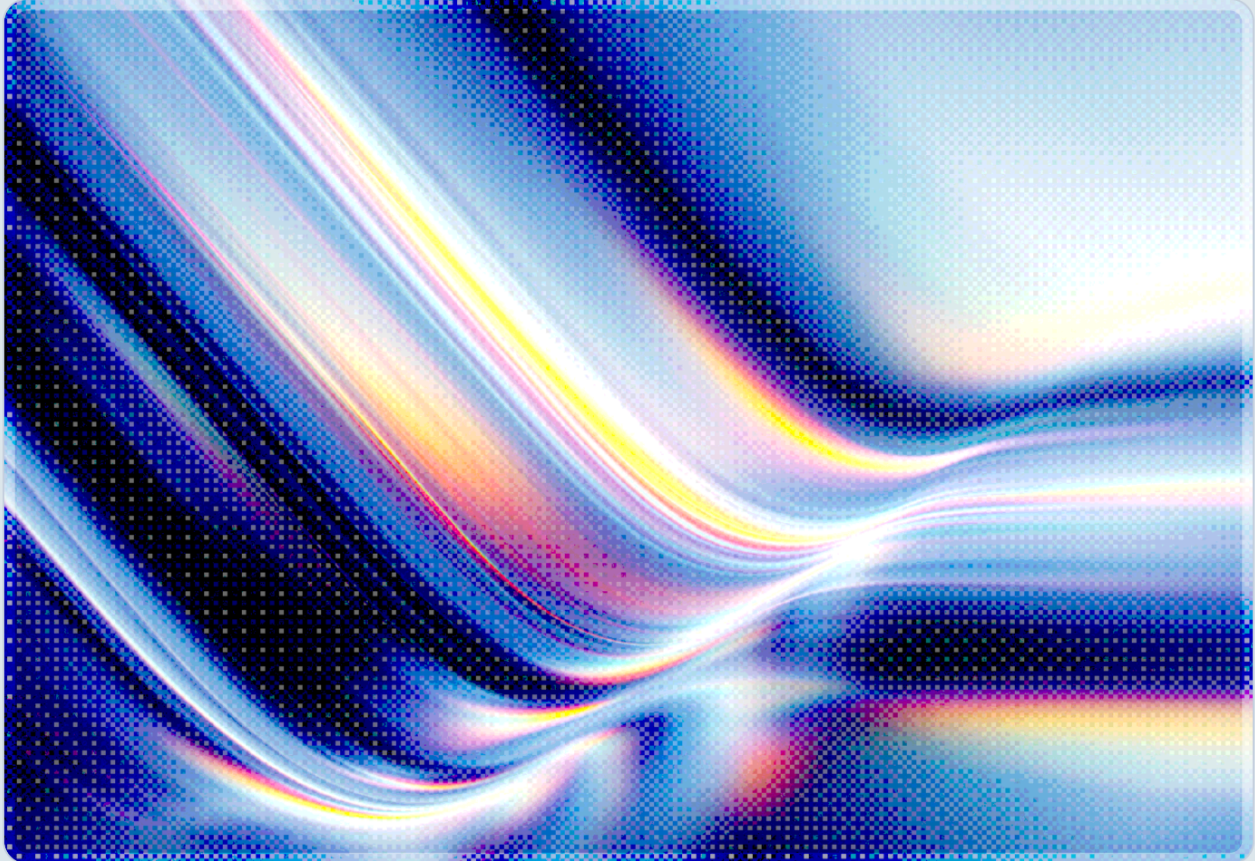


# x402 Payment SDK Flaws: The Missing Advisory Trail

Uncapped spending, silent fixes, and why a spending cap is not a price check

2026-10-26

 AI-assisted Rapid Research



**© 2026 Cloud Security Alliance. Some rights reserved.**

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

*This document was generated with AI assistance and has not undergone official CSA review and approval processes.*

---

## Key Takeaways

On October 22, 2026, the security firm V12 disclosed 25 vulnerabilities in the x402 agent-payment codebase, three of them rated critical [1]. For enterprises, CSA considers the most consequential to be a default client with no spending limit, which let a malicious API ask an automatically signing agent for the full balance of a supported token [1][2]. CSA's research note of October 25 covered the three critical findings and the upgrade path in detail [17]. This note follows up on that analysis. It draws on a review of the x402 Foundation's public repository and advisory records on October 26 and on new data published since the disclosure. It focuses on two questions the initial coverage left open: how complete the project's public security record is, and whether the patched default is enough on its own.

