INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-27 06:39 UTC

CVE-2026-39832: SSH Agent Key Constraint Bypass in golang.org/x/crypto Affects Microsoft Azure Linux Docker Buildx

CVE VULNERABILITY | **CRITICAL** | CVSS 9.1

SCC Item ID	SCC-CVE-2026-0228
Type	CVE Vulnerability
CVE ID	CVE-2026-39832
Severity	CRITICAL
CVSS Base Score	9.1
EPSS Score	0.0003 (9th percentile)
Affected Products	Microsoft azl3 docker-buildx 0.14.0-11 on Azure Linux 3.0 (golang.org/x/crypto/ssh/agent)
Published	2026-05-27T01:40:27
Discovery Source	Msrc Patch Tuesday

Executive Summary

A critical flaw in the SSH agent component of golang.org/x/crypto silently drops key-level restrictions, such as use-confirmation prompts and lifetime limits, when forwarding SSH keys, leaving those keys available to anyone who can intercept or influence the forwarded agent session. Microsoft's azl3 docker-buildx 0.14.0-11 package on Azure Linux 3.0 is the confirmed affected distribution, disclosed as part of May 2026 Patch Tuesday. Organizations using Azure Linux 3.0 container build pipelines with SSH agent forwarding enabled are exposed to unauthorized lateral movement using credentials that should have been constrained.

Technical Analysis

CVE-2026-39832 (CVSS 9.1, CWE-281: Improper Preservation of Permissions) is a permission-stripping defect in golang.org/x/crypto/ssh/agent. When SSH keys carrying agent constraints, such as ssh.ConstraintLifetime or ssh.ConstraintExtension requiring per-use confirmation, are forwarded through the agent, constraint metadata is stripped and those constraints are not preserved in the forwarded session. The receiving agent holds the key with no restrictions, violating the policy the key owner intended. Affected package: Microsoft azl3 docker-buildx 0.14.0-11 on Azure Linux 3.0. MITRE techniques: T1552.004 (Unsecured Credentials: Private Keys) and T1563.001 (Remote Service Session Hijacking: SSH Hijacking). EPSS score of 0.0003 (8.83rd percentile)

indicates low current exploitation probability, but the architectural nature of the flaw makes it high-value for targeted post-compromise activity. No CISA KEV listing as of data timestamp. The vulnerability was disclosed alongside CVE-2026-39833 in the MSRC May 2026 CVRF feed. NVD detail record: https://nvd.nist.gov/vuln/detail/CVE-2026-39832 (T1 source, verify currency). MSRC advisory: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2026-39832 (T1 source, verify currency).

Action Checklist

1. Step 1: Containment, Identify all Azure Linux 3.0 hosts running docker-buildx version 0.14.0-11 (azl3 package). Disable SSH agent forwarding (ForwardAgent no) in SSH client configs and Docker build pipeline configurations immediately until the patched package is available and deployed. Document the temporary restriction as a remote access configuration change.
2. Step 2: Detection, Query your asset inventory for any host running azl3 docker-buildx 0.14.0-11 on Azure Linux 3.0. Review SSH daemon logs and Docker build logs for agent-forwarded sessions: look for AuthorizedKeysFile hits combined with build job executions. Check for SSH_AUTH_SOCK environment variables in Docker build container contexts, their presence confirms agent forwarding was active.
3. Step 3: Eradication, Monitor the Microsoft Azure Linux 3.0 package repository for the updated azl3 docker-buildx package. Once released, apply the update and confirm the installed version supersedes 0.14.0-11. Validate the update was applied via automated patch management tooling and log the remediation event.
4. Step 4: Recovery, After patching, re-enable SSH agent forwarding only where operationally required and confirm key constraints are preserved end-to-end by testing with a constrained key (lifetime-limited or confirmation-required). Rotate any SSH private keys used in affected build pipelines to limit the window of exposure for unguarded keys. Monitor subsequent build pipeline SSH sessions to confirm constraint enforcement is restored.
5. Step 5: Post-Incident, This vulnerability exposed a control gap in SSH agent forwarding governance. Review pipeline design to minimize or eliminate agent forwarding in favor of short-lived ephemeral SSH keys scoped to individual build jobs (least privilege principle). Update the vulnerability management process to include golang.org/x/crypto as a tracked dependency across all container build tooling.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO and legal counsel immediately if forensic evidence confirms SSH_AUTH_SOCK was accessible within a build container context during the exposure window, as this indicates live key material for production systems or regulated environments (PII/PHI repositories, PCI-scoped infrastructure) may have been exposed to container workloads or any process with access to the forwarded agent socket, potentially triggering breach notification obligations under applicable regulations.

