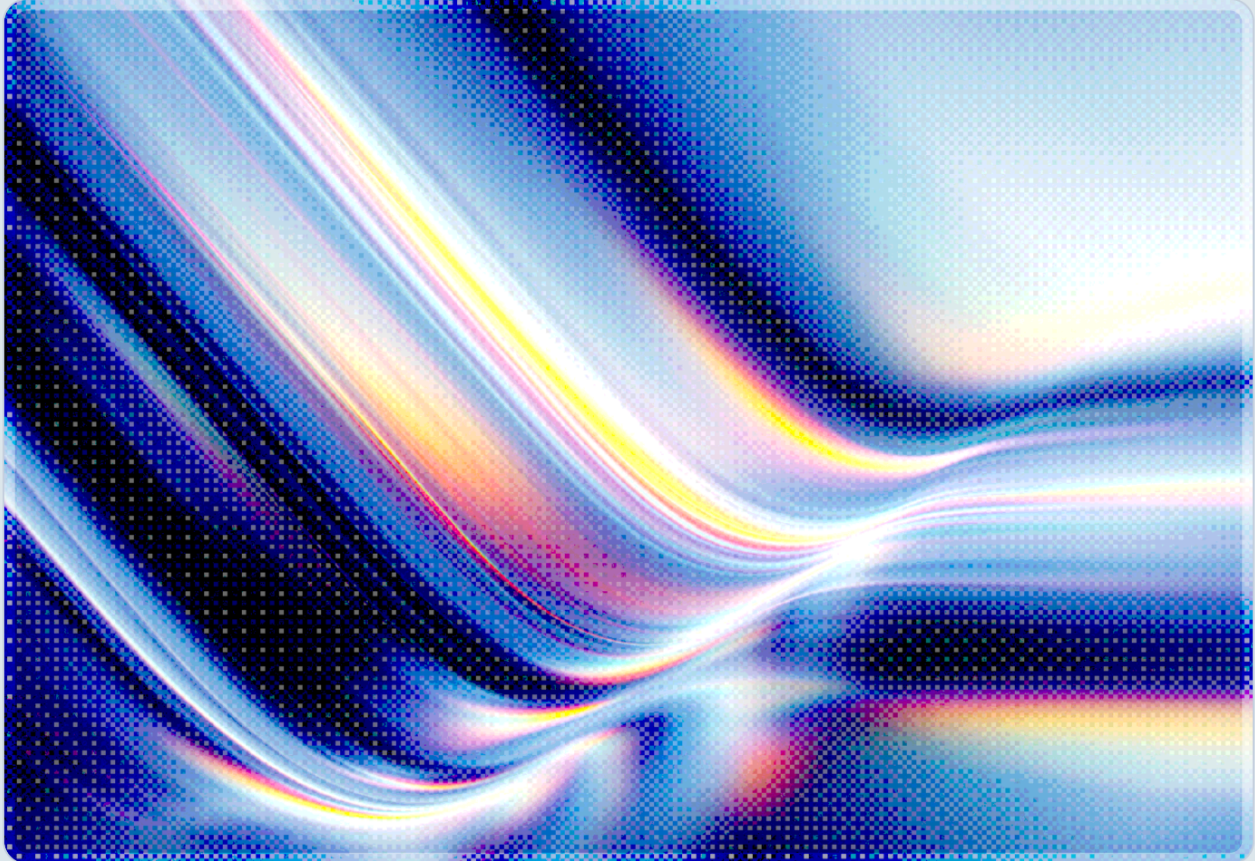


When Trust Infrastructure Fails: Keys and Relays as Chokepoints

Lessons from the eMudhra CA, UpdraftCentral, ConnectWise Cloud Backup, and Tor Exit Relay Incidents of October 2026

2026-10-21

 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

In the span of roughly three weeks in October 2026, four unrelated incidents exposed the same structural weakness: a single key, credential, or relay operator sitting upstream of thousands or millions of downstream parties. The private keys for eight eMudhra intermediate certificate authorities were generated with so little entropy that anyone could recover them from the public certificates, putting forged TLS certificates for any domain within reach of any attacker [1][3]. The practical damage appears to have been limited: no certificates from these intermediaries had been logged to Certificate Transparency before they were blocked, and eMudhra states that none had issued end-entity certificates [1][4]. Connection keys for the UpdraftCentral WordPress remote-management service were, according to the vendor, likely taken in a breach of its website and then used to push signed commands that installed backdoors on customer sites, with roughly 400,000 sites potentially exposed [6]. A vendor-held application credential for ConnectWise Cloud Backup was abused to sign in to Microsoft 365 tenants, with unauthorized sign-ins confirmed in 162 tenants and attempts observed against almost 1,000 [7]. Separately, attackers altered the exit policies of more than 300 Tor exit relays, at least some of them by compromising operators' servers, and used them to intercept SSH logins [9].