The repository record shows a pattern. The fixes themselves were fast. Pull requests addressing all three critical findings were merged between August 12 and August 18, 2026, within six days of V12's report (V12 dates the fixes to August 20; the merge dates in the repository are August 12–18) [1][6][7][8][9][23][24]. The disclosure trail, however, is thin. As of October 26, the repository has published only two security advisories, the newer one in March 2026, and neither concerns these findings [4][5]. The GitHub Advisory Database lists no entries at all for `@x402/fetch` or `@x402/extensions` [21]. The pull request that added the spending cap was titled as a feature, and the package changelog lists the release under "Minor Changes" [6][10]. A separate SIWx server-side authorization bypass has since received a CVE from a third-party numbering authority, CVE-2026-113603, which the project has not published as a repository advisory. The GitHub Advisory Database holds only an unreviewed entry for it with no package mapping, so Dependabot and `npm audit` do not flag it [13][22]. Its listed affected range extends to the latest releases reported in October coverage, and the linked fix remains unmerged [2][13][15].

The patched default is also narrower than it may appear. A \$1 per-payment cap limits how much a hostile server can take in a single payment, but it does not tell the client what the right price is. An October 21 testnet reproduction against the patched v2.28.0 client showed it signing an authorization for ten times a resource's price, because the inflated amount fell under the cap [11]. On-chain measurement of Base settlements in the same period found payments far above sellers' usual prices settling normally. The researchers caution that this pattern can also come from legitimate tiered pricing [12].

Security teams should therefore take three steps. They should upgrade x402 dependencies by direct version comparison, not by waiting for scanner alerts. They should configure a per-request price expectation, not only a cap. And they should treat the absence of x402 advisories as missing information, not as evidence of safety.

# Background

x402 revives the HTTP `402 Payment Required` status code as a payment handshake. A server answers a request for a paid resource with a 402 response that lists acceptable payment terms (amount, asset, network, and recipient). The client signs a payment that meets one of those terms, usually a stablecoin transfer authorization, and retries the request with the payment attached. A facilitator service can verify and settle the payment for the merchant. Coinbase created the protocol [16]. On April 2, 2026, the Linux Foundation announced that it would host the x402 Foundation to govern the protocol neutrally, with payments, cloud, and commerce companies among the founding participants [16]. Because the protocol is designed for autonomous agents paying per request, the reference SDKs in TypeScript, Python, and Go can effectively act as the authorization layer for agent spending wherever operators rely on their defaults.

V12's write-up gives the disclosure timeline. Its researcher reported the 3 critical and 13 high-severity findings on August 12, 2026, and the critical issues were fixed by August 20. The 9 medium- and low-severity findings followed on September 9 [1]. Only the three critical findings are described publicly: the uncapped default client, a Solana facilitator that accepted an `ApproveChecked` instruction (which moves no tokens) as a `TransferChecked` payment, and a Sign-In-With-X (SIWx) client that signed login challenges relayed from other sites [1]. The Hacker News reported on October 23 that the fixes shipped in releases dated August 18 (TypeScript packages at 2.23.0). It also reported that the Foundation had published no advisory, so `npm audit` and PyPI showed no vulnerabilities for the last affected releases [2]. Neither V12 nor the Foundation has reported exploitation in the wild [2]. The sources reviewed for this note do not include a public statement from the Foundation about how the fixes were disclosed.

## Security Analysis

The analysis below addresses the two open questions in turn. It first examines the project's disclosure record and what that record means for the tooling enterprises use to find vulnerable dependencies. It then turns to the patched spending cap and asks whether it protects agents against a server that overcharges within the cap's limit.

## Fast Fixes, Missing Records

The repository history does not suggest a maintainer team that ignored the report. The TypeScript fix for the Solana discriminator check, PR #3132, was opened and merged on August 12, the same day V12 reports submitting its findings. Its description states plainly that the TypeScript path "validated the token program at instruction index 2 but not the instruction discriminator" [7]. The SIWx origin-binding fix (PR #3133) and the TypeScript spend-controls change (PR #3124) merged on August 13. The Python spend controls (PR #3154) and the Go spend-controls and SIWx fixes (PRs #3156 and #3193) merged on August 18 [6][8][9][23][24]. PR #3124 was opened on August 11, one day before V12's reported submission date, so it may have been under way independently or prompted by earlier informal contact; the sources reviewed do not say which [6]. By the standard of time to remediation, this is a fast response.