Recovery Notes	After patching docker-buildx to the remediated azl3 version and rotating all SSH keys forwarded through affected pipelines, run a minimum 7-day monitoring period on all build pipeline SSH sessions using auditd or equivalent logging to confirm that key lifetime and confirmation constraints are now enforced by the patched golang.org/x/crypto/ssh/agent implementation. Verify no lateral movement occurred during the exposure window by auditing SSH authentication events on all systems whose keys were forwarded through the affected build hosts — specifically check for authentication events from unexpected source IPs or at unusual hours against the key fingerprints identified in Step 4. Re-validate pipeline security posture at 30 days post-recovery by re-running the constraint validation test (Step 4 compensating control) to confirm the fix persists through any subsequent package updates.
Forensic Artifacts	SSH daemon logs (/var/log/secure or /var/log/auth.log) on affected Azure Linux 3.0 build hosts: search for 'agent forwarding enabled' syslog entries correlated with the build pipeline service account username and timestamps of docker-buildx job executions during the exposure window — these entries confirm agent forwarding was active and which sessions were affected. | Docker build container environment snapshots: `docker inspect ` JSON output for containers spawned by docker-buildx 0.14.0-11 jobs, specifically the Env[] and Mounts[] arrays showing SSH_AUTH_SOCK socket path and bind mounts — direct evidence that the agent socket was injected into the build container context where key constraints were silently dropped. | golang.org/x/crypto module version in the vulnerable docker-buildx binary: output of `go version -m /usr/bin/docker-buildx` showing the exact x/crypto version linked, establishing which builds of docker-buildx were affected and confirming the vulnerable code path was present on each host. | CI/CD pipeline execution logs (Azure Pipelines audit events, GitHub Actions workflow run logs, or Jenkins build console output): specifically build steps invoking `docker buildx build --ssh default` or exporting SSH_AUTH_SOCK, which identify every build job that transmitted key material through the vulnerable agent forwarding path during the exposure window. | SSH agent socket filesystem artifacts: `ls -la /tmp/ssh-*/` and `ss -xp | grep agent` output captured during or shortly after active build jobs on affected hosts — identifies the socket path exposed to container workloads and any non-pipeline processes that may have accessed it, supporting assessment of unauthorized key use.

Per-Action IR Details

Step 1: Containment — Identify all Azure Linux 3.0 hosts running docker-buildx version 0.14.0-11 (azl3 package). Disable SSH agent forwarding (ForwardAgent no) in SSH client configs and Docker build pipeline configurations immediately until the patched package is available and deployed. Per NIST AC-17, document the temporary restriction as a remote access configuration change.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Choosing a Containment Strategy

Controls: NIST AC-17 (Remote Access), NIST CM-6 (Configuration Settings), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Run `rpm -qa docker-buildx` on each Azure Linux 3.0 host to confirm version 0.14.0-11; pipe through a hostlist via `pdsh` or a simple bash for-loop: `for h in $(cat hosts.txt); do ssh $h rpm -qa docker-buildx; done`. To disable agent forwarding fleet-wide without a config management tool, push a one-liner: `grep -rn 'ForwardAgent' /etc/ssh/ssh_config /etc/ssh/ssh_config.d/ ~/.ssh/config` to find all active ForwardAgent directives, then `sed -i 's/ForwardAgent yes/ForwardAgent no/gI' /etc/ssh/ssh_config`. For Docker build pipelines, grep CI/CD pipeline YAML files for `SSH_AUTH_SOCK` exports or `--ssh default` flags in `docker buildx build` invocations.

Evidence: Before disabling forwarding, capture the current active state: run `env | grep SSH_AUTH_SOCK` inside any running build containers to confirm live agent socket exposure. Snapshot `/etc/ssh/ssh_config`, `~/.ssh/config`, and any pipeline-specific SSH config files as-is. Collect `docker inspect ` output for active build containers to record mounted sockets and environment variables. Record `ss -xp | grep agent` or `ls -la /tmp/ssh-*/` to identify active agent socket

paths on the host. Preserve these before remediation to establish the pre-containment exposure baseline.