The common lesson is that trust delegated to an intermediary is only as strong as that intermediary's key handling, and that defenders rarely have an inventory of where such delegations exist in their own environments. Chrome and Firefox blocked the affected eMudhra intermediaries within hours of the report, and Apple and Microsoft followed on October 19 and 20 [1][4][5]. CT-enforcing browsers were also protected in the interim, because no certificates from the affected intermediaries had been logged [1]. Many non-browser clients, containers, and language runtimes have neither safeguard, and they required package updates and redeployment [2][5]. Remote-management and backup integrations carry standing, high-privilege access that persists until a customer explicitly removes it, which means a vendor-side breach can become a customer-side compromise without any customer-side vulnerability.

Organizations should treat this cluster of events as a prompt to take three steps. First, update and redeploy every component that carries its own trust store, not just browsers. Second, inventory and prune third-party remote-management keys and consented cloud applications. Third, adopt authentication methods, such as key-based SSH and pinned host keys, that remain secure even when a network intermediary is hostile. None of the four incidents involved an AI system directly, but AI and agentic systems increasingly hold delegated credentials and rely on the same TLS libraries and package trust stores, so the same chokepoints apply to them.

Background

Modern infrastructure depends on a small number of parties that vouch for, or carry traffic on behalf of, everyone else. Certificate authorities sign the certificates that let a TLS client trust a server it has never seen before. Remote monitoring and management (RMM) platforms, backup services, and site-management dashboards hold standing credentials that let a vendor or managed service provider act inside customer systems. Relay and anonymity networks route traffic through operators the user does not know. Each of these arrangements concentrates trust deliberately, because concentration is what makes them efficient. The cost of that efficiency is that a single failure at the concentration point propagates to every party downstream, often faster than those parties can learn that anything has happened.

The eMudhra incident began on October 17, 2026, when browser root programs received a researcher's report that the private keys for eight eMudhra subordinate CA certificates were compromised [1][3]. Six of the intermediates were TLS issuing CAs and two were restricted to code signing [4]. The keys had not been stolen; they were mathematically weak. Researcher Scott Helme found that one P-384 private key was simply the number 4, and other recovered keys were similarly small integers such as 56 and 115 [3]. According to eMudhra's October 21 statement, the intermediates were created under its ECC root on April 30, 2025, using FIPS 140-2 Level 3 HSMs, and a flaw in a key wrap and unwrap step produced a second set of weak keys that were the ones ultimately certified and published [4]. eMudhra stated that only one of the eight intermediates was in production and that none had issued end-entity certificates, while noting that its root cause investigation was still ongoing [4].

The remote-management incidents followed a different pattern. TeamUpdraft, the developer of the UpdraftPlus backup plugin, detected unauthorized access to its website over the weekend of October 17-18, 2026, and believes the attacker reached a customer database containing connection keys for UpdraftCentral, UpdraftVault, and UpdraftClone [6]. Attacks using those keys reportedly began on October 18 [6]. Days earlier, Huntress reported that an attacker had abused a credential held in ConnectWise's own tenant for its Cloud Backup application (formerly SkyKick) to obtain Microsoft Graph tokens against customer tenants that had consented to the app [7][8].

The Tor incident predates the others. Between October 1 and 5, 2026, attackers altered the exit policies of hundreds of Tor relays to redirect SSH traffic to a set of interception servers [9]. The Tor Project's directory authorities began assigning BadExit flags on October 4, and by October 5 no relay carrying the attacker's policy remained in the consensus [9]. The initial foothold, at least for one large relay family, was traced to a reused operator password and an internet-facing server with SSH password authentication enabled, after which the attacker installed the Ebury OpenSSH backdoor [9].