The weakness is in how those changes were communicated. PR #3124 is titled "feat(TS): spend controls." Its description presents the \$1 default cap and asset allowlist as new configuration, and it does not mention a vulnerability [6]. In the `@x402/fetch` changelog, 2.23.0 appears under "Minor Changes." The SIWx change is described as a new required argument to `createSIWxPayload`, with no reference to the session-takeover risk that V12 identified and that the change closes [1][10]. A developer reading the changelog would see an API change, not a security fix. The spending cap is the fix most exposed to reversal, because the same release that introduced it also provides `spendControls: false` to remove it [6]. A team whose integration broke under the new \$1 cap could restore the old behavior with one flag, without knowing that the flag also brings back a critical vulnerability.

The project's formal security channels explain part of this. The repository's `SECURITY.md` asks reporters to contact a Technical Steering Committee member or maintainer privately through the project's Slack workspace. It notes that a bug bounty is "coming soon." It does not commit the project to publishing advisories, requesting CVEs, or labeling security releases [3]. The repository's advisory page lists only two published advisories, GHSA-3j63-5h8p-gf7c from August 2025 and GHSA-qr2g-p6q7-w82m from March 2026. Both carry the generic title "x402 SDK Security Advisory" and give little technical detail [4][5]. The March advisory shows the project has used GitHub's advisory process before. The August fixes, publicly disclosed in October, show that the process has not been used consistently.

## Why the Gap Matters to Tooling

Most enterprises learn about vulnerable dependencies from software composition analysis (SCA), which matches installed versions against advisory databases. A query of the GitHub Advisory Database on October 26 returned no advisories for the npm packages `@x402/fetch` or `@x402/extensions`, and only GHSA-qr2g-p6q7-w82m for the Python `x402` package [21]. Dependabot and `npm audit`, which draw on the GitHub Advisory Database, therefore had nothing to report. Unless a team's SCA

vendor independently added these findings, an organization pinned to a pre-fix version had no automated signal that its agents' default payment client was uncapped for roughly two months, from the August 18 fixes to V12's October 22 publication. The same condition likely continues for teams that rely on GitHub-Advisory-driven tooling and missed the news coverage. CSA did not survey commercial SCA feeds for this note.

The SIWx server-side flaw shows the same gap in a case that is still open. On October 17, a contributor filed public issue #3838, reporting that the SIWx settle hook stored a payment entitlement under the client-supplied `PaymentPayload.resource.url` [14]. VulnCheck, which assigns CVEs as a numbering authority, recorded this on October 20 as CVE-2026-113603. It is a medium-severity authorization bypass (CVSS 4.0 score of 6.9) through which a client pays for the cheapest SIWx-protected route and then reuses its sign-in header to reach more expensive routes [13]. The record lists affected ranges of `@x402/extensions` 2.3.1 through 2.28.0, Python `x402` 2.11.0 through 2.25.0, and the Go module 2.16.0 through 2.28.0. Those are the latest releases reported in October coverage [2] [13]. The linked fix, PR #3840, was opened on October 18 and was still open on October 26 [15].

This flaw harms merchants rather than paying agents, and its severity is moderate. The issue was filed publicly rather than through the private channel `SECURITY.md` describes, and a third-party numbering authority assigned a CVE within three days [3][13][14]. As of October 26, the project's advisory page does not reflect it, and the only GitHub Advisory Database entry is an unreviewed record (GHSA-vxw2-vmfp-vrg3) with no affected packages mapped [4][22]. Together with the absent advisories for the August fixes, this suggests the project's advisory process is not currently the reliable record of its security fixes.

