INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-09 06:26 UTC

KVM Hypervisor Arbitrary Code Execution via Malicious Template Registration (CVE-2026-25077)

CVE VULNERABILITY | HIGH | CVSS 8.8

SCC Item ID	SCC-CVE-2026-0149
Type	CVE Vulnerability
CVE ID	CVE-2026-25077
Severity	HIGH
CVSS Base Score	8.8
Affected Products	KVM hypervisor (specific platform/version not confirmed from available sources, see confidence note)
Published	2026-05-08
Discovery Source	Gemini

Executive Summary

A high-severity vulnerability (CVE-2026-25077, CVSS 8.8) in the KVM hypervisor allows any authenticated account user to execute arbitrary code on the underlying hypervisor host by registering a malicious template with a crafted file name. Exploitation requires only standard account-level access, meaning a compromised tenant or malicious insider can break out of the virtualized environment entirely. Organizations running KVM-based virtualization for workload hosting face risks of full host compromise, data loss, and disruption to all virtual machines sharing that host.

Technical Analysis

CVE-2026-25077 is a CWE-78 (OS Command Injection) and CWE-20 (Improper Input Validation) vulnerability in the KVM hypervisor's template registration workflow. By default, account users are permitted to register templates used for deploying virtual machine instances. The vulnerability stems from missing file name sanitization in the template registration process. An attacker supplying a crafted file name within a malicious template can inject commands executed in the context of the hypervisor host process. Successful exploitation yields arbitrary code execution on the KVM host, with potential impact to all guest VMs on that host: resource integrity loss, confidentiality breach, and denial of service. MITRE ATT&CK relevance: T1059 (Command and Scripting Interpreter), primary technique; T1190 (Exploit Public-Facing Application) and T1574 (Hijack Execution Flow) apply if the management interface is internet-exposed or if privilege escalation is required. Affected

versions and patch availability: Consult the NVD entry (https://nvd.nist.gov/vuln/detail/CVE-2026-25077) and GitHub Security Advisory (https://github.com/advisories/GHSA-vhgc-6rjx-f6vv) for platform-specific version details and vendor patch status. EPSS score and CISA KEV status are not yet populated; treat as unscored pending NVD enrichment.

Action Checklist

1. Step 1: Containment. Immediately audit which account users hold template registration permissions on KVM hosts. Restrict or revoke template registration rights for non-administrative accounts until patched. Platform-specific implementation varies: CloudStack, OpenStack, Proxmox, and other KVM management frameworks handle template permissions differently. Consult your platform vendor's security advisory for CVE-2026-25077 to identify affected versions and permission-setting procedures. Disable template registration at the platform policy level if possible.
2. Step 2: Detection. Review hypervisor host process logs and template registration audit logs for file names containing special characters, shell metacharacters (semicolons, backticks, pipe symbols, dollar signs, ampersands), or path traversal sequences (../, ..). Check KVM management API logs for unexpected template registration calls, especially from accounts that do not routinely register templates. Look for anomalous child processes spawned from the KVM host management process (libvirtd or equivalent). SIEM query guidance: alert on template registration events where file name fields contain non-alphanumeric characters outside expected naming conventions.
3. Step 3: Eradication. Apply vendor-supplied patches once confirmed available via the authoritative advisory (NVD: https://nvd.nist.gov/vuln/detail/CVE-2026-25077 and GHSA-vhgc-6rjx-f6vv). If patching is not yet available, enforce strict file name allowlists in template registration input handling as a compensating control. Remove any templates registered during the exposure window that contain anomalous file names. Rotate credentials for accounts with template registration rights as a precaution.
4. Step 4: Recovery. After patching, verify remediation by testing template registration with crafted file name inputs in a non-production environment to confirm injection is blocked. Audit all templates currently registered on affected hosts; remove or re-validate any templates registered by non-administrative accounts during the exposure window. Monitor KVM host process trees and system call activity for 72 hours post-remediation to detect any staged payloads left by prior exploitation. Restore any affected virtual machine workloads from known-good snapshots if host-level compromise is suspected.
5. Step 5: Post-Incident. This vulnerability exposes a control gap in default-permissive template registration policies: standard account users should not hold the ability to register templates without input validation controls in place. Implement least-privilege access policies for template and image management functions across all hypervisor platforms. Introduce input validation and file name sanitization requirements into your secure development and vendor procurement checklists. Map this gap to CIS Benchmark controls for hypervisor hardening and review NIST SP 800-53 SI-10 (Information Input Validation) compliance for virtualization infrastructure.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to senior IR leadership and initiate breach notification assessment if forensic evidence (auditd EXECVE events, unexpected libvirtd child processes, or anomalous template file names in logs) indicates active exploitation has occurred, as host-level KVM compromise grants an attacker access to all guest VM workloads and data on the affected hypervisor, triggering potential PII/PHI breach notification obligations under HIPAA, GDPR, or applicable state law.
Recovery Notes	After patching CVE-2026-25077, validate that the vendor fix correctly sanitizes or rejects file name inputs at the template registration API layer by re-running injection test payloads in a staging KVM environment before returning production hosts to service. All VM workloads hosted on any KVM host where exploitation cannot be ruled out should be treated as potentially compromised and restored from pre-exposure snapshots with hashes verified against known-good baselines. Maintain elevated monitoring of KVM host process trees, libvirtd child processes, and outbound network connections from hypervisor hosts for a minimum of 72 hours post-patch, extending to 7 days if any indicators of prior exploitation were identified during the investigation.
Forensic Artifacts	libvirtd process log ('/var/log/libvirt/libvirtd.log') — will contain the raw template registration API calls including the crafted file name string used to trigger code execution via CVE-2026-25077; grep for shell metacharacters and path traversal sequences in 'name' fields | Linux auditd EXECVE records ('ausearch -m EXECVE -ts ') on the KVM host — will capture any OS-level commands spawned by a shell injected through the malicious template file name, with parent PID traceable back to libvirtd | OpenStack Glance DB or CloudStack database template registration records — preserve the raw 'name' column value for each template registered during the exposure window, as the database stores the unsanitized input that triggered the vulnerability | KVM host storage pool directory listing and file metadata ('/var/lib/libvirt/images/' or configured pool path) — the registered malicious template file may be present on disk with an anomalous file name; capture 'ls -la --full-time' and file hashes before any cleanup | LiME memory capture of the KVM hypervisor host — if exploitation is suspected, a full host memory image will preserve any attacker shellcode, dropped payloads, or reverse shell artifacts injected into the hypervisor process space through the arbitrary code execution primitive