Security Analysis

Anatomy of the Four Chokepoints

Although the incidents differ in mechanism, they share a structure that can be compared directly. In each case, a single upstream secret or operator was trusted by a large population that had no practical way to evaluate that trust on its own. The table below summarizes what was trusted, how the trust failed, and what limited the damage.

Incident	Trusted chokepoint	Failure mode	Reported exposure / impact	Victim-side precondition	Primary containment
eMudhra intermediate CAs	Six TLS and two code-signing intermediate CA keys	Weak key generation; keys recoverable from public certificates	Potential: non-CT-enforcing clients trusting the emSign ECC Root CA - G3 hierarchy or the Sectigo AAA cross-signs; no misissued certificates observed [1][4][5]	None	CRLSet/OneCRL-style blocks, CT enforcement, root store updates [1][2][5]
UpdraftCentral	Vendor-held remote-management connection keys	Vendor database breach (per vendor); stolen keys signed attacker commands	~10% of UpdraftPlus sites, ~400,000 potentially exposed [6]	None	Key rotation, plugin update to remove cloud keys, manual key deletion [6]
ConnectWise Cloud Backup	Vendor-tenant	Credential theft	162 tenants with	Consent granted to	Credential revocation, service

Incident	Trusted chokepoint	Failure mode	Reported exposure / impact	Victim-side precondition	Primary containment
	application credential with broad Microsoft 365 consent	(method not disclosed)	confirmed sign-ins; ~1,000 with attempts [7][8]	the app	suspension, customer disables app [7][8]
Tor exit relays	Relay operators' servers	Password reuse, Ebury backdoor, malicious exit policy	333 to 377 relays; 7.46% of exit capacity at peak [9]	Password SSH login, in some cases after accepting an unfamiliar host key	BadExit/MiddleOnly flags via directory authorities [9]

Three observations emerge from this comparison. The first is that, in three of the four incidents, the downstream victims had no vulnerability of their own to patch. Nethemba, one of the firms investigating the UpdraftCentral attacks, reported that the targeted sites it examined were running current WordPress and UpdraftPlus versions that were already patched against a previously disclosed vulnerability [6]. This indicates that ordinary patch hygiene did not protect against an attacker presenting a legitimate key through a legitimate channel. The Tor incident differs: its initial foothold and its user-side impact both depended on password-based authentication, which is the subject of a later section. The second observation is that containment, where it worked quickly, was performed by a central party with the authority to revoke trust for everyone at once: the browser root programs, the Tor directory authorities, and the vendors themselves. The third is that containment was uneven across the population. Parties outside the reach of those central mechanisms were left to discover and remediate exposure on their own.

The Long Tail of Trust Stores

The eMudhra response demonstrates both the strength of the modern Web PKI and its blind spot. Chrome corroborated the report and shipped a CRLSet update blocking the affected intermediates within about two hours, without requiring a browser restart, and pushed an equivalent update to Android devices through Google Play Services [1]. Google also reported that no TLS certificates chaining to the affected intermediates had been logged to Certificate Transparency before the block, which, Google noted, means CT-enforcing clients could not have accepted a forged certificate from these CAs before the block [1]. Apple and Microsoft blocked the certificates on October 19 and 20 respectively, according to the tracking site maintained by Mozilla's Dennis Jackson [4][5]. Google has announced that the emSign ECC Root CA - G3 will be removed from the Chrome Root Store around October 27, 2026, which will break some legitimate certificates chaining to that root [1]. The

scope of exposure is also broader than that single root. Each of the eight keys appears in a certificate cross-signed by Sectigo's AAA Certificate Services, a root that Apple's trust list still includes, so trust stores containing either root are in scope [4][5].

Outside the browser, the picture is considerably more fragmented. Many non-browser TLS clients neither enforce Certificate Transparency nor check revocation, so they continue to trust the compromised intermediates until their bundled trust store is updated [5]. The list of components that required patches is a useful map of where trust actually lives in a typical software estate: Linux distribution ca-certificates packages, NSS, the curl CA bundle, rustls-webpki and webpki-roots, Go, Node.js, Electron, Python's certifi, pip, uv, Composer, Elixir's castore, and Perl's Mozilla::CA, among others [5]. The Go team assigned CVE-2026-111943 and shipped fixes in Go 1.27.3 and 1.26.10 [4]. On macOS, built-in tools such as curl, git, and the system Python read a separate certificate file that was not updated with the browser fixes [5].

