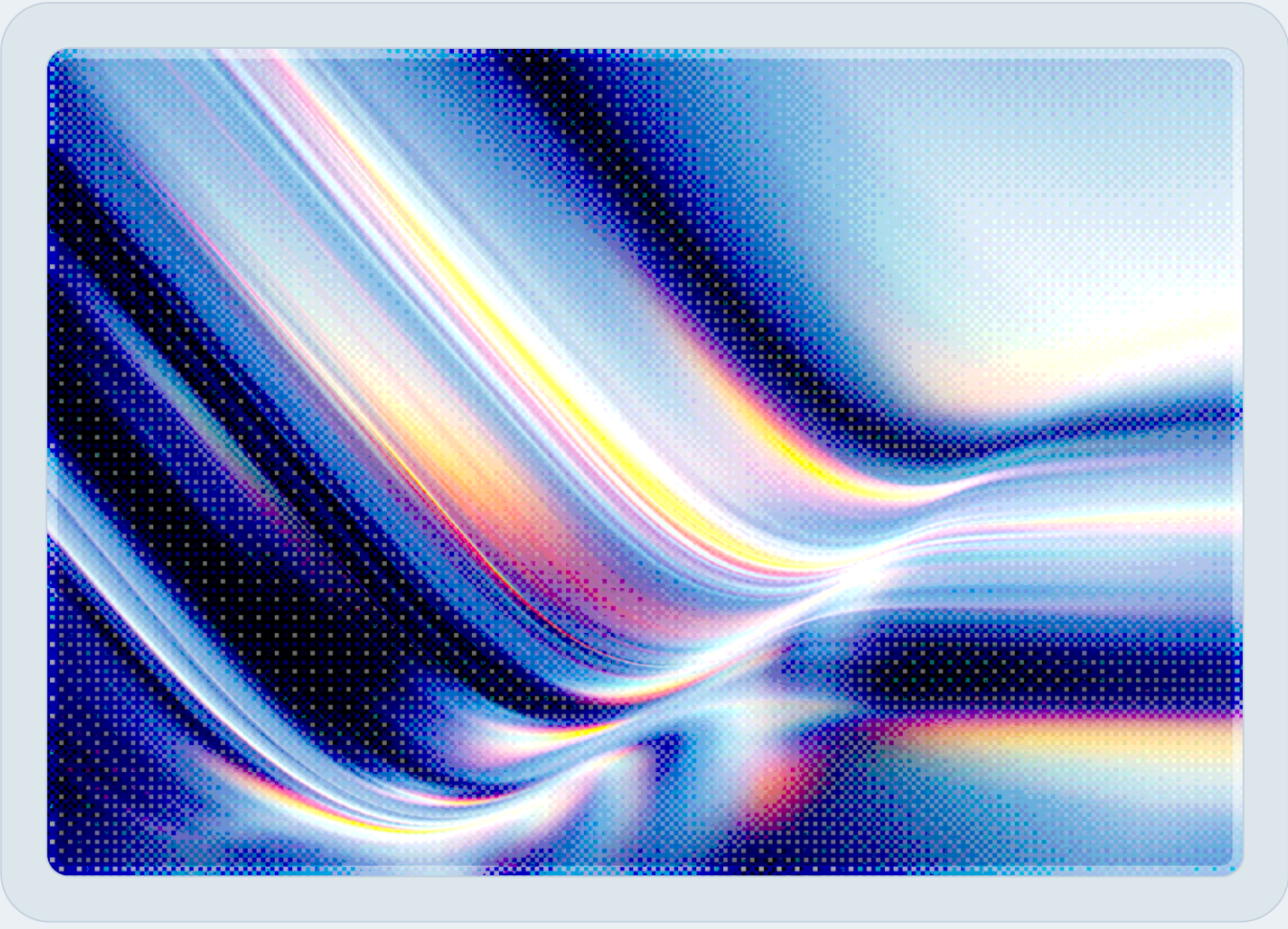
CSAI Foundation | Cloud Security Alliance