Per-Action IR Details

Step 1: Containment — Immediately audit which account users hold template registration permissions on KVM hosts. Restrict or revoke template registration rights for non-administrative accounts until patched. Confirm whether your KVM deployment platform (e.g., Apache CloudStack, OpenStack, or vendor-specific management layer) exposes this default permission and disable it at the platform policy level if possible. Consult the GitHub Security Advisory GHSA-vhgc-6rjx-f6vv for affected component identification.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-6 (Least Privilege), NIST CM-7 (Least Functionality), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 6.2 (Establish an Access Revoking Process)

Compensating: On OpenStack: run 'openstack role list --user --project ' to enumerate accounts holding 'member' or custom roles with template registration rights; revoke with 'openstack role remove'. On Apache CloudStack: use the CloudStack API call 'listAccounts' filtered by role type and cross-reference with 'listIsos'/'listTemplates' to find non-admin registrations. On bare KVM with libvirt: inspect '/etc/libvirt/libvirtd.conf' for unix_sock_rw_group membership and remove non-admin accounts from that group immediately, then restart libvirtd.

Evidence: Before revoking any permissions, capture a full export of current role assignments: 'openstack role assignment list --format json > role_snapshot_$(date +%Y%m%d%H%M%S).json' or CloudStack API

'listAccounts&listRoles' output. Preserve '/var/log/libvirt/libvirtd.log' and platform-level audit logs (OpenStack Keystone logs at '/var/log/keystone/keystone.log', CloudStack management server logs at '/var/log/cloudstack/management/management-server.log') before any changes, as these record the identity of accounts that exercised template registration rights prior to revocation.

Step 2: Detection — Review hypervisor host process logs and template registration audit logs for file names containing special characters, shell metacharacters (semicolons, backticks, pipe symbols, dollar signs, ampersands), or path traversal sequences (../, ..). Check KVM management API logs for unexpected template registration calls, especially from accounts that do not routinely register templates. Look for anomalous child processes spawned from the KVM host management process (libvirtd or equivalent). SIEM query guidance: alert on template registration events where file name fields contain non-alphanumeric characters outside expected naming conventions.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST IR-5 (Incident Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: On the KVM host, run: 'grep -E "[;&|`$]|\.\.[\/]" /var/log/libvirt/libvirtd.log' to surface template registration calls containing shell metacharacters or path traversal sequences. To detect anomalous child process spawning from libvirtd, deploy Sysmon for Linux (sysmonforlinux) with a config capturing ProcessCreate events and filter for parent process name 'libvirtd' with unexpected child processes (bash, sh, python, curl, wget): 'grep -A5 "libvirtd" /var/log/sysmon/sysmon.log | grep -E "bash|sh|python|curl|wget"'. For OpenStack, parse Nova API logs with: 'grep -i "register_template\|create_template" /var/log/nova/nova-api.log | grep -vE "^[a-zA-Z0-9_\-\.]+$"' on the name field.

