INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-03-29 18:34 UTC

Critical n8n Flaws Allow Remote Code Execution and Exposure of Stored Credentials

CVE VULNERABILITY | CRITICAL | CVSS 9.8

SCC Item ID	SCC-CVE-2026-0014
Type	CVE Vulnerability
CVE ID	CVE-2026-0014
Severity	CRITICAL
CVSS Base Score	9.8
Affected Products	n8n workflow automation platform (specific versions unconfirmed from available data; patches available per sources)
Published	2026-03-11

Executive Summary

Two critical vulnerabilities in the n8n workflow automation platform allow unauthenticated remote code execution and exposure of stored credentials, with a public proof-of-concept exploit already available. Organizations running n8n, particularly self-hosted instances used in automated workflows, face server takeover risk and potential credential theft from connected systems. Patches are available; any unpatched deployment should be treated as actively at risk given PoC availability.

Technical Analysis

Two now-patched critical flaws affect the n8n workflow automation platform. Specific CVE identifiers were not confirmed from available data; source articles reference the vulnerabilities but CVE IDs could not be extracted without direct article access. Confidence in version-specific scope is LOW; consult the n8n release notes and official security advisories for precise version ranges. The MITRE and CWE mappings below are inferred from attack pattern description; they have not been confirmed against official CVE records (pending CVE ID confirmation). CWE mapping from item data: CWE-94 (Code Injection) drives the RCE vector; CWE-522 (Insufficiently Protected Credentials) drives the credential exposure; CWE-20 (Improper Input Validation) is the likely root cause enabling both. MITRE ATT&CK alignment: T1059 (Command and Scripting Interpreter), T1190 (Exploit Public-Facing Application), T1552 (Unsecured Credentials). Reported CVSS base score: 9.8. A public proof-of-concept is confirmed available for at least one RCE vulnerability per Field Effect reporting, elevating practical exploitation risk substantially. Attack scenario: an attacker exploits the input validation flaw to inject and execute code server-side, then harvests credentials stored within n8n workflow configurations. Self-hosted and publicly exposed n8n instances carry the highest exposure. Note: CISA KEV listing status: not identified

(cisa_kev field: false). EPSS score not populated; treat PoC availability as the primary exploitation signal.

Action Checklist

1. Step 1, Patch immediately: Update all n8n instances to the latest patched release. Consult n8n's official release notes and security advisory page for version guidance.
2. Step 2, Isolate exposed instances: If patching cannot happen immediately, take publicly exposed n8n instances offline or restrict access to trusted IP ranges via firewall rules or reverse proxy ACLs.
3. Step 3, Inventory deployments: Identify all n8n instances across your environment, cloud, on-premises, containerized, developer workstations. Include shadow IT; n8n is frequently self-installed.
4. Step 4, Rotate stored credentials: Audit credentials stored in n8n workflow configurations (API keys, database passwords, service account tokens). Rotate any that were accessible on potentially compromised instances. Revoke and reissue rather than just rotating where possible.
5. Step 5, Check for compromise indicators: Review n8n server logs and host-level process execution logs for anomalous command execution, unexpected outbound connections, or unauthorized workflow modifications. See detection guidance below.
6. Step 6, Notify affected stakeholders: Alert application owners and data owners for any systems whose credentials are stored in n8n workflows. If compromise cannot be ruled out, treat as a credential exposure incident and follow your IR playbook.
7. Step 7, Review architecture: Evaluate whether n8n instances require public internet exposure. Enforce least-privilege on service accounts used in workflows. Add n8n to your asset inventory and vulnerability management scope if not already present.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO and external IR firm immediately if any n8n instance was publicly exposed for >24 hours before patch, if credential compromise is confirmed through access logs, or if production workflows connected to critical systems (payment, healthcare, identity) cannot be patched within 4 hours.
Recovery Notes	Post-remediation: restore n8n instances from clean backup (created after patch application); re-validate all integrated systems show no unauthorized access or data exfiltration using upstream system audit logs (CloudTrail, database logs, API provider logs); document lessons learned around n8n deployment governance (no shadow IT, mandatory asset tagging, network segmentation); implement quarterly n8n vulnerability assessments as part of your vulnerability management program.
Forensic Artifacts	n8n application logs (/root/.n8n/logs or container log output via docker logs) | n8n database exports and workflow JSON configurations | Linux auditd logs or Windows Event Log (Event ID 4688 for process execution, 4688 with CommandLine parameter) | Reverse proxy access logs (nginx/Apache) and firewall logs showing inbound connections to n8n ports | Upstream system audit logs: AWS CloudTrail, database transaction logs, API provider request logs showing credential usage timestamps