Three JFrog Artifactory Flaws Chained for Admin Takeover

2026-09-18

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

Wiz Research has documented sustained in-the-wild exploitation of three JFrog Artifactory vulnerabilities — CVE-2026-42018, CVE-2026-42016, and CVE-2026-82329 — between August 15 and September 8, 2026, with attackers converting unauthenticated network access into full administrative control in as little as two HTTP requests [1][2]. Two of the flaws, CVE-2026-42018 (CVSS 7.5) and CVE-2026-42016 (CVSS 8.1), can be chained together: the first leaks an internal anonymous-user access token even when anonymous access is disabled, and the second lets that low-privilege token be exchanged for one carrying full administrator scope [1][6]. The third, CVE-2026-82329 (CVSS 9.8, CWE-287), stands alone as a critical authentication bypass rooted in a default configuration flaw that makes Artifactory's cluster-join signing key mathematically derivable by anyone, letting an unauthenticated attacker forge a token that JFrog Access accepts as a legitimate, permanent administrator credential [4][5]. The U.S. Cybersecurity and Infrastructure Security Agency added CVE-2026-82329 to its Known Exploited Vulnerabilities catalog in early September 2026, urging remediation by September 5, underscoring how quickly exploitation followed disclosure [7]. Once inside, attackers have deployed a layered set of malicious Groovy plugins and a custom Rust-based backdoor with command-and-control capabilities, created persistent administrator accounts, and in some cases stolen the LDAP credentials, user databases, and cryptographic signing keys stored inside compromised instances [1][3]. Six weeks after JFrog's disclosures, Wiz found that vulnerable-instance rates had fallen only modestly — to 59, 62, and 49 percent for the three flaws respectively — and, critically, patching alone does not remove attacker-created admin accounts or stolen credentials, meaning many organizations that believe they have closed the hole are still hosting a live backdoor [1][8].

Background

JFrog Artifactory functions as the central binary and package repository for a large share of enterprise software delivery pipelines, storing build artifacts, container images, and dependency packages that downstream applications pull from continuously. That central position is precisely what makes authentication flaws in Artifactory so consequential: an attacker who gains administrative control of an organization's artifact repository does not merely access one system, but gains a foothold from which to poison the software supply chain feeding every application, service, and deployment pipeline that depends on it. Wiz Research's investigation into active exploitation, published in coordination with

reporting from The Hacker News, SecurityWeek, and BleepingComputer, found that threat actors recognized this leverage quickly and moved from vulnerability disclosure to organized post-exploitation activity within days [1][2][3][9].

The two chainable vulnerabilities were disclosed and patched separately over the summer: CVE-2026-42016 was fixed on July 27, 2026, and CVE-2026-42018 followed on August 12, 2026, both affecting self-hosted Artifactory deployments across a wide range of the 7.111.x through 7.161.x release branches [1][6]. CVE-2026-82329, the more severe standalone flaw, was disclosed and patched on August 28, 2026 [4]. Wiz's telemetry indicates that exploitation of the chained pair began around August 15 and continued through September 8, while exploitation of the critical standalone flaw began within days of its own disclosure — a pattern consistent with the increasingly compressed window between vulnerability publication and mass scanning that has become typical for internet-facing enterprise infrastructure [1][8]. At the time of initial disclosure, Wiz estimated that 67 to 69 percent of scanned organizations ran a vulnerable Artifactory version; six weeks later, that figure had fallen only to a range of 49 to 62 percent depending on which of the three flaws is measured, with the critical CVE-2026-82329 seeing the fastest remediation given its severity rating [1].

The root causes of these flaws differ but share a common theme: each involves Artifactory's internal token- and key-handling logic failing to distinguish between "no value" and "an empty or default value" in a way that a security boundary depends on. CVE-2026-42018 allows a caller who has never authenticated to retrieve an internal token normally reserved for Artifactory's own anonymous-user identity, by requesting a specific access-token endpoint with a trailing-slash variant that bypasses the check meant to block that retrieval [1]. CVE-2026-42016 then compounds the problem: when that anonymous token — or any low-privilege token — is presented to Artifactory's token-exchange endpoint, the service verifies the token's cryptographic signature and issuer but does not verify that the requested scope of the new token is actually permitted for the presenter, allowing a token upgrade from anonymous to full administrator [1][6]. Security researchers documented cases where the entire sequence, from initial unauthenticated request to a freshly minted administrator account, completed in under five minutes [2].