Step 2: Detection — Query your asset inventory (CIS 1.1) for any host running azl3 docker-buildx 0.14.0-11 on Azure Linux 3.0. Review SSH daemon logs and Docker build logs for agent-forwarded sessions: look for AuthorizedKeysFile hits combined with build job executions. Check for SSH_AUTH_SOCK environment variables present in Docker build container contexts — their presence confirms agent forwarding was active. Reference NIST AU-6 for log review scope and frequency.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Signs of an Incident

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-3 (Content of Audit Records), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 8.2 (Collect Audit Logs)

Compensating: Query asset inventory with osquery: `SELECT name, version, arch FROM deb_packages WHERE name LIKE '%docker-buildx%';` (adapt to `rpm_packages` for RPM-based Azure Linux). For log review without a SIEM, run `grep -E 'Accepted|publickey|agent' /var/log/auth.log` (or `/var/log/secure` on Azure Linux) filtered to the build pipeline service account's username across the exposure window. To detect SSH_AUTH_SOCK leakage into containers, parse Docker daemon logs at `/var/lib/docker/containers/*/json.log` and pipe through `jq` filtering for build jobs that ran during the window. Write a Sigma rule targeting `CommandLine contains '--ssh default'` or `SSH_AUTH_SOCK` in process environment for any host-level EDR or auditd deployment.

Evidence: Collect `/var/log/secure` or `/var/log/auth.log` entries showing SSH agent forwarding handshakes (look for `agent forwarding requested` and `agent forwarding enabled` syslog entries from sshd). Capture Docker build daemon logs from `/var/lib/docker/` for all build jobs executed on affected hosts during the vulnerability window — specifically jobs invoking `docker buildx build` with `--ssh` flag or SSH_AUTH_SOCK in the container environment. Pull CI/CD pipeline execution logs (GitHub Actions, Azure Pipelines, Jenkins) for any build that ran on the affected host and had agent forwarding enabled. Document all SSH key fingerprints that were forwarded during this period by parsing `ssh-add -L` output captured from build job logs if available.

Step 3: Eradication — Apply the updated azl3 docker-buildx package from the Microsoft Azure Linux 3.0 package repository per the MSRC May 2026 Patch Tuesday advisory (https://msrc.microsoft.com/update-guide/vulnerability/CVE-2026-39832). Confirm the installed version supersedes 0.14.0-11. Per CIS 7.3, validate the update was applied via automated patch management tooling and log the remediation event.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: Eliminate Components of the Incident

Controls: NIST SI-2 (Flaw Remediation), NIST CM-3 (Configuration Change Control), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: Apply the patch via `tdnf update docker-buildx` on each Azure Linux 3.0 host (tdnf is the native package manager for Azure Linux). Validate the remediated golang.org/x/crypto/ssh/agent version by running `docker buildx version` and cross-referencing the embedded Go module list: `docker buildx bake --print 2>/dev/null | head` or inspect the binary with `go version -m /usr/bin/docker-buildx | grep golang.org/x/crypto` to confirm the patched crypto version is linked. For hosts without direct internet access, mirror the updated package from the Azure Linux 3.0 repository and deploy via `tdnf install --nogpgcheck `. Log each remediation event to a central file: `echo "$(date -u) $(hostname) docker-buildx patched to $(rpm -q docker-buildx)" >> /var/log/patch_remediation.log`.

Evidence: Before applying the patch, capture the pre-patch binary hash for chain-of-custody: `sha256sum /usr/bin/docker-buildx > /evidence/docker-buildx-prepatch.sha256`. Extract the full Go module dependency list from the vulnerable binary: `go version -m /usr/bin/docker-buildx > /evidence/docker-buildx-modules-prepatch.txt` — this will show the exact golang.org/x/crypto version that contains the key constraint bypass. Preserve the pre-patch RPM metadata: `rpm -qi docker-buildx > /evidence/rpm-prepatch-info.txt`. These artifacts document the vulnerable state and confirm the scope of the flaw for post-incident reporting.

Step 4: Recovery — After patching, re-enable SSH agent forwarding only where operationally required and confirm key constraints are preserved end-to-end by testing with a constrained key (lifetime-limited or

confirmation-required). Rotate any SSH private keys that were forwarded through affected build pipelines during the exposure window, per D3-CRO (Credential Rotation). Monitor subsequent build pipeline SSH sessions via NIST AU-12 audit record generation to confirm constraint enforcement is restored.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: Restore Systems to Normal Operation

Controls: NIST AU-12 (Audit Record Generation), NIST AC-6 (Least Privilege), NIST IA-5 (Authenticator Management), CIS 5.2 (Use Unique Passwords)

Compensating: To validate that key constraints are now honored post-patch, add a test key with a 60-second lifetime via `ssh-add -t 60 ~/.ssh/test_key` and forward it through a build pipeline invocation; confirm the key expires and is removed from the agent after 60 seconds using `ssh-add -L` polling. For credential rotation, generate new ED25519 key pairs (`ssh-keygen -t ed25519 -C 'buildpipeline-rotated-$(date +%Y%m%d)'`) and update all CI/CD secret stores (GitHub Actions Secrets, Azure Key Vault, Jenkins Credentials) that held the compromised keys. Monitor post-recovery build sessions by enabling auditd rules: `auditctl -w /tmp -p rwxa -k ssh_agent_socket` to log any process accessing agent sockets under /tmp.