The table below summarizes the public disclosure record for the x402 security issues discussed in this note as of October 26, 2026.

| Issue                              | Fix Merged                                                    | Project Advisory (GHSA) | CVE  | Visible to Advisory-Driven SCA |
|------------------------------------|---------------------------------------------------------------|-------------------------|------|--------------------------------|
| Uncapped default client (critical) | August 13 (TS),<br>August 18 (Py, Go),<br>2026 [6]<br>[9][23] | None                    | None | No                             |
| Solana approval                    | August 12,<br>2026 [7]                                        | None                    | None | No                             |

| Issue                                             | Fix Merged                                      | Project Advisory (GHSA)                                                                        | CVE                              | Visible to Advisory-Driven SCA                                                                                                     |
|---------------------------------------------------|-------------------------------------------------|------------------------------------------------------------------------------------------------|----------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| accepted as payment (critical)                    |                                                 |                                                                                                |                                  |                                                                                                                                    |
| SIWx client signs relayed challenges (critical)   | August 13 (TS),<br>August 18 (Go), 2026 [8][24] | None                                                                                           | None                             | No                                                                                                                                 |
| SIWx server entitlement under client URL (medium) | Open as of October 26 [15]                      | None (unreviewed GHSA-vxw2-vmfp-vrg3 in the GitHub Advisory Database, no package mapping) [22] | CVE-2026-113603 (VulnCheck) [13] | Not in GitHub-Advisory-driven tools ( <code>npm audit</code> , Dependabot); visible in tools that ingest CVE/NVD or VulnCheck data |
| Earlier Solana verification flaw                  | Before March 2026                               | GHSA-qr2g-p6q7-w82m [5]                                                                        | None listed                      | Yes                                                                                                                                |

Sources: x402 repository pull requests, issues, and advisory pages; GitHub Advisory Database; VulnCheck. Fix dates are pull-request merge dates, not release dates.

## A Cap Is Not a Price

The second open question is whether the patched default actually protects agents. After 2.23.0, the default client caps each payment at \$1 and accepts only recognized US-dollar stablecoins [2][6]. That limits the worst case from one payment to one dollar. It does not stop a server from overcharging within that limit. On October 21, a contributor using the open-source Argus-Agent-Test-Lab filed issue #3866 against `@x402/fetch` v2.28.0. A test server priced a resource at 0.01 USDC and then returned a 402 response asking for 0.10 USDC. The official client, configured with a \$1 `maxAmountPerPayment` , signed an EIP-3009 transfer authorization for the inflated amount in three of three sessions on the Base

Sepolia testnet [11]. Reviewers in the thread pointed out that the test shows a signed authorization, not a confirmed on-chain overpayment, and the reporter accepted that limitation [11]. The underlying point stands: the client treats whatever amount the server declares as the price, and the only check is the cap.

The SDK does provide a better control, but only on an opt-in basis. An `onBeforePaymentCreation` hook lets the caller compare the requested amount, recipient, network, and asset with a price it already expects, and refuse to sign on a mismatch. The issue notes that the official client example does not demonstrate this hook. Agents built from that example will therefore pay any amount under the cap [11]. For an agent making many small purchases, a tenfold overcharge on each one could add up to a meaningful systematic loss while staying far below any per-payment cap.

Independent on-chain data shows that this pattern is real at least in shape. Agentic Finance Graph (AFG) analyzed stablecoin settlements to x402 sellers listed in the Bazaar discovery service on Base mainnet between October 12 and October 23, 2026. The sample covered 80 sellers with at least 20 settlements each, totaling 32,605 settlements and \$18,950.78 [12]. Of those settlements, 1,126 (3.5 percent) were at least ten times the seller's median payment. Twenty-five were at least one hundred times the median and at least \$1, worth \$1,928.23 in total and sent by 13 payers to 7 sellers. The largest was \$588.00 against a \$0.01 median [12]. AFG states explicitly that tiered or usage-based pricing produces the same pattern. It presents the figures as an upper bound on what the overcharge path could have produced, not as a count of attacks [12].

Two qualifications follow from AFG's data. First, the largest outliers exceed the \$1 default cap, so those payers were not using the patched default as shipped, and the cap would have blocked them. AFG also reports that 10 of the 13 payers behind these large cases made only one or two settlements in the period, that none used ERC-8004-bound agent wallets, and that the large cases "look like single purchases, not an agent in a loop" [12]. Second, settlements at ten times the median of a low-priced seller can still fall well under a \$1 cap, as the \$0.01 median cited above illustrates, but AFG does not report how many of the 1,126 were under \$1. CSA therefore reads the data cautiously. It does not show that overcharging happened. It does show that payments far above a seller's normal price settle on mainnet without any protocol-level signal, and that for low-priced resources a flat cap set well above the expected price would not catch them.