Per-Action IR Details

Step 1, Patch immediately: Update all n8n instances to the latest patched release. Consult n8n's official release notes and security advisory page for version guidance.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2.1 (preparation phase: tools, resources, training)

Controls: NIST 800-53 SI-2 (Flaw Remediation), CIS 2.3 (Address Unauthorized Software)

Compensating: Before patching: export n8n database backup using `docker exec npm run db:export` or native backup tool; document current version with `curl http://localhost:5678/api/v1/health`. For air-gapped deployments, download patch offline and validate checksum against n8n's GitHub release page before deployment.

Evidence: Capture n8n version string from /api/v1/health endpoint, database export timestamp, container image digest (docker inspect), and full n8n application logs from startup to patch application. Preserve pre-patch state for forensic comparison.

Step 2, Isolate exposed instances: If patching cannot happen immediately, take publicly exposed n8n instances offline or restrict access to trusted IP ranges via firewall rules or reverse proxy ACLs.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.2.3 (containment: short-term and long-term strategies)

Controls: NIST 800-53 AC-3 (Access Enforcement), CIS 1.3 (Address Unauthorized Network Access)

Compensating: Use iptables (Linux) or Windows Firewall to restrict port 5678 to localhost only: `iptables -A INPUT -p tcp --dport 5678 -s 127.0.0.1 -j ACCEPT; iptables -A INPUT -p tcp --dport 5678 -j DROP`. For Kubernetes, update NetworkPolicy to deny all ingress and permit only internal service discovery. Document all IP blocks with timestamp.

Evidence: Capture firewall rule state before and after isolation (iptables-save output, Windows Firewall export via netsh advfirewall show allprofiles). Document reverse proxy access logs for the 24 hours prior to isolation to identify potential reconnaissance or exploit attempts. Preserve n8n request logs (nginx/Apache access logs if present).

Step 3, Inventory deployments: Identify all n8n instances across your environment, cloud, on-premises, containerized, developer workstations. Include shadow IT; n8n is frequently self-installed.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2.1 (preparation: asset inventory and baseline)

Controls: NIST 800-53 CM-8 (Information System Component Inventory), CIS 1.1 (Establish and Maintain Software Inventory)

Compensating: Scan network with `nmap -p 5678 ` to locate default n8n port; cross-reference with process discovery (`ps aux | grep n8n` on Linux, `Get-Process | Select-String n8n` on Windows). Query DNS logs for n8n domain patterns and check cloud provider APIs (AWS EC2 describe-instances, Azure VM list) for n8n tags. Document each instance with IP, hostname, version, and owner.

Evidence: Capture network scan results, DNS query logs for the past 30 days, process snapshots from all systems, and cloud resource inventory snapshots. Preserve running process memory dump if available (volatility or native OS tools) to confirm n8n version and active module configuration.

Step 4, Rotate stored credentials: Audit credentials stored in n8n workflow configurations (API keys, database passwords, service account tokens). Rotate any that were accessible on potentially compromised instances. Revoke and reissue rather than just rotating where possible.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.2.5 (eradication: removing threat components and restoring systems)

Controls: NIST 800-53 IA-4 (Identifier Management), NIST 800-53 IA-5 (Authentication), CIS 5.4 (Use Multi-Factor Authentication)

Compensating: Export n8n workflows as JSON using the API or UI export function and grep for credential patterns (API_KEY, password, secret, token). Manually audit each stored credential against your secrets vault (HashiCorp Vault, AWS Secrets Manager, Azure Key Vault). For each exposed credential, immediately revoke in the upstream system (e.g., AWS IAM user access key deletion, database password reset, API provider token revocation). Log

timestamp and approver for each revocation.

Evidence: Capture exported workflow JSON files (sanitize before storing in security system). Document credential inventory with exposure timeline. Preserve application logs from upstream systems (AWS CloudTrail, database audit logs) showing the time window when exposed credentials were used — this establishes whether attackers accessed them. Screenshot secrets vault audit logs showing revocation events.

