes. Any change — a domain that changes hands, a hash that changes, a CDN that redirects — generates an alert within **24 hours**.

Result. The two incidents on this page would be detected before material impact — the BA case, through detection of the injection in the payment page's JavaScript; Ticketmaster/Inbenta, through the hash change in the script loaded from the vendor's origin.

OTHER REAL CASES · PART 03 · DNS, MX AND DOMAINS

The digital supply chain also lives in DNS

Subdomains pointing to abandoned resources, compromised DNS providers, orphaned records — a class of risk that appears in no code repository.

Whitepaper 2026
CSURFACE

Where the risk lives in DNS. Modern applications depend on dozens of names — their own subdomains, records pointing to cloud services, external DNS and email providers, SPF includes. Each of these names is itself a node of the digital supply chain. When control of one of them is lost — because the resource was deallocated, the domain expired or the provider was compromised — the trust that pointed to it remains valid, but now serves whoever assumes control.

Subdomain takeover via orphaned CNAME

The most common pattern: an organization's subdomain — for example, **blog.empresa.com.br** — points, via a CNAME record, to a cloud service (Heroku, AWS S3, GitHub Pages, Azure App Service). The service was deallocated at some point, but the CNAME remained. Anyone can then recreate the cloud resource with the same name, and the subdomain begins to serve their content — under the organization's trusted name.

Documented in hundreds of public reports on HackerOne, with cases involving Microsoft, Uber, Shopify, Starbucks and multiple large organizations. Abandoned resources in **Azure, AWS, Heroku, GitHub Pages and Fastly** are the most frequently affected.

Impact. Authenticated phishing under a real domain, malware distribution under valid HTTPS, cookie theft in sibling domains (see hypothetical scenario, page 8). Typically detected only after exploitation by researchers or attackers.

Sea Turtle · upstream DNS hijacking (2017-2019)

A nation-state campaign documented by Cisco Talos. Attackers compromised **DNS providers** in at least 13 countries — chiefly in the Middle East and North Africa — and modified the NS and A records of target organizations: governments, energy companies, intelligence agencies. With DNS hijacked, they redirected traffic to infrastructure under their control and issued valid TLS certificates under the victims' names, enabling credential capture in apparently legitimate sessions.

More than **40 organizations** in 13 countries were identified as direct targets.

Impact. Silent capture of credentials and traffic of strategic organizations; compromise of entire chains of official email; use of valid certificates to evade detection by TLS pinning or manual inspection.

MX and SPF pointing to orphaned domains

A less visible, but common, variation: **MX** records or **SPF** includes of the organization — or of integrated vendors — point to third-party domains that were decommissioned or expired. The record remains published and valid. If the domain is re-registered by an attacker, it becomes an authorized origin to receive email (MX) or to send authenticated email on behalf of the organization (SPF include).

The banking case detailed on **page 9** is the most severe manifestation of this pattern — an SPF include pointing to the expired domain of an email marketing vendor.

Impact. Receipt of email destined for the orphaned subdomain (password recovery, invoices, vendor communications); or delivery of mass phishing authenticated by SPF as if from the legitimate organization — addressed in detail in the next section.

OTHER REAL CASES · PART 03 · HEADERS AND PROTECTIONS

What the application permits matters as much as what it loads

Four header and cookie configurations whose absence or permissiveness determines whether the browser will resist or amplify a supply chain attack.

Whitepaper 2026
CSURFACE

Why this section exists. The three previous pages dealt with what is in the digital supply chain — scripts, dependencies, DNS records. This page deals with the browser protections that decide the outcome when one of those elements is compromised. In almost every public case, a well-configured protection would have significantly reduced the impact or blocked the exploitation.

Permissive or absent Content-Security-Policy

The CSP defines, in the HTTP header, which origins may load scripts, styles and connections. A restrictive policy blocks the execution of scripts from unauthorized origins; a permissive one (with wildcards or `unsafe-inline`) or an absent one leaves the browser to accept any script.

CSP protects against **injection of new origins** — XSS, or a vendor serving from a previously unseen origin. It does **not**, however, protect when the already-authorized origin is itself compromised (the Polyfill.io case). For that scenario, the direct defense is **SRI**, in the block alongside.