CVE-2026-82329 has a more specific, and in some respects more severe, root cause. Bishop Fox's technical analysis found that on a default Artifactory installation where the `additional-join-keys` cluster configuration setting is left unset, the underlying lookup returns an empty string rather than a null value that later validation logic would reject. That empty string passes through several successive checks that examine only whether a lookup technically succeeded, not whether the resulting value is meaningful, with the effect that Artifactory's cluster-join signing secret becomes a fixed, publicly computable value: a key ID equal to the SHA-256 hash of an empty string, and a signing secret consisting of thirty-two bytes of standard PKCS7 padding [4]. Because the endpoint that accepts cluster-join requests is intentionally left unauthenticated — legitimate joining nodes have no user identity

of their own yet — any attacker can compute the predictable signing secret offline, forge a JSON Web Token, and submit it to that endpoint, which returns a permanent, unrestricted Artifactory Access administrator credential with no expiration [4][5]. GitHub's Advisory Database classifies CVE-2026-82329 as CWE-287 (Improper Authentication) with a CVSS base score of 9.8, reflecting an attack that requires no privileges, no user interaction, and results in a complete loss of confidentiality, integrity, and availability [5].

Security Analysis

The practical impact of these vulnerabilities extends well beyond credential theft, because administrative access to Artifactory grants attackers a code-execution surface through the platform's own plugin architecture. Wiz's research identified six distinct malicious Groovy plugins deployed across compromised instances, organized into three tiers of increasing sophistication. The first tier — files such as `rce.groovy`, `cmd.groovy`, and `jwtToken.groovy` — provides straightforward remote command execution and token-minting capability and is designed to keep working even after the underlying authentication vulnerability is patched, since the plugin itself, once installed, does not depend on the original flaw to function [1]. A second-tier plugin, `JFrogImage.groovy`, is purpose-built for credential and secret harvesting: it operates in multiple modes that allow an attacker to export Artifactory's configuration, dump user account databases, steal the cryptographic keys Artifactory uses to sign Access tokens, or simply execute arbitrary commands, giving a single implant broad reconnaissance and persistence utility [1]. The most advanced tier includes `metrics.groovy` and `httpSession.groovy`, which inject custom servlet filters directly into Artifactory's running Java process to create in-memory backdoors that support network tunneling and that actively truncate logs to frustrate forensic reconstruction; the `metrics.groovy` filter notably remains dormant unless a request arrives bearing a specific hashed value in a custom HTTP header, a design consistent with an intent to evade both automated scanning and casual log review [1].

Beyond the Groovy plugins, multiple attackers deployed a custom-written backdoor implemented in Rust that establishes command-and-control connectivity independent of Artifactory's own process space, a choice that likely complicates both static detection and takedown, since a Rust binary can be dropped and executed with no dependency on the Artifactory application itself [1][8]. Attackers also created new administrator accounts using naming patterns that fell into two broad categories: some used conspicuously artificial names such as `0xterror` or randomly generated strings following a `svc_[a-zA-Z0-9]{8}` pattern, apparently indifferent to detection, while others chose names designed to blend into legitimate infrastructure, such as `jfrog-distribution` or `repo-`

`service`, suggesting an intent toward longer-term, quieter persistence [1][2]. In several observed cases, attackers also attached their own SSH public keys to newly created or hijacked accounts, providing a persistence mechanism entirely independent of Artifactory's own authentication stack [3].