Cloudflare's advisory illustrates the operational consequence. Its remediation guidance for customers included updating client software, rebuilding and redeploying container images, reviewing mutual TLS trust stores and uploaded CAs, and validating protection through each application's actual TLS path rather than assuming that a working browser or curl test proves that every service is protected [2]. Cloudflare also warned explicitly against resolving resulting certificate errors by disabling verification, which would remove the protection the updates restore [2]. For organizations running AI agents, inference gateways, and data pipelines, this guidance is especially relevant. In our experience, these workloads are commonly built on Python, Node.js, and Go runtimes with pinned base images, and they make outbound TLS connections to model APIs and tool endpoints. An attacker with a forged certificate and a position on the network path could intercept credentials or tamper with responses on those connections.

The two code-signing intermediates present a separate exposure. Chrome's TLS verifier would not build a path through a code-signing-restricted intermediate, so the browser blocks did not address them [4]. Any environment that uses code-signing trust for allow-listing, driver loading, or agent tool verification should confirm that the affected intermediates are explicitly distrusted on its platforms, rather than relying on TLS-focused remediation.

Remote-Management Keys as Standing Privilege

The UpdraftCentral and ConnectWise incidents suggest that remote-management and SaaS integration keys function, in effect, as a second PKI that few organizations inventory. When a site owner connects a WordPress installation to a cloud management dashboard, or a tenant administrator consents to a backup application, the result is a long-lived credential that authorizes a third party to act with high privilege. In our assessment, such grants frequently outlive the business need that created them, are seldom reviewed, and fall outside the scope of the vulnerability scanning and patch management processes that organizations rely on.

In the UpdraftCentral case, attackers used stolen keys to send signed commands that installed backdoor plugins and created administrator accounts; Patchstack recorded at least 99 malicious plugins under distinct names such as "Site Health Cache" [6]. Remediation was complicated by an implementation defect. According to

Nethemba, the initial plugin update intended to remove UpdraftCentral Cloud keys contained a bug that prevented it from doing so, requiring a further release and manual key deletion by site owners [6]. This sequence illustrates a structural difficulty: when the vendor is the chokepoint, the vendor's revocation mechanism is also a single point of failure.

The ConnectWise incident followed the same pattern at the identity-provider layer. Huntress reported that the abused credential was a client assertion residing in the vendor's own tenant, which meant it was valid against every customer tenant that had consented to the application [7]. After gaining access, the attacker in one tenant used a second ConnectWise application to add itself to the Global Administrator role and added new client secrets to two other applications, with expiration dates in 2028, establishing persistence that would survive revocation of the original credential [7][8]. ConnectWise stated that it had no visibility into customer tenants and could not confirm whether any individual customer was accessed, leaving each customer responsible for its own investigation [8]. A similar structural risk applies to AI agents and copilots that are granted OAuth consent to mailboxes, file stores, and code repositories. An agent platform that relies on a single multi-tenant application credential could present a similar blast radius, and customers could face the same lack of visibility if that credential were compromised.

Hostile Intermediaries and Authentication Design

The Tor incident tests a different assumption: that a network intermediary can be untrusted without harm, provided the endpoints authenticate one another. The attack succeeded only where that endpoint authentication was weak. About a dozen of the malicious exits served a fake SSH host key, so users who had previously connected saw a changed-key warning, while first-time users saw an ordinary prompt to accept a new key; the remaining 300 or so redirected traffic through the same servers without altering the key [9]. Users whose password logins reached the interceptor exposed those passwords, which the interceptor tested for sudo access before attempting to install malware [9]. Key-based SSH logins could not be relayed to the real server, because under RFC 4252 the client's signature covers a session-unique identifier that the interceptor cannot reuse [9][10]. That protects the credential but not the session. As the reporting on the incident notes, the interceptor could still accept a key login itself and show the user a fake shell, capturing anything typed there [9].