Evidence: Preserve the following before any log rotation occurs: (1) '/var/log/libvirt/libvirtd.log' — contains template registration requests with file name strings; grep for GHSA-vhgc-6rjx-f6vv-relevant API calls. (2) OpenStack Nova API log ('/var/log/nova/nova-api.log') or CloudStack management server log for REST/API calls to template registration endpoints with anomalous 'name' parameters. (3) Linux auditd records ('ausearch -m EXECVE -ts today') on the KVM host to catch any shell commands spawned during exploitation. (4) '/proc//fd' and 'ss -tulpn' output captured at time of detection to identify any unexpected file descriptors or network listeners opened by a potentially compromised libvirtd process.

Step 3: Eradication — Apply vendor-supplied patches once confirmed available via the authoritative advisory (NVD: https://nvd.nist.gov/vuln/detail/CVE-2026-25077 and GHSA-vhgc-6rjx-f6vv). If patching is not yet available, enforce strict file name allowlists in template registration input handling as a compensating control. Remove any templates registered during the exposure window that contain anomalous file names. Rotate credentials for accounts with template registration rights as a precaution.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST SI-10 (Information Input Validation), NIST CM-7 (Least Functionality), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: Until the vendor patch is available, implement a file name allowlist at the libvirt API layer using an AppArmor or SELinux policy that restricts the characters permitted in template name strings passed to the KVM management daemon. On OpenStack, deploy a Nova API policy.json modification or a custom middleware filter that rejects template registration requests where the 'name' field matches the regex '[^a-zA-Z0-9_\-\.]'. For CloudStack, use the API firewall (HAProxy ACL or nginx location block) to reject template registration POST bodies containing shell metacharacters before they reach the management server. Enumerate and quarantine suspicious templates with: 'openstack image list --format json | python3 -c "import sys,json,re; [print(i) for i in json.load(sys.stdin) if re.search(r\"[;&|\`\$\.\.]\", i[\"Name\"])]"'.

Evidence: Before removing suspicious templates, forensically image each flagged template file from the KVM host storage backend (typically '/var/lib/libvirt/images/' or the configured storage pool path) using 'dd if=/var/lib/libvirt/images/

of=/evidence/.img bs=4M' and hash with 'sha256sum'. Capture the template metadata record from the platform database (Nova Glance DB or CloudStack DB) before deletion to preserve the registered file name string, submitting account ID, and registration timestamp as evidence of the injection attempt.

Step 4: Recovery — After patching, verify remediation by testing template registration with crafted file name inputs in a non-production environment to confirm injection is blocked. Audit all templates currently registered on affected hosts; remove or re-validate any templates registered by non-administrative accounts during the exposure window. Monitor KVM host process trees and system call activity for 72 hours post-remediation to detect any staged payloads left by prior exploitation. Restore any affected virtual machine workloads from known-good snapshots if host-level compromise is suspected.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CP-10 (System Recovery and Reconstitution), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Verify patch efficacy in a non-production KVM host by attempting template registration via the API with payloads such as 'name=test;id', 'name=../../../etc/passwd', and 'name=test`whoami`' and confirming HTTP 4xx rejection or sanitized storage with no command execution. For 72-hour post-patch monitoring on the production KVM host, use auditd with a rule watching libvirtd's working directory and child process spawning: 'auditctl -a always,exit -F arch=b64 -S execve -F ppid=$(pgrep libvirtd) -k kvm_child_exec'; review with 'ausearch -k kvm_child_exec'. Use 'osquery' with the 'processes' and 'process_events' tables to continuously surface any new processes parented to libvirtd: 'SELECT pid, name, cmdline, parent FROM processes WHERE parent=(SELECT pid FROM processes WHERE name="libvirtd");'

Evidence: Prior to restoring any VM workloads, capture full memory of the KVM host using 'LiME' (Linux Memory Extractor) kernel module to preserve any in-memory staged payloads or attacker tooling that may have been injected into the hypervisor host process space via CVE-2026-25077 exploitation. Hash all known-good VM snapshots with sha256sum before restoration and verify hashes post-restore to confirm snapshot integrity. Retain '/var/log/libvirt/' directory contents, auditd logs, and syslog from the entire exposure window (from earliest plausible exploitation date through patch application) for post-incident review.

