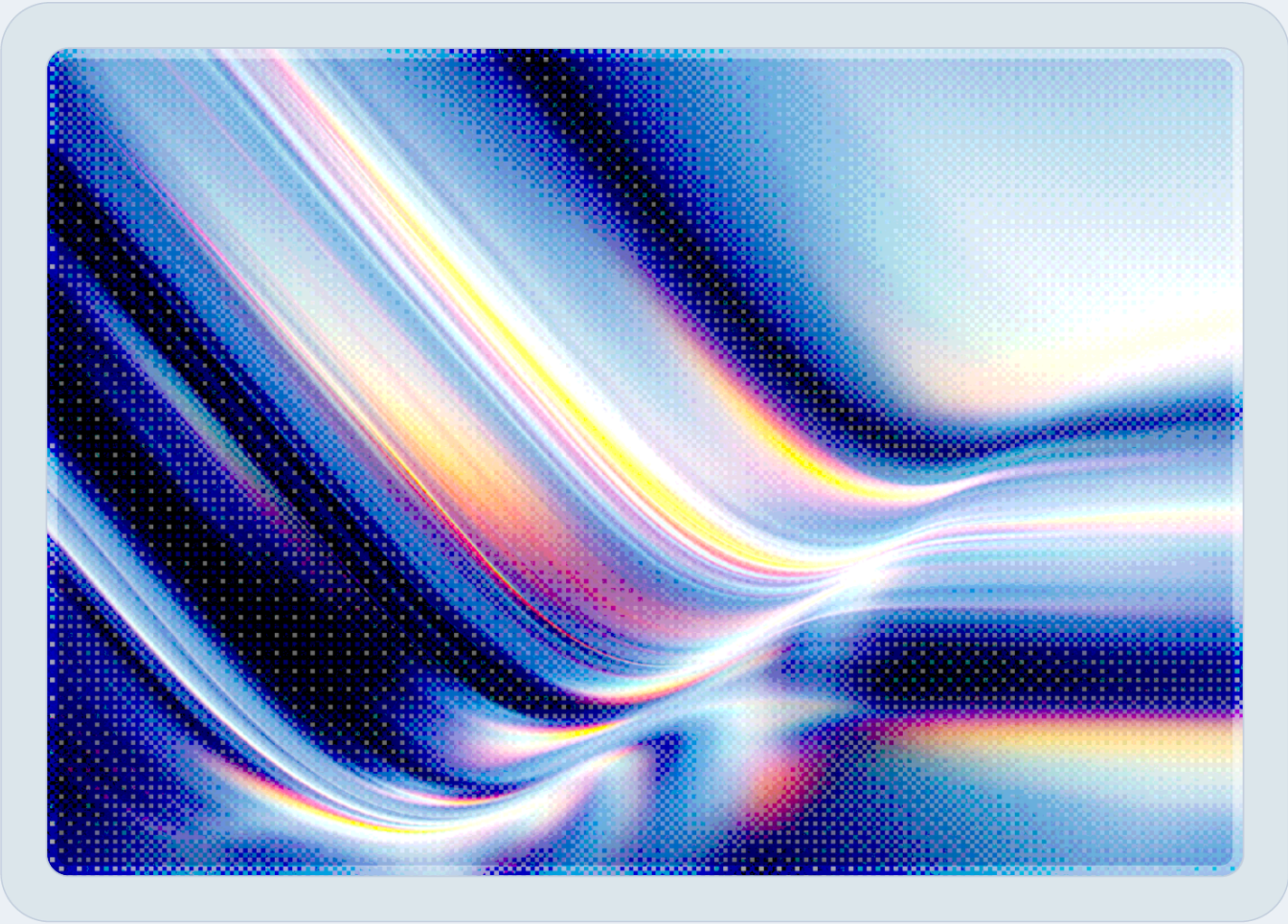
CSAI Foundation | Cloud Security Alliance