This distinction is the most transferable lesson of the incident. Protocols that bind authentication to the specific session, as SSH public-key authentication and properly validated TLS do, limit credential theft through a hostile intermediary, while verified host keys and validated server certificates are what prevent the interception itself. Protocols that transmit a reusable secret, such as a password, after a trust-on-first-use decision provide neither protection. The root cause on the operator side, as reported by one affected relay family, was password reuse across SSH, sudo, and a backup passphrase stored in plaintext, combined with password authentication enabled on an internet-facing server [9]. In other words, the relay network was compromised through the same class of weakness that the attacker then exploited against relay users.

Recommendations

Immediate Actions

Organizations should begin by confirming that every TLS-consuming component, not only browsers, has received an updated trust store that distrusts the eMudhra intermediates. This includes operating system certificates packages, language runtimes and package managers, and any container image whose base layer or bundled CA packages predate the relevant distrust update, regardless of build date; images should be rebuilt and redeployed rather than patched in place [2][5]. Teams can verify protection using the test endpoints and offline chain bundle published on the incident tracking site [5]. Any site or service using a certificate that chains to emSign ECC Root CA - G3 should obtain a replacement from another hierarchy before the scheduled Chrome root removal [1]. Under no circumstances should resulting certificate errors be resolved by disabling verification [2].

WordPress operators using UpdraftPlus should update to version 1.26.13 (free) or 2.26.13 (Premium) or later, the releases that correctly remove cloud keys, delete existing UpdraftCentral connection keys, and review sites for the indicators of compromise published by TeamUpdraft, Patchstack, and Nethemba [6]. Microsoft 365 administrators should disable, rather than delete, the ConnectWise Cloud Backup and related Cloud Management applications until the vendor confirms credential reset, and should review all enterprise applications for client secrets or certificates added since October 13, 2026, the review window Huntress recommends [7]. Anyone who connected to an SSH server over Tor between October 1 and 5 and accepted an unfamiliar or changed host key should change the affected password and check the server for Ebury indicators [9].

Short-Term Mitigations

Over the following weeks, organizations should build an inventory of delegated trust. For PKI, this means identifying every trust store in use, including private trust stores embedded in applications, appliances, and agent frameworks, and establishing an owner and update path for each. For remote management and SaaS integrations, it means enumerating every third-party key, OAuth consent, service principal, and RMM agent with privileged access, and removing those without a current business need. Microsoft Entra app management policies can restrict or cap the lifetime of client secrets, and the number of identities holding Application Administrator roles should be minimized [7].

Monitoring should be extended to the specific persistence techniques observed in these incidents. Audit logs should alert on applications adding themselves to directory roles, on new credentials added to existing applications, and on service principal sign-ins from unexpected infrastructure [7]. For SSH, organizations should disable password authentication on internet-facing hosts, require key-based or certificate-based authentication, and distribute known host keys through configuration management or SSH certificates so that users are never asked to make trust-on-first-use decisions.

Strategic Considerations

The broader implication is that trust concentration should be managed as a risk category in its own right. Third-party risk assessments for RMM, backup, and agent platforms should ask how vendor-held keys and credentials are generated, stored, scoped, and revoked, and whether a single vendor-side secret can authenticate to all customers. Procurement should favor architectures that scope credentials per customer, support customer-held keys, and provide customer-side audit visibility. Organizations that operate their own CAs or HSM-based signing should review key-generation ceremonies for independent verification of key quality. According to eMudhra's preliminary account, weak keys were certified even though generation used FIPS 140-2 Level 3 HSMs [4]. Whether the flaw lay in the HSM or in surrounding tooling has not been publicly established, but the case illustrates the value of verifying key quality independently at generation time.

AI programs warrant similar scrutiny. Although none of the four incidents involved AI systems directly, agent deployments commonly involve the same forms of delegated trust. Agents receive OAuth grants to enterprise data, call tools over TLS through runtime-specific trust stores, and are often managed through vendor-hosted control planes. Security teams should apply the same inventory, scoping, and revocation discipline to agent credentials that these incidents show is necessary for RMM and backup integrations, and should prefer authentication mechanisms that bind to the session so that a compromised intermediary cannot replay what it observes.

CSA Resource Alignment

CSA's [Key Management Lifecycle Best Practices](#) addresses the failure at the center of the eMudhra incident [11]. The guide is organized around the phases of the key lifecycle, including generation and revocation. eMudhra's preliminary account of the root cause, a flaw in a key wrap and unwrap step that yielded weak keys even though generation used certified hardware, concerns the generation phase the guide addresses. While the investigation is ongoing, the case reinforces the value of verifying key quality independently at generation time rather than relying on HSM certification alone.