Impact. Without a restrictive CSP, any script from any origin is executed. A single compromised component in the digital supply chain obtains full control of the page — reading forms, exfiltrating non-HttpOnly cookies, redirecting.

CORS with wildcard or reflected origin

The `Access-Control-Allow-Origin` header determines which origins may read responses from an authenticated API. Configured with `*` or dynamically reflecting the request's origin (without validation), it allows any external site to make authenticated requests on behalf of the logged-in user — as long as the browser sends cookies — and read the result.

This pattern amplifies supply chain attacks: a script compromised at any origin can read sensitive data from the target organization's API, even when hosted on another domain.

Impact. Leakage of authenticated data — balances, histories, profiles — to any origin that executes code in the user's browser, including compromised origins in the digital supply chain.

Missing Subresource Integrity (SRI)

SRI is an HTML attribute that carries, alongside the `<script src="...">` tag, a cryptographic hash of the expected content. The browser computes the hash of the file received and compares it before executing. If the content of the origin server has changed — because the CDN was altered, compromised or replaced — the browser refuses execution.

Without SRI, the browser accepts whatever content the CDN server returns. A swap like the Polyfill.io one passes silently.

Impact. The alteration of a file on the CDN — deliberately or through vendor compromise — executes with no warning signal at all. SRI is the direct cryptographic defense against the Polyfill.io risk category.

Cookies without protection flags

Session cookies must be configured with three flags: **Secure** (HTTPS only), **HttpOnly** (not accessible via JavaScript) and **SameSite** (Lax or Strict — restricts sending in cross-site requests). The absence of any one of them widens the impact of a compromise in the digital supply chain.

HttpOnly, in particular, is the critical defense against capture via injected JavaScript — a common vector in all the cases on page 5. Its absence turns an XSS or a compromised script into direct account takeover.

Impact. Without HttpOnly, any script in the digital supply chain — legitimate or compromised — can read the session cookie and send it to an attacker-controlled endpoint. It is the central vector of the hypothetical scenario on the next page.

CRITICAL SCENARIOS · PART 04 · SCENARIO 01

Subdomain takeover + cookie issued for the parent domain

How a forgotten subdomain pointing to a deallocated cloud resource turns into account takeover at scale — with no alert, no MFA, no easy trail.

Whitepaper 2026
CSURFACE

The scenario. A financial institution operates **app.banco.com.br** as a critical application, with authentication and MFA. To support single sign-on across internal services, the **session cookie is issued for the parent domain** — `Domain=.banco.com.br`. The organization operates dozens of subdomains; one of them — a promotional page for a campaign that ended in 2022 — points, via CNAME, to a cloud bucket that was deallocated. The DNS record remained published.

The chain of events of the attack

1. **Reconnaissance.** The attacker enumerates the public subdomains of `banco.com.br` and identifies `promo2022.banco.com.br` pointing, via CNAME, to a cloud bucket whose name is not registered. The subdomain is eligible for takeover.
2. **Seizing the resource.** The attacker registers the bucket with the same name in the cloud provider. The existing CNAME now resolves to infrastructure under their control. The subdomain is served under valid HTTPS — the provider issues a certificate for the requested name.
3. **Publishing the payload.** The attacker hosts on the bucket a simple HTML page with a capture script — reading `document.cookie` and sending it to the attacker's endpoint via `fetch()`. Because the victim will visit a subdomain of `banco.com.br`, the browser will attach the cookie issued for `.banco.com.br`.
4. **Luring the victim.** The link `promo2022.banco.com.br` still appears in old marketing emails, archived materials, unrevoked internal redirects and responses indexed by search engines. A mere fraction of victims clicking is enough for the attack to have material scale.
5. **Capture and abuse.** The victim's browser — already authenticated in `app.banco.com.br` in another tab — sends the session cookie to the compromised subdomain. If the cookie lacks **HttpOnly**, the attacker's JavaScript reads it and exfiltrates it. The attacker authenticates in `app.banco.com.br` using the captured cookie — **after the MFA, because the session is already established**.

Why this scenario is particularly severe. The attacker operates *after* the MFA — the captured session is already authenticated. The request originates from the victim's IP or a similar IP, which hinders detection by SIEM. The compromised subdomain has valid HTTPS and the organization's real name. And the vector is silent: no log of the main application shows the capture — it occurs on a subdomain outside monitoring.

The three simultaneous conditions

The scenario only works when three conditions coexist in the organization:

1. An orphaned subdomain pointing to a deallocated cloud resource — available for takeover.

2. A session cookie issued for the parent domain — attached to all subdomains.

3. A cookie without the HttpOnly flag — readable by JavaScript in the browser.

Each condition, in isolation, is a defensible technical decision in some context. The combination of the three is the vulnerability — and no isolated application analysis sees it.

How CSURFACE detects it. Continuous discovery maps all the organization's public subdomains; inspection identifies CNAMEs pointing to deallocated resources (subdomain takeover); headless analysis records the scope and flags of the main application's cookies. The three conditions generate a single critical alert, before the attacker gets there first.

CRITICAL SCENARIOS · PART 04 · SCENARIO 02

Second-grade takeover · when the vendor's domain expires

First-grade takeover affects a subdomain of your own. Second-grade affects an entire domain in the chain — usually a vendor's — that expired and was re-registered. All inherited trust passes, with no alarm, to the new owner.

Whitepaper 2026
CSURFACE

The concept. A third-party domain — an email marketing vendor, a campaign partner, a CDN, a discontinued integrator — expires through failure to renew, cessation of operations, an M&A divestiture or simple oversight. The consuming organization, however, keeps DNS records, SPF includes, CNAMEs and script references pointing to that domain. The domain is re-registered by an attacker. All the inherited trust — SPF authentication, script execution, email receipt — passes instantly to the new controller, with no technical alarm on the organization's side.

The case · a bank with SPF authorizing an already-expired vendor domain

A bank contracted, in 2018, an email marketing vendor — `envio-camp.com.br` — to send transactional and promotional communications. So that these messages would pass the anti-spam filters, the bank included the vendor's domain in its SPF record:

```
v=spf1 include:_spf.envio-camp.com.br include:_spf.outras.com.br ~all
```

Years later, the vendor ceased operations. The include, however, remained in the bank's SPF — the domain was in no inventory of active vendors, and the content of the record was not reviewed.

In the **first days of operation** of CSURFACE at the bank, the mapping of the chain of domains referenced in the external configuration identified that `envio-camp.com.br` **was already expired** — the domain authorized by the SPF include was, at that moment, free for public registration by any party. The alert was escalated to the security team, which removed the include from the SPF before a third party could re-register the domain. **The vector was neutralized before it could be exploited.**

What would have been at risk without the detection. With the include active and the domain re-registered by a third party, the attacker would publish `_spf.envio-camp.com.br` authorizing their own IPs and send SPF-authenticated emails as if from the bank. Anti-spam filters would accept them — SPF passes, IP authorized. Hundreds of thousands of authenticated phishing emails could be delivered to the bank's customers; credential theft at scale; regulatory exposure before federal banking regulators and the FTC. The signal — the expiration of the domain in the chain — had been public for more than a year. It was precisely this signal that the platform captured.

Other variations of the same pattern

CNAME to an expired domain. A subdomain points to `cdn-do-fornecedor.com`, a vendor that shut down in 2021. The domain expires; the attacker registers it and begins serving scripts or pages under the client organization's name.

MX to an expired domain. A subdomain (`fatura.empresa.com.br`) has an MX record pointing to the server of a defunct vendor. The attacker registers the domain and receives emails addressed to that subdomain — including password recovery, invoices and vendor communications.

Script src to an expired domain. Legacy pages load, via `<script src>`, a file from a domain that no longer exists. The attacker registers the domain and executes code under the trusted name of the hosting page — a vector equivalent to Polyfill.io.

Orphaned DKIM. A DKIM record pointing to a public key on a defunct domain can be hijacked, allowing the attacker to sign emails as if from the legitimate organization.

How CSURFACE detects it. The platform continuously monitors all domains referenced in the organization's external configuration — including SPF includes, MX, CNAMEs, NS and script references. For each one, it observes registration state, expiration date and ownership changes in WHOIS. The imminent — or consummated — expiration of a domain in the chain is a critical alert, brought to the teams before re-registration is possible.

WHY THE TRADITIONAL APPROACH FAILS · PART 03

The questionnaire does not see code

Why traditional TPRM and SCA approaches cover the contract — and leave out the real technical risk.

Three gaps in the traditional approaches

Vendor questionnaires do not see code. Traditional third-party risk management (TPRM) assesses vendors through self-declared questionnaires. This meets compliance requirements, but says little about the real risk: one does not fill out a questionnaire for each of the dozens of npm dependencies of an application.

Traditional SAST and SCA look only at proprietary code and package-manager dependencies. They do not see the JavaScript injected at runtime by a tag manager, they do not see the external CDN that serves a critical file, and they do not track domain ownership changes.

CSP is a defense; diagnosis requires separate inspection. A well-configured Content-Security-Policy blocks unlisted scripts. But most sites start with a permissive policy and never restrict it — and CSP, on its own, does not reveal which scripts the application actually loads.

The common gap. None of the traditional approaches was designed to inspect what the application executes in real time, nor to continuously monitor the posture and ownership of vendors after contracts are formalized.

Where each approach stops

WHAT NEEDS TO BE SEEN	TRADITIONAL TPRM / SCA	CSURFACE MAPPING
JavaScript loaded at runtime	Not covered	**Headless rendering**
External CDN of a critical file	Not covered	**Mapped by inspection**
Domain ownership change	Not covered	**WHOIS alert**
Transitive dependencies beyond N1	Partial	**Three levels**
Vendor's real external posture	Self-declared	**Externally verified**
Assessment frequency	Point-in-time / annual	**Continuous**

The practical result. The organization maintains a repository of completed questionnaires and a formal compliance indicator — while the highest-risk technical dependencies have never been inspected and remain unmonitored after contracts are formalized.

MAPPING ACROSS 3 LEVELS · PART 04

From the direct dependency to the subvendor of the subvendor

The CSURFACE approach: three levels of technical depth, mapped and continuously monitored.

THE DIGITAL SUPPLY CHAIN, UNFOLDED · THREE LEVELS OF TECHNICAL DEPTH

1

N1 · Direct dependencies

For each discovered asset, the platform identifies the JavaScript loaded at runtime via headless analysis, the package manifests in public repositories, the APIs called at runtime and the webhooks received.

2

N2 · Subvendors

For each Level 1 vendor, CSURFACE maps its dependencies: which CDNs it uses, which APIs it calls and where it hosts its infrastructure. The transitive dependency of a library is also unfolded here.

3

N3 · Subvendor of the subvendor

The maximum depth that public data allows — the point where Polyfill.io was: the domain operator, the IP and the CDN that actually serve the code to the user's browser.

LEVEL 1 · DIRECT DEPENDENCY

Your application
contracts the marketing vendor

The contract the organization knows about and that appears in the vendor inventory.

LEVEL 2 · SUBVENDOR

Marketing vendor
embeds the polyfill script in the page

The inherited dependency — added by third parties, with no direct contract with the organization.

LEVEL 3 · SUBVENDOR OF THE SUBVENDOR

polyfill.io
domain operated by a third party · unknown IP and CDN

The node that changed hands and began serving malicious code — invisible until the chain is unfolded.

Signals monitored continuously · for each node of the chain

CRITICAL **Ownership changes (WHOIS)**
Alert when a domain in the chain changes hands — the exact signal that was missing in the Polyfill.io case.

WARNING **Vendor's external posture**
Vulnerabilities and configurations exposed at the perimeter of each node of the chain.

WARNING **DNS changes**
Changes in NS, MX and A records that indicate a redirection of infrastructure.

CRITICAL **Leaked credentials**
Vendor credentials for sale or exposed in public forums and repositories.

WARNING **Certificates**
Renewals, change of issuer and proximity of expiration of each certificate.

CRITICAL **Incident disclosures**
Public announcements of a compromise of the vendor itself.

Headless analysis performs the full rendering of the page, including dynamic scripts; monitoring operates continuously, with no agent installed in the client's environment.

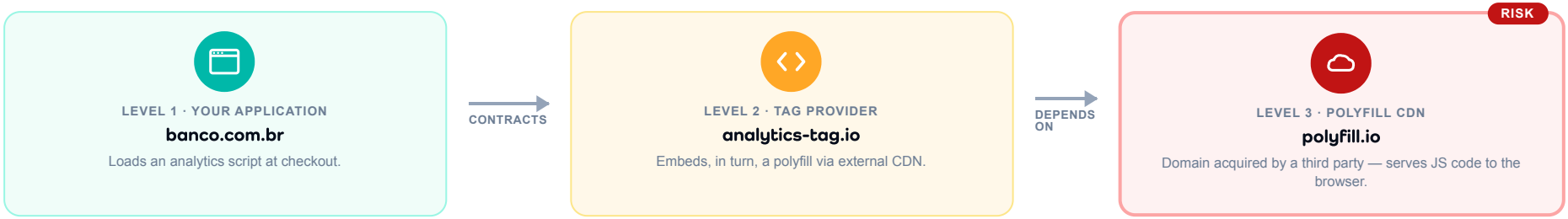
MAPPING ACROSS 3 LEVELS · PART 05 · DIAGRAMS

Two practical diagrams · web and DNS

The same three-level structure applied to two distinct categories — page code and email records. In both, the risk lives at the third level.

The pattern is structural and holds for any category of vendor. In both examples, the organization contracts a Level 1 vendor, this vendor in turn depends on a Level 2, and the real risk emerges at a Level 3 not mapped by the organization — yet which decides the outcome.

EXAMPLE 01 · CHAIN OF SCRIPTS IN THE BROWSER



EXAMPLE 02 · EMAIL AUTHORITY CHAIN (SPF)

Where CSURFACE detects this pattern. Continuous discovery maps both chains — JavaScript at runtime (headless rendering) and domains in external configuration (SPF includes, CNAME, MX, NS, script src). For each node across the three levels, the platform observes registration state, ownership in WHOIS, DNS changes and content variations. The signal — change of ownership, imminent expiration, hash alteration — reaches the team before exploitation.

READING BY ROLE · PART 06

The same digital supply chain, three different cuts

Managers, engineers and analysts need distinct cuts of the same data to decide, configure and respond to threats in the digital supply chain.

Why there are three cuts. Digital supply chain risk crosses three organizational layers with different horizons — capital, configuration and operation. The **manager** needs the aggregate figure in currency to deliberate and account for it; the **engineer** needs the technical detail to remediate; the **analyst** needs the context and the indicators to respond in real time. Without this segmentation, the same report serves all three poorly.

For the manager · CISO, risk manager

What matters. Aggregate financial exposure, adherence to the approved risk appetite, compliance with applicable standards (GLBA/FFIEC, state privacy laws, NIS2, PCI DSS) and coverage metrics — how many vendors are under monitoring, how many critical alerts are open, how long it took from detection to resolution.

What they receive from CSURFACE. An executive dashboard with a consolidated inventory of vendors across three levels, open critical alerts, comparison against appetite, auditable evidence for the committee. Periodic reports that can be presented to the board with no additional technical translation.

How they act. Approves remediation investment based on the trade-off of exposure reduction. Communicates the organization's position to the board, the regulator and the auditor with verifiable data. Compares the internal posture with that of industry peers.

For the engineer · AppSec, DevSecOps

What matters. Complete technical details of the finding — which script was altered, which hash changed, which subdomain points to which deallocated resource, which SPF include references which expired domain. The exploitation vector, the context in the architecture and the remediation path.

What they receive from CSURFACE. Technical alerts with full context — affected domain, origin resource, evidence of the change (hash before/after, WHOIS before/after), integration with CI/CD and ticketing pipelines. Concrete remediation recommendation for each category.

How they act. Removes the problematic dependency, configures a restrictive CSP and SRI on critical scripts, updates SPF records, removes orphaned CNAMEs, applies `HttpOnly` and `SameSite` flags on session cookies. Documents the change in the technical inventory.

For the analyst · SOC, incident response

What matters. Real-time threat context — indicators of compromise, activity observed in the chain's components, correlation with threat intelligence, signs of active exploitation at industry peer organizations.

What they receive from CSURFACE. Alerts correlated with threat intelligence and with the vendor's external posture, indication of exploitation in progress, temporal context (when the signal appeared, when it changed, who else was affected), IOC enrichment for investigation.

How they act. Investigates anomalous activity originating from external components, conducts containment when the vector is confirmed, communicates internally, escalates to the engineer for remediation and to the manager for deliberation. Maintains the forensic trail.

APPLICATION AND RESULTS · PART 05

Preventing the next Polyfill.io

A representative use case and what to expect in the first weeks of operation.

TRADITIONAL INVENTORY

1×

vendors known and declared in contract — CDN, analytics, A/B, chat, AdTech.

→

AFTER THE THREE-LEVEL MAPPING

49×

components in the real digital supply chain — invisible until the chain is unfolded.

WHAT CSURFACE OBSERVES · THE PLATFORM'S OWN DATA

2.8×
direct vendors (Level 1) for each of your own inventoried assets

11×
expansion of the chain adding subvendors (Level 2)

49×
real size of the chain unfolded to Level 3

Illustrative scenario · mid-sized e-commerce

An e-commerce operation with 12 sites, three frontends (web, mobile web and B2B) and more than 80 external JavaScript scripts in the DOM activated the CSURFACE mapping. In the first week, the platform unfolded the chain to the third level — **multiplying by more than 40× the number of vendors** that appeared in the traditional inventory. Three critical alerts were issued:

CRITICAL **A polyfill.io script still active**
A tag manager, on an old page, still pointed to the domain — never removed after the public incident.

WARNING **A chat CDN domain expiring in 11 days**
With the risk of being registered by a third party and beginning to serve code under the trusted name.

CRITICAL **A tracking pixel with a new owner**
The vendor changed ownership two months earlier — a WHOIS change detected by the platform.

Time-to-value

- **Day 1** — onboarding; the initial headless crawl is completed in a few hours.
- **Day 3** — complete inventory of dependencies across the three levels.
- **Week 1** — continuous monitoring active; first alert on every change.
- **Month 1+** — continuous operation, with a per-vendor risk dashboard.

Want to discover the size of your digital supply chain — free of charge?

In a preliminary analysis at no cost, CSURFACE maps the three levels of vendors for one of your public assets and returns the complete technical report.

Map my chain →

COMPLIANCE · PART 06

The same work that protects and proves

How continuous mapping of the digital supply chain meets regulatory requirements with no additional effort.

Regulatory requirements met

STANDARD	REQUIREMENT	HOW THE MAPPING CONTRIBUTES
FFIEC / GLBA	Continuous assessment of critical third parties	**Continuous, not point-in-time, monitoring**
ISO 27001	Supplier relationships (Annex A)	**Verifiable technical inventory**
NIST CSF	Supply Chain Risk Management	**Three-level mapping**
PCI DSS 4.0	Management of service providers (Req. 12.8)	**Always-current list of third parties**
EU NIS2	Supply chain risk management	**Evidence of continuous assessment**

Applicable to operations subject to each regulation; NIS2 applies to organizations operating in the European Union.

Compliance as a by-product. Continuous mapping of the digital supply chain serves security and compliance at once. The verifiable, always-updated technical inventory is, at once, the security control and the evidence the auditor asks for — with no rework.

From the questionnaire to technical evidence

The traditional model proves compliance with a repository of self-declared questionnaires. The CSURFACE mapping replaces that model with a continuously updated technical inventory:

- Each vendor identified by direct technical inspection.
- Each change recorded with a date and event type.
- Each critical alert documented from detection to resolution.

The result is an audit trail that describes the chain as it technically exists, established through direct, verifiable inspection of the assets.

NEXT STEPS

Discover the subvendor before it discovers you

Digital supply chain analysis is one of the capabilities of the CSURFACE platform — a Continuous Exposure Management platform. In an integrated architecture, it brings together continuous discovery of the external surface with Machine Learning, digital supply chain analysis, active exploitability validation where test coverage exists, with passive monitoring across the rest of the surface, threat intelligence with dynamic prioritization and monitoring of leaked credentials. It operates entirely externally — with no agent and no installation — with optional cloud, WAF and CIEM integrations available for organizations seeking greater depth of visibility.

Map one of your sites

In a 30-minute technical session, CSURFACE performs the digital supply chain mapping of an asset of your choice across the three levels of depth.

Use the risk calculator

Estimate your sector's annual expected loss with the CSURFACE risk calculator, calibrated by market benchmark.

Read the ML Discovery whitepaper

Continuous discovery of the external surface with Machine Learning — the foundation on which vendor mapping operates.

See your real digital supply chain — mapped three levels deep

Get a free preliminary analysis