Step 5: Post-Incident — This vulnerability exposes a control gap in default-permissive template registration policies: standard account users should not hold the ability to register templates without input validation controls in place. Implement least-privilege access policies for template and image management functions across all hypervisor platforms. Introduce input validation and file name sanitization requirements into your secure development and vendor procurement checklists. Map this gap to CIS Benchmark controls for hypervisor hardening and review NIST SP 800-53 SI-10 (Information Input Validation) compliance for virtualization infrastructure.

NIST Phase: Post Incident

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

Controls: NIST IR-8 (Incident Response Plan), NIST SI-10 (Information Input Validation), NIST AC-6 (Least Privilege), NIST SI-5 (Security Alerts, Advisories, and Directives), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 4.7 (Manage Default Accounts on Enterprise Assets and Software)

Compensating: Document a hypervisor-specific hardening runbook that adds the following standing checks to your periodic configuration review: (1) monthly audit of libvirt unix socket group membership and OpenStack/CloudStack role assignments for template management rights; (2) a Sigma rule (deployable to any log shipper without a SIEM) detecting template registration API calls with non-alphanumeric characters in name fields — publish to your team's shared detection-as-code repository; (3) add 'file name input validation for hypervisor template/image registration' as a mandatory line item in vendor RFP security questionnaires and third-party software procurement checklists going forward.

Evidence: For the lessons-learned record, compile: the timeline of the exposure window (first vulnerable version deployment date through patch application), a list of all accounts that held template registration rights during that window, all template registration events from platform logs during the exposure period (exported as structured JSON for retention), and the forensic images of any suspicious templates collected during eradication — retain per your organization's incident record retention policy (minimum 1 year recommended, aligned with NIST AU-11 (Audit Record Retention)).

Detection Guidance

Focus detection on the template registration pathway of the KVM management layer. Query template registration audit logs for file name fields containing shell metacharacters: semicolons (;), pipe characters (|), backticks (`), dollar signs ($), ampersands (&), and path traversal strings (../ or ..\ ). Alert on any template registration event from a standard (non-administrative) account, particularly outside business hours or from unusual source IPs. On the KVM host, monitor libvirtd (or equivalent host management daemon) for unexpected child process spawns, especially shells (bash, sh, dash) or network utilities (curl, wget, nc) initiated from the management process. If your SIEM ingests Linux auditd logs, create rules targeting execve syscalls with parent processes matching the KVM management daemon. There are no published IOCs for this CVE at this time; behavioral and log-pattern detection is the primary approach until threat actor exploitation activity is documented.

Framework Mappings

MITRE-ATTACK

- **T1574** — Hijack Execution Flow
- **T1059** — Command and Scripting Interpreter
- **T1190** — Exploit Public-Facing Application

NIST-800-53R5

- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-10** — Information Input Validation

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **2.5** — Allowlist Authorized Software
- **16.10** — Apply Secure Design Principles in Application Architectures

ISO-27001-2022

- **A.8.26** — Application security requirements
- **A.8.8** — Management of technical vulnerabilities

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1574	Hijack Execution Flow	Persistence
T1059	Command and Scripting Interpreter	Execution
T1190	Exploit Public-Facing Application	Initial-Access

Sources

Source	URL	Tier
CVE-2026-25077 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-25077?utm_source=feedly	T1
CVE-2026-25077 - Exploits & Severity - Feedly	https://feedly.com/cve/CVE-2026-25077	T3
Account users are allowed by default to register... · CVE-2026-25077	https://github.com/advisories/GHSA-vhgc-6rjx-f6vv	T3
How Cloudflare responded to the “Copy Fail” Linux vulnerability	https://blog.cloudflare.com/copy-fail-linux-vulnerability-mitigation/	T3
Exploitation of 'Copy Fail' Linux Vulnerability Begins - SecurityWeek	https://www.securityweek.com/exploitation-of-copy-fail-linux-vulner...	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-25077	T1

Generated 2026-05-09 06:26 UTC by TJS Security Command Center