The CSAI Foundation lab note [AI Developer Ecosystem Concentration: Critical Infrastructure's Hidden Risk](#) develops the concentration-risk thesis that this note applies to trust infrastructure [12]. That analysis frames shared control points in the developer toolchain as sources of systemic blast radius. The October 2026 incidents extend the same reasoning to certificate authorities, vendor-held management keys, and relay operators, and the long list of runtimes and package managers that needed eMudhra patches shows that the developer ecosystem is itself a distribution channel for trust decisions.

[Zero Trust Guidance for Critical Infrastructure](#) provides a framework for treating remote access paths as explicitly governed protect surfaces rather than implicitly trusted channels [13]. Its emphasis on continuous verification and least-privilege access applies directly to RMM and backup integrations, whose standing high-privilege grants run counter to Zero Trust's least-privilege and continuous-verification principles. Finally, the [AI](#)

[Controls Matrix \(AICM\) v1.1](#) maps these recommendations to auditable controls [14]. Its Cryptography, Encryption and Key Management domain covers key generation and revocation, its Identity and Access Management domain covers delegated and application credentials, and its Supply Chain Management, Transparency and Accountability domain covers the third-party assurance questions that vendor-held keys raise.

The CSAI Foundation research note [ShinyHunters' OAuth Pivot: A Year of SaaS Supply-Chain Breaches](#) examines the pattern seen in the ConnectWise incident from another angle [15]. It analyzes vendor-held OAuth and connected-app credentials abused across customer SaaS tenants, and it offers a longer record of the standing-privilege problem that this note identifies in remote-management and backup integrations.

References

- [1] Google Chrome and Android Security Teams. "[Protecting Chrome and Android Users from Compromised eMudhra Intermediate CA Certificates](#)." Google Blog, October 20, 2026.
- [2] Cloudflare. "[8 eMudhra certificates are compromised. Here's what you need to know](#)." Cloudflare Blog, October 20, 2026.
- [3] Simon Sharwood. "[Indian certification authority eMudhra's private keys compromised – one of them was the number '4'](#)." The Register, October 20, 2026.
- [4] Swati Khandelwal. "[Eight eMudhra CA Keys Compromised, Triggering Emergency Fixes Across Browsers and Software](#)." The Hacker News, October 20, 2026 (updated October 21, 2026).
- [5] Dennis Jackson. "[eMudhra Intermediate CA Compromise](#)." Incident tracking site, October 20, 2026.
- [6] Bill Toulas. "[Stolen UpdraftCentral keys used to backdoor WordPress sites](#)." BleepingComputer, October 20, 2026.
- [7] Huntress. "[ConnectWise \(SkyKick\) Vendor Credentials Abused in M365 Attacks](#)." Huntress Blog, October 16, 2026 (updated October 19, 2026).
- [8] Juha Saarinen. "[Stolen ConnectWise 'SkyKick' credential exposed Microsoft 365 tenants](#)." iTnews, October 18, 2026.
- [9] Swati Khandelwal. "[Attackers Hijacked Over 300 Tor Exit Relays to Intercept SSH Connections](#)." The Hacker News, October 21, 2026.
- [10] T. Ylonen and C. Lonvick. "[The Secure Shell \(SSH\) Authentication Protocol \(RFC 4252\)](#)." IETF, January 2006.
- [11] Cloud Security Alliance. "[Key Management Lifecycle Best Practices](#)." CSA Cloud Key Management Working Group, 2023.
- [12] CSAI Foundation. "[AI Developer Ecosystem Concentration: Critical Infrastructure's Hidden Risk](#)." CSA Lab Space, 2026.
- [13] Cloud Security Alliance. "[Zero Trust Guidance for Critical Infrastructure](#)." CSA, 2024.
- [14] Cloud Security Alliance. "[AI Controls Matrix \(AICM\) v1.1](#)." CSA, 2025.
- [15] CSAI Foundation. "[ShinyHunters' OAuth Pivot: A Year of SaaS Supply-Chain Breaches](#)." CSA Lab Space, July 2026.