Gitea CVE-2026-27771: Private Container Registry Exposure

~Four-Year Access Control Flaw Affects 30,000+ Self-Hosted Deployments

2026-05-28

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- CVE-2026-27771 (CVSS 8.2 [4]) affects all Gitea versions that include the built-in container registry, prior to 1.26.2, allowing unauthenticated remote actors to pull private container images through a fundamental access control failure in the platform's built-in OCI registry.
- NoScope, a UK-based security firm, identified approximately 31,750 internet-facing Gitea instances as likely affected across more than 30 countries; the flaw persisted undisclosed for approximately four years since the container registry feature was introduced.
- Forgejo, the widely used community fork of Gitea, shares the same container registry codebase and has been independently confirmed as vulnerable by the discovering researchers.
- Container images routinely contain credentials, API keys, TLS certificates, production configuration, and compiled source code; unauthorized registry read access provides an attacker with a comprehensive reconnaissance capability that can materially accelerate subsequent attacks.
- Organizations running Gitea or Forgejo should update to Gitea v1.26.2 immediately; where upgrade is not feasible in the short term, setting `[service].REQUIRE_SIGNIN_VIEW=true` in the Gitea configuration enforces authentication globally as a temporary mitigation.

Background

Gitea is a self-hosted, open-source platform for Git repository management, offering code review, issue tracking, integrated CI/CD pipeline support, and package registries including an OCI-compliant container registry. Its lightweight architecture and permissive MIT licensing position it as a common alternative to commercial repository hosting for organizations seeking to retain intellectual property entirely within controlled infrastructure. Gitea is commonly adopted in sectors where data sovereignty or regulatory constraints make third-party hosted platforms impractical, including healthcare, aerospace, defense contracting, financial services, and telecommunications [5].

The container registry feature, integrated into Gitea roughly four years before this disclosure, allows development teams to store and distribute Docker and OCI container images alongside source code in the same platform. This colocation model eliminates the operational overhead of running separate registry infrastructure and has proven particularly attractive to smaller engineering teams, government agencies, and regulated-sector enterprises that need private internal distribution without introducing additional external dependencies. The implicit security assumption underlying this deployment model is that images stored in private repositories remain inaccessible to the outside world—an assumption that CVE-2026-27771 invalidates entirely for all affected versions.

The vulnerability was identified by NoScope, a UK-based security research and penetration testing firm, through its autonomous penetration testing agent. Using Shodan to enumerate internet-facing Gitea instances carrying the platform's default identifiers, the agent methodically tested registry access controls and confirmed that private container images could be retrieved without authentication [1]. NoScope responsibly disclosed the findings to the Gitea maintainer team, which assigned CVE-2026-27771, attributed the discovery to NoScope, and released the remediation in Gitea v1.26.2 on May 20, 2026 [2][7]. Public disclosure followed on May 25, 2026.

Security Analysis

The Access Control Failure

CVE-2026-27771 is an access control vulnerability assigned a CVSS score of 8.2 [4]. The core defect lies in how Gitea's container registry evaluated repository visibility. When a Gitea repository was marked private, the platform's web interface and native Git operations correctly enforced authentication requirements. However, the OCI registry API operated under a separate code path that did not consult the repository visibility state before serving content. As a result, the registry endpoint honored anonymous requests for image manifests and layer blobs from repositories explicitly designated as private, returning the full image content without issuing an authentication challenge [3][4].

An attacker exploiting this vulnerability requires no Gitea account, no password, and no prior knowledge of the organization beyond the reachable hostname of the Gitea instance. A standard `docker pull` command—or any OCI-compatible client—directed at the target instance's registry endpoint is sufficient to retrieve images from private repositories. There is no evidence of active exploitation in the wild reported at time of disclosure, but given the four-year exposure window and the ease of exploitation, the possibility that prior unauthorized access occurred cannot be dismissed [1][4].

The Gitea v1.26.2 patch addresses the immediate access control failure and adds UI indicators that surface package visibility state more clearly to administrators. The maintainers have noted that the underlying permissions architecture for packages represents a longer-term engineering effort that extends beyond this single patch [2]. Organizations should verify access enforcement behavior on upgraded instances rather than treating the patch as a complete architectural resolution.

What Container Images Reveal

The risk profile of this vulnerability is substantially elevated by the typical contents of private container images. A container image represents a complete snapshot of an execution environment, and the build practices common across modern software development mean that sensitive materials are commonly packaged into images that development teams intend to remain strictly internal [3]. A single private container image may contain compiled or interpreted application source code, a complete dependency graph pinned to specific versions, configuration files referencing production database connection strings and internal service endpoints, API keys and service account tokens embedded at build time, TLS certificates and private keys bundled for inter-service communication, and environment variable defaults populated with credentials inherited from the build environment [3][4].