Evidence: Capture `ssh-add -L` output immediately before key rotation to inventory every forwarded key that was exposed — record fingerprints as evidence of the exposure scope. Pull CI/CD pipeline secret access logs for the exposure window (GitHub audit log API, Azure DevOps audit events) to identify every job that accessed the forwarded SSH keys. After rotation, document the new key fingerprints and store alongside the old ones for the incident record. Retain auditd logs showing post-patch agent socket activity for 90 days minimum to satisfy NIST AU-11 (Audit Record Retention) and support any downstream access review.

Step 5: Post-Incident — This vulnerability exposed a control gap in SSH agent forwarding governance. Review pipeline design to minimize or eliminate agent forwarding in favor of short-lived ephemeral SSH keys scoped to individual build jobs (aligned with NIST AC-6, Least Privilege, and D3-CH, Credential Hardening). Update the vulnerability management process (CIS 7.1) to include golang.org/x/crypto as a tracked dependency across all container build tooling.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Lessons Learned

Controls: NIST AC-6 (Least Privilege), NIST RA-3 (Risk Assessment), NIST SI-2 (Flaw Remediation), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 2.1 (Establish and Maintain a Software Inventory)

Compensating: Inventory golang.org/x/crypto as a tracked transitive dependency across all container build tooling by running `grep -r 'golang.org/x/crypto' /path/to/repos --include='go.sum' --include='go.mod'` across your codebase repositories and adding the package to your software inventory (CIS 2.1). Integrate OSV Scanner (`osv-scanner`) or `govulncheck` into CI/CD pipelines as a free, Google-maintained tool that will alert on future golang.org/x/crypto CVEs before deployment. Design replacement pipeline architecture using `docker buildx build --secret id=ssh_key,src=<(ssh-agent -s)` with per-job ephemeral keys generated via a short-lived certificate authority (e.g., Vault SSH Secrets Engine or a bash script wrapping `ssh-keygen -t ed25519` with `-t 300` lifetime) to eliminate persistent agent forwarding entirely.

Evidence: Compile the full incident timeline from evidence collected in steps 1-4: affected host list with exposure window per host, all forwarded key fingerprints, all build jobs that ran during the window with agent forwarding active, and the pre/post-patch binary hashes. This package forms the lessons-learned evidence base. Document any CI/CD pipeline configurations that enabled `--ssh default` or `ForwardAgent yes` as the root-cause control gap. Retain all collected artifacts per your data retention policy (NIST AU-11) and submit golang.org/x/crypto to your software dependency tracking system with a CVE watch rule to ensure future advisories against this library surface automatically.

Detection Guidance

Primary detection targets: Azure Linux 3.0 hosts with docker-buildx installed. Run 'rpm -q docker-buildx' or query your asset inventory for the azl3 package at version 0.14.0-11. In SSH daemon logs (/var/log/auth.log or journalctl -u sshd), look for sessions where agent forwarding was accepted: search for 'agent forwarding' or 'SSH_AUTH_SOCK' entries correlated with docker build job timestamps. In Docker build logs, look for build steps that invoked SSH operations, any step using RUN --mount=type=ssh or explicit ssh-agent calls during the build. This vulnerability does not produce behavioral IOCs; detection relies on asset configuration audit (docker-buildx version) and log correlation (SSH_AUTH_SOCK presence in build logs). Correlate against T1552.004 (Private Key access) by reviewing whether any SSH keys used in forwarded sessions have subsequently appeared in authentication events on systems outside the expected build pipeline scope.

Framework Mappings

MITRE-ATTACK

- **T1552.004** — Private Keys
- **T1563.001** — SSH Hijacking

CIS-V8

- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.23** — Information security for use of cloud services

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1552.004	Private Keys	Credential-Access
T1563.001	SSH Hijacking	Lateral-Movement

Sources

Source	URL	Tier
MSRC Update Guide	https://msrc.microsoft.com/update-guide/vulnerability/CVE-2026-39832	T1
(consolidated)	https://api.msrc.microsoft.com/cvrf/v3.0/cvrf/2026-May	T1
(consolidated)	https://msrc.microsoft.com/update-guide/vulnerability/CVE-2026-39833	T1
CVE-2026-39832 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-39832	T1

Source	URL	Tier
Linux Distros Unpatched Vulnerability : CVE-2026-39832 | Tenable®	https://www.tenable.com/plugins/nessus/316565	**T3**

Generated 2026-05-27 06:39 UTC by TJS Security Command Center