Step 5, Check for compromise indicators: Review n8n server logs and host-level process execution logs for anomalous command execution, unexpected outbound connections, or unauthorized workflow modifications. See detection guidance below.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.1.1 (analysis: indicators of compromise and event triage)

Controls: NIST 800-53 SI-4 (Information System Monitoring), CIS 3.9 (Implement and Maintain File Integrity Monitoring)

Compensating: Parse n8n application logs (`/root/.n8n/logs` or configurable path) with grep for ERROR, EXEC, RUN, or code execution patterns. Use `auditctl` (Linux) to monitor system calls: `auditctl -w /root/.n8n -p wa -k n8n_changes` for 7 days prior. Check command history: `history` on Linux, `Get-History` on PowerShell. Correlate against netstat or ss for unexpected outbound connections on ports 80/443/custom.

Evidence: Collect full n8n application logs (entire directory if multi-file). Capture audit logs from `ausearch` or Windows Event Log (Event ID 4688 for process creation, 3389 for outbound connections). Extract bash history, PowerShell history, and process execution timeline. Preserve network traffic captures (tcpdump/Wireshark .pcap) for 24 hours prior to detection if available. Screenshot workflow UI showing last modified timestamps.

Step 6, Notify affected stakeholders: Alert application owners and data owners for any systems whose credentials are stored in n8n workflows. If compromise cannot be ruled out, treat as a credential exposure incident and follow your IR playbook.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §3.4 (post-incident activities: lessons learned and notification)

Controls: NIST 800-53 IR-4 (Incident Handling), CIS 6.2 (Develop Incident Response Plan)

Compensating: Generate credential exposure report listing each system, credential type, exposure window, and remediation status. Document notification via email or ticketing system with timestamp and recipient acknowledgment. If compromise is suspected, follow your organization's data breach notification policy (state-specific requirements, regulatory obligations). Escalate to Legal and Privacy if PII/PHI is involved.

Evidence: Preserve notification log (email send receipts, ticket records). Document acknowledgment responses from stakeholders. Maintain chain of custody on the credential inventory and exposure analysis (date, analyst, signature). Screenshot or export incident ticket showing severity classification and escalation path.

Step 7, Review architecture: Evaluate whether n8n instances require public internet exposure. Enforce least-privilege on service accounts used in workflows. Add n8n to your asset inventory and vulnerability management scope if not already present.

NIST Phase: Recovery

Reference: NIST 800-61r3 §2.1 (preparation: architecture and configuration baseline) and §3.3 (recovery: system restoration)

Controls: NIST 800-53 AC-2 (Account Management), NIST 800-53 AC-6 (Least Privilege), CIS 1.2 (Actively Manage All Hardware and Software)

Compensating: Audit current network ACLs and routing rules for n8n ingress using `iptables -L -v -n` or cloud console exports. For each workflow, document required upstream integrations and create service accounts with minimal IAM permissions (e.g., AWS IAM role with only S3 read, no admin). Deploy n8n behind a reverse proxy (nginx/Apache) with authentication and rate limiting. Add n8n to your vulnerability scanner configuration (NESSUS template, Qualys) and schedule monthly scans. Document architectural changes in your CMDB or infrastructure-as-code repository.

Evidence: Export network topology diagram showing n8n placement and data flows. Capture IAM policy documents for all n8n service accounts (before and after least-privilege hardening). Preserve vulnerability scanner baseline reports. Screenshot or export asset inventory entries showing n8n version, deployment type, and owner. Document architectural review meeting notes with sign-off.

Detection Guidance

Focus detection on three signals: (1) Unexpected process spawning from the n8n process, on Linux, look for n8n spawning shells (bash, sh, python) or network tools (curl, wget, nc). Query host EDR or auditd logs for parent process = n8n and child process in [bash, sh, python3, curl, wget, netcat]. (2) Anomalous outbound connections from n8n hosts, baseline normal n8n callback destinations and alert on new external IPs or domains, especially to infrastructure not matching configured integrations. (3) Unauthorized workflow creation or modification, n8n logs workflow activity; review for workflow changes made outside normal business hours or by accounts not associated with your team. If n8n is running in Docker, examine container logs with: docker logs [container_name] and look for stack traces or error patterns associated with code injection attempts. SIEM query pattern (generic): source_process='n8n' AND (child_process IN ('bash','sh','cmd','python','perl','ruby') OR outbound_dest NOT IN [known_integration_ips]). Note: absence of known IOCs in available source data does not indicate absence of active exploitation. Assume threat activity is ongoing given PoC availability.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`https://github.com/n8n-io/n8n/releases`	Official n8n release page — verify patched version and release notes. Confirm this URL resolves before use