For compiled applications or those where source code is not separately exposed, the intelligence value of a container image to an attacker can substantially exceed that of source code alone, given the inclusion of runtime credentials, dependency graphs, and infrastructure topology. Static analysis of a container image's filesystem can reveal business logic, authentication mechanisms, network topology, infrastructure naming conventions, and the full dependency supply chain of an application—capabilities that would otherwise require significant compromise depth to obtain. For organizations building AI and machine learning pipelines, the risk is compounded: images in these environments may additionally contain model weights, training dataset references, inference configurations, and references to proprietary cloud services or data sources. Unauthorized read access to a private container image can provide reconnaissance value comparable in scope to that of a targeted intrusion, particularly where images embed credentials and infrastructure topology that would otherwise require active compromise to obtain.

Scope and Population at Risk

NoScope's research identified approximately 31,750 internet-facing Gitea instances as likely vulnerable at the time of discovery, distributed across more than 30 countries [1]. The highest concentrations were reported in China, the United States, Germany, France, and the United Kingdom [4]. The affected organizations span healthcare providers, aerospace manufacturers, retail infrastructure operators, internet service providers, and enterprise software development teams across both private and public

sector environments [4][5]. Because the flaw has been present since the container registry feature was first introduced, any organization that used Gitea's registry to store private images during that window should consider the possibility of prior exposure when assessing incident response obligations.

Forgejo, the community fork of Gitea, maintains its own release cycle but shares the same container registry implementation. NoScope independently confirmed Forgejo is affected by the same vulnerability through direct testing [4]. At the time of this writing, Forgejo had not yet released a corresponding patch; Forgejo users should monitor the project's security announcement channel on Codeberg [8] and apply the Gitea configuration workaround—or equivalent network-layer restrictions—as an interim measure.

The detection challenge for affected organizations is meaningful. Because the Gitea container registry served image content to anonymous requests as a normal code path, server access logs will not distinguish legitimate pulls from unauthorized access without substantial enrichment. Organizations assessing their exposure should review registry access logs for unexpected source IP addresses and geographic regions, particularly against repositories containing high-value artifacts. Where log retention is insufficient to reconstruct the full exposure window, any credentials or keys embedded in container images built over the past four years should be treated as potentially compromised for the purpose of risk assessment.

Discovery by Autonomous Penetration Testing

The method of discovery carries significance beyond this specific vulnerability. The NoScope autonomous penetration testing agent performed what amounts to a systematic, large-scale access control audit across tens of thousands of publicly reachable Gitea instances—a scope of testing that would be impractical to conduct manually. This reflects a broader shift in the vulnerability research landscape: automated agents can now enumerate and test self-hosted infrastructure at internet scale, meaning that access control misconfigurations in widely deployed platforms are less likely to remain undiscovered for extended periods, particularly among well-resourced and motivated actors. Organizations running self-hosted development infrastructure should assume that such systematic testing by both researchers and threat actors conducting similar scanning opportunistically is ongoing, and should treat rapid patching cadences for internet-exposed platforms as an operational baseline rather than an aspirational goal.

Recommendations

Immediate Actions

The highest-priority action for any organization running Gitea is to update to version 1.26.2. The release is available through the official Gitea release channel [2]. For organizations that cannot complete an upgrade in the immediate term, the configuration workaround of setting `[service].REQUIRE_SIGNIN_VIEW=true` in the Gitea `app.ini` configuration file enforces authentication across all content served by the instance, including the container registry. This setting is unsuitable for deployments where certain repositories or registries are intentionally public, as it enforces authentication universally and will break anonymous access to publicly designated resources.

Organizations that have used Gitea's container registry for private images should initiate a credentials rotation exercise. API tokens, service account keys, database passwords, TLS certificates, and other sensitive material present in images stored on the instance during the vulnerability's exposure window should be rotated as a precautionary measure, with priority given to credentials that provide access to production systems, cloud infrastructure, or external services.

Short-Term Mitigations

Security teams should audit container images stored in affected registries to identify which images contain embedded credentials or sensitive configuration. Open-source static analysis and secret scanning tools designed for container image filesystems can assist in prioritizing which images present the highest exposure risk. This audit should inform both the credentials rotation scope and any necessary notifications to affected stakeholders. Teams should also assess whether container build pipelines are injecting secrets at build time—the most common cause of credentials appearing in image layers—and migrate toward runtime secret injection mechanisms such as Kubernetes Secrets, HashiCorp Vault, or equivalent, which prevent credentials from being baked into distributable artifacts.