A notable finding, previously noted by CSA's own initial analysis of this campaign, is a gap in how Artifactory's audit logging records activity performed with a token derived from the anonymous-user identity: privileged actions taken through the CVE-2026-42018/CVE-2026-42016 chain are recorded in Artifactory's logs simply as `token:anonymous`, without any distinguishing detail that would let a defender separate legitimate anonymous traffic from an attacker who has silently escalated that identity to administrator [1]. This means organizations reviewing their own logs for signs of compromise cannot rely on account names alone; they must instead look for behavioral anomalies, such as an anonymous-attributed token performing account creation, plugin installation, or configuration export, none of which anonymous access should ever legitimately perform. WatchTowr's independent honeypot telemetry corroborates this reconnaissance pattern at scale, observing attackers systematically enumerating users, groups, credential stores, and federated identity configurations once administrative tokens were obtained — behavior consistent with attackers casing an environment for lateral movement opportunities rather than opportunistic, one-off compromise [6].

Perhaps the most consequential operational finding from this campaign is that patching the underlying authentication flaws does not remediate an already-compromised instance. Because the attack's persistence mechanisms — rogue administrator accounts, planted Groovy plugins, stolen signing keys, and independently established Rust backdoors — do not depend on the vulnerable code path remaining open, an organization that applies JFrog's fix but does not separately audit for post-exploitation artifacts will patch the front door while leaving an attacker in full possession of a key. Wiz's data showing that a substantial share of previously vulnerable instances remain exposed weeks after disclosure compounds this risk: any instance patched late, or patched without a corresponding compromise assessment, should be treated as a candidate for prior compromise rather than a resolved incident [1][8].

Recommendations

Immediate Actions

Organizations running self-hosted JFrog Artifactory should confirm their current version against JFrog's fixed releases without delay: 7.111.21 or later, 7.117.28 or later, 7.125.20 or later, 7.133.29 or later, 7.146.38 or later, or 7.161.20 or later, any of which resolves all three vulnerabilities [1][6]. Patching must be paired with an active compromise assessment rather than treated as sufficient on its own, given that none of the persistence mechanisms observed in this campaign depend on the vulnerable code remaining reachable.

That assessment should include a full audit of the Artifactory administrator account list for any account the organization cannot positively attribute to a known administrator or automated service, a search of installed plugins for unrecognized Groovy scripts (particularly any matching the `rce.groovy`, `cmd.groovy`, `jwtToken.groovy`, `JFrogImage.groovy`, `metrics.groovy`, or `httpSession.groovy` naming pattern documented by Wiz), and a review of SSH keys attached to service and administrator accounts [1]. Any instance that was internet-reachable and running an affected version at any point between mid-July and late August 2026 should be assumed compromised until the review proves otherwise, and organizations should rotate Artifactory's Access signing keys, LDAP bind credentials, and any secrets stored in Artifactory configuration as a precaution regardless of what the review finds [1][3].

Short-Term Mitigations

Organizations should review Artifactory access logs specifically for the behavioral signatures Wiz has published: an unauthenticated request to an access-token endpoint returning a 401 followed shortly by a 200 on a related variant, tokens attributed to `token:anonymous` performing privileged actions such as user creation or plugin management, and any successful request to the cluster-join endpoint that was not initiated by the organization's own infrastructure [1]. Because on-instance logs can themselves be tampered with or truncated by the resident Groovy implants described above, this review should be cross-referenced against independent telemetry — network flow logs, reverse-proxy access logs, or SIEM ingestion that predates any suspected compromise — rather than relying solely on Artifactory's own audit trail. Network-level restrictions limiting Artifactory management and API endpoints to known administrative source ranges reduce the population of systems that can reach these endpoints and should be applied as a defense-in-depth measure even after patching, since they narrow the blast radius of any future authentication-layer flaw in the same product family.

Strategic Considerations

This campaign demonstrates that artifact and package repositories occupy a structurally privileged position in the software supply chain, and that authentication failures in that layer propagate risk to every downstream system that trusts the repository's contents. Organizations should treat Artifactory, and any comparable centralized build-artifact or container-registry platform, with the same access-control rigor applied to identity providers and code-signing infrastructure, including routine review of administrator account rosters, mandatory multi-factor authentication for human administrative access, and monitoring specifically tuned to detect privilege escalation rather than only unauthorized access. The recurrence of token-scope validation failures — where a system correctly verifies a credential's authenticity but not its intended authority — as the root cause of two of these three flaws also argues for organizations to ask

their DevOps tooling vendors directly whether token issuance and exchange logic is independently scoped and tested, rather than assuming that signature validation alone constitutes sufficient authorization control.