## Implications for Agent Deployments

Together, these findings point to a design issue that goes beyond any single bug. In x402, the payee proposes the price, and the payer's software decides whether to sign. Before August, the reference client signed every proposal. Now it signs every proposal under one dollar. Neither default reflects what the agent's operator actually agreed to pay. Agents also choose which URLs to request in part from content they retrieve. An agent steered by prompt injection or a poisoned tool listing could therefore reach a

server that overcharges, or that relays a login challenge from another site. CSA's inference is that the payment authorization decision needs an independent price expectation set outside the agent's reasoning. The disclosure record shows that operators cannot count on upstream advisories to tell them when the protections they depend on change.

## Recommendations

The recommendations below follow from the two findings of this note. Because upstream advisories cannot be relied on, the immediate actions center on direct inventory and version checks. Because the patched cap does not verify price, the short-term mitigations center on explicit price expectations enforced outside the agent.

### Immediate Actions

Organizations should first inventory every agent, service, and framework that depends on x402 packages, including transitive dependencies, and compare versions directly against the fixed releases instead of relying on scanner output. CSA's October 25 note gives the full version table [17]. At minimum, TypeScript clients need `@x402/fetch` or `@x402/axios` 2.23.0 or later, SIWx users need `@x402/extensions` 2.23.0 or later, Solana facilitators need `@x402/svm` 2.23.0 or later, and Python and Go clients need releases from August 18, 2026 or later [2][9][23][24]. Teams should search code and configuration for `spendControls: false`. Every occurrence should be treated as a disabled security control that needs a documented owner and justification [6]. Merchants that use the SIWx server extension should review route entitlements and track PR #3840 until a fixed release ships [13] [15].

### Short-Term Mitigations

Over the next several weeks, teams should replace round-number caps with explicit price expectations. The first change is to set `maxAmountPerPayment` close to the price the agent actually expects to pay for each class of resource, not to a generic ceiling [12]. The second is to implement an `onBeforePaymentCreation` check that compares amount, recipient, network, and asset against a quote or allowlist the operator controls, and refuses to sign on any mismatch [11][12]. The third is to log every 402 challenge alongside the signed authorization and alert when the requested amount for the same resource changes between requests or exceeds the recorded expectation. Taken together, these changes turn the cap from the only line of defense into a backstop behind a price check.

These checks are more reliable in a signing proxy or policy service that the agent cannot reconfigure than in in-process settings that a manipulated agent might influence. Wallets that agents sign from should hold only a short operating period's working balance, so that a control failure has a bounded cost.

## Strategic Considerations

Organizations that depend on x402 should ask the x402 Foundation and its member companies to publish GitHub security advisories and request CVEs for the three August fixes and for any of the remaining 22 V12 findings with security impact. They should also ask the Foundation to commit, in `SECURITY.md`, to labeling security fixes in changelogs and release notes. The Linux Foundation hosting model and the involvement of payment networks among the founding participants [16] may give enterprise members leverage to make that request. Vendor-risk teams evaluating agent platforms that embed x402 should ask how the vendor tracks upstream security changes when no advisory exists. More broadly, CSA recommends treating every agent payment capability as delegated financial authority. That authority should have an accountable owner, a defined price expectation, and a revocation path. The infrastructure that carries it should meet the disclosure expectations applied to other payment software: published advisories, CVE assignment for security fixes, and clearly labeled security releases.

## CSA Resource Alignment

The most directly relevant CSA work is the AI Safety Initiative's research note *x402 Agent-Payment SDK Flaws: Uncapped Spending and Silent Fixes* [17], published October 25, 2026. That note analyzes the three critical findings, gives the full version reference, and maps the issues to CSA guidance on agent identity and authorization. The present note builds on it. It adds the repository's advisory record, the third-party CVE for the SIWx server flaw, and evidence that the patched cap does not prevent overcharging within its limit. Readers should consult both notes together.

CSA's research note *Reforming Coordinated Vulnerability Disclosure for the Autonomous Bug Hunter Era* [18] argues that disclosure infrastructure is under strain. It warns specifically about findings that enter vendor programs and produce "no CVE, no advisory, and no public record." The x402 case is a concrete example of that warning. V12's 25 findings led to fast code fixes but no public advisories. The one CVE that does exist was assigned by a third party from a public bug report. The reforms that note recommends, including reporting transparency and clear accountability for the disclosing party, apply directly to open-source projects that sit on payment paths.