Where feasible, organizations should restrict inbound network access to Gitea's registry API endpoints to documented, authorized source CIDR ranges. This network-layer control reduces exposure to any residual vulnerabilities and provides a defense-in-depth complement to the application-layer access control fix delivered in v1.26.2. Registry access policies should be reviewed to ensure they conform to the principle of least privilege, with pull access limited to authenticated identities that have a documented and current operational need.

Strategic Considerations

CVE-2026-27771 illustrates a systemic risk pattern in integrated development platforms: sub-systems that operate across different trust boundaries may enforce access controls independently and inconsistently. In this case, Gitea's repository access controls functioned correctly at the Git and web layers while the container registry operated under a different enforcement model. Organizations running integrated development platforms—regardless of vendor—should explicitly scope vulnerability management programs to include all platform sub-systems: registry endpoints, package management APIs, CI/CD orchestrators, and webhook receivers. These components may present independent attack surfaces that are not covered by conventional web-application vulnerability scanning profiles.

The four-year undisclosed lifespan of this vulnerability should also inform how organizations approach access control validation in self-hosted platforms. Functional testing that verifies a "private" designation behaves as expected from the perspective of an unauthenticated external requester—not merely from the authenticated user's perspective—is a straightforward verification step that would have surfaced this class of defect. Container registry access controls, package registry permissions, and artifact repository visibility settings should be included in periodic security validation exercises as discrete test cases.

CSA Resource Alignment

CVE-2026-27771 maps directly to the Identity and Access Management domain of the CSA Cloud Controls Matrix, particularly IAM-01 (Identity and Access Management Policy and Procedures) and IAM-09 (User Access Management), which require that access to cloud-hosted systems and data stores be enforced through verified identity and that access control configurations be subject to periodic correctness review. Gitea operators assessing their control posture should include container and package registry endpoints in the scope of IAM domain audits, not only source code repository access.

CSA's published container security guidance—including the *Best Practices for Implementing a Secure Application Container Architecture* [6] and *Challenges in Securing Application Containers and Microservices*—addresses the full lifecycle of container security risk, from registry access controls and image provenance verification through runtime isolation and incident response. These publications provide actionable control frameworks for self-hosted registry operators evaluating their posture in light of this disclosure and developing longer-term improvements to their container security programs.

CSA's AI Controls Matrix (AICM) is directly relevant for organizations using Gitea registries as part of AI and machine learning development pipelines. AICM controls addressing supply chain integrity and data protection at the model artifact layer apply to container images that carry model weights, inference

configurations, or training pipeline components. The unauthorized retrieval of such images is a supply chain integrity event, and organizations should evaluate their AICM control coverage for artifact storage and distribution systems.

The MAESTRO framework for agentic AI threat modeling addresses deployment infrastructure risks at Layer 3, including unauthorized access to model artifacts and inference configurations. Organizations building agentic AI systems that rely on self-hosted container registries for model distribution should explicitly map CVE-2026-27771-class risks—unauthenticated artifact retrieval from private registries—into their MAESTRO threat models and ensure that compensating controls are in place at the network and application layers.

CVE-2026-27771 illustrates a class of access control failure that Zero Trust architectures are specifically designed to prevent. Where Zero Trust principles are applied, authentication and authorization are verified at every request boundary with no inherited trust between platform layers; Gitea's registry operated on an implicit-trust model that the Zero Trust approach explicitly rejects. Security architects evaluating integrated development platforms should apply Zero Trust validation methodology to verify that access control enforcement is consistent across all exposed endpoints, not solely those subject to routine user interaction.

References

[1] NoScope. "Gitea Instances Exposing Private Container Images." NoScope Blog, May 25, 2026.

[2] Gitea. "Gitea 1.26.2 is Released." Gitea Blog, May 20, 2026.

[3] Orca Security. "Gitea Container Registry Exposes Private Images to Unauthenticated Attackers." Orca Security Blog, May 2026.

[4] The Hacker News. "Gitea Vulnerability Exposes Private Container Images without Authentication." The Hacker News, May 27, 2026.

[5] TechTimes. "Gitea Flaw Left 30,000 Deployments' Private Container Images Readable for 4 Years." TechTimes, May 27, 2026.

[6] Cloud Security Alliance. "Best Practices for Implementing a Secure Application Container Architecture." CSA, 2019.

[7] Gitea. "Security Advisories." Gitea, accessed May 2026.

[8] Forgejo. "Security Announcements." Codeberg, accessed May 2026.