CSA Resource Alignment

CSA's own JFrog Artifactory Token-Chaining Flaws Enable Backdoors [10], published September 11, 2026, is the most directly relevant prior CSA analysis of this exact campaign and first identified the `token:anonymous` logging gap discussed above; this note extends that analysis with the confirmed CVSS and CWE classification for CVE-2026-82329, the technical root cause of its empty-key derivation flaw, WatchTowr's independent honeypot corroboration, and the finding that patching does not remove attacker-created persistence. Readers responding to this incident should treat the two notes as complementary rather than duplicative. CSA's LiteLLM AI Gateway: KEV-Listed Attack Chain Enables Full Takeover [12] documents a structurally similar pattern in a different product category — chained, KEV-listed CVEs escalating from limited access to full server takeover with credential exfiltration — and the detection and response guidance CSA developed there, particularly around treating any KEV-listed authentication flaw as requiring compromise assessment rather than patch-and-close, applies directly to the Artifactory case. CSA's Sapphire Sleet Poisons Mastra AI npm Supply Chain [13] is relevant because it examines a related downstream risk in the software supply chain: in that campaign, attackers compromised a maintainer's publishing credentials to hijack an npm scope and backdoor 145 packages within an 88-minute window, illustrating that whoever controls publishing rights over a package scope or artifact repository is positioned to backdoor every package or build artifact it distributes. Organizations that consume artifacts from a potentially compromised Artifactory instance should apply the same package-integrity verification practices CSA recommended in that analysis. Finally, this incident maps to the Threat and Vulnerability Management and Application and Interface Security domains of the CSA AI Controls Matrix (AICM) v1.1 [11], which set expectations for authenticated access control, credential lifecycle management, and vulnerability remediation timeliness in the DevSecOps toolchains that increasingly underpin AI development pipelines.

References

[1] Wiz Research. "Artifactory Under Attack: In-the-Wild Exploitation of CVE-2026-42016, CVE-2026-42018 & CVE-2026-82329." Wiz Blog, September 2026.

[2] The Hacker News. "Attackers Chain JFrog Artifactory Flaws to Gain Admin Control and Plant Backdoors." The Hacker News, September 2026.

[3] SecurityWeek. "Three JFrog Artifactory Flaws Exploited for Backdoor Deployment." SecurityWeek, September 2026.

[4] Bishop Fox. "CVE-2026-82329: Unauthenticated Administrative Access in JFrog Artifactory via an Empty Cluster Join Key." Bishop Fox Blog, 2026.

[5] GitHub Advisory Database. "CVE-2026-82329: JFrog Artifactory Authentication Weakness." GitHub, August 2026.

[6] GitHub Advisory Database. "CVE-2026-42016: Incorrect Authorization Validation of User Token in JFrog Artifactory." GitHub, July 2026.

[7] Qualys ThreatPROTECT. "CISA Added JFrog Artifactory Vulnerability to its Known Exploited Vulnerabilities Catalog (CVE-2026-82329)." Qualys ThreatPROTECT, September 3, 2026.

[8] The Register. "More JFrog Artifactory bugs under attack, and all 3 have patches." The Register, September 11, 2026.

[9] BleepingComputer. "Artifactory flaws chained in attacks deploying backdoor malware." BleepingComputer, September 2026.

[10] Cloud Security Alliance. "JFrog Artifactory Token-Chaining Flaws Enable Backdoors." Cloud Security Alliance Labs, September 11, 2026.

[11] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, 2026.

[12] Cloud Security Alliance. "LiteLLM AI Gateway: KEV-Listed Attack Chain Enables Full Takeover." Cloud Security Alliance Labs, 2026.

[13] Cloud Security Alliance. "Sapphire Sleet Poisons Mastra AI npm Supply Chain." Cloud Security Alliance Labs, 2026.