At the control level, the AI Controls Matrix (AICM) v1.1 [19] provides the mapping for assurance work. Its Threat and Vulnerability Management domain covers tracking and remediating vulnerable upstream components when advisories are missing. Its Identity and Access Management domain covers the delegated financial authority an agent wallet represents. Its Supply Chain Management, Transparency, and Accountability domain covers the disclosure obligations enterprises should expect from suppliers such as the x402 Foundation. Teams threat-modeling agent payment flows can use MAESTRO [20] to analyze the trust boundary between the agent, the external APIs it calls, and the payment layer. In this case, the 402 handshake is where a counterparty's proposal becomes the agent's signed commitment.

# References

- [1] Renwa, V12. "[25 Vulnerabilities in the x402 Protocol: Inside Three Critical Findings](#)." V12 Blog, October 22, 2026.
- [2] Khandelwal, Swati. "[Critical x402 SDK Flaw Could Let Malicious APIs Drain AI Agents' Token Balances](#)." The Hacker News, October 23, 2026.
- [3] x402 Foundation. "[Security Policy \(SECURITY.md\)](#)." GitHub, accessed October 26, 2026.
- [4] x402 Foundation. "[Security Advisories, x402 Repository](#)." GitHub, accessed October 26, 2026.
- [5] x402 Foundation. "[GHSA-gr2g-p6q7-w82m: x402 SDK Security Advisory](#)." GitHub, March 2026.
- [6] x402 Foundation. "[feat\(TS\): spend controls, Pull Request #3124](#)." GitHub, opened August 11, 2026, merged August 13, 2026.
- [7] x402 Foundation. "[fix\(ts\): require TransferChecked discriminator in svm exact, Pull Request #3132](#)." GitHub, merged August 12, 2026.
- [8] x402 Foundation. "[fix\(ts\): bind SIWX client challenge to the request origin, Pull Request #3133](#)." GitHub, merged August 13, 2026.
- [9] x402 Foundation. "[feat\(py\): add spend controls, Pull Request #3154](#)." GitHub, merged August 18, 2026.
- [10] x402 Foundation. "[@x402/fetch CHANGELOG](#)." GitHub, accessed October 26, 2026.
- [11] igor-vii. "[@x402/fetch client signs and submits a payment for ANY amount the server declares in the 402 requirements, Issue #3866](#)." GitHub, October 21, 2026.
- [12] Agentic Finance Graph. "[What a Disclosed x402 SDK Flaw Looks Like on Base, Measured](#)." DEV Community, October 24, 2026.
- [13] VulnCheck. "[x402 @x402/extensions 2.3.1 through 2.28.0 SIWX Payment Bypass via PaymentPayload.resource \(CVE-2026-113603\)](#)." VulnCheck, October 20, 2026.
- [14] Leeziao (contributor). "[SIWX settle hook records the entitlement under the client-supplied PaymentPayload.resource.url, Issue #3838](#)." GitHub, October 17, 2026.

- [15] x402 Foundation. "[fix: record SIWX payments on the server request path, Pull Request #3840](#)." GitHub, opened October 18, 2026.
- [16] Linux Foundation. "[Linux Foundation is Launching the x402 Foundation and Welcoming the Contribution of the x402 Protocol](#)." Linux Foundation, April 2, 2026.
- [17] Cloud Security Alliance. "[x402 Agent-Payment SDK Flaws: Uncapped Spending and Silent Fixes](#)." CSA Lab Space, October 25, 2026.
- [18] Cloud Security Alliance. "[Reforming Coordinated Vulnerability Disclosure for the Autonomous Bug Hunter Era](#)." CSA Lab Space, June 7, 2026.
- [19] Cloud Security Alliance. "[AI Controls Matrix \(AICM\) v1.1](#)." Cloud Security Alliance, 2026.
- [20] Cloud Security Alliance. "[Agentic AI Threat Modeling Framework: MAESTRO](#)." Cloud Security Alliance, February 6, 2025.
- [21] GitHub. "[GitHub Advisory Database: x402](#)." GitHub, accessed October 26, 2026.
- [22] GitHub. "[GHSA-vxw2-vmfp-vrg3 \(CVE-2026-113603\), Unreviewed Advisory](#)." GitHub Advisory Database, October 20, 2026.
- [23] x402 Foundation. "[feat\(go\): spend controls, Pull Request #3156](#)." GitHub, merged August 18, 2026.
- [24] x402 Foundation. "[fix\(go\): bind SIWX client challenge to the request origin, Pull Request #3193](#)." GitHub, merged August 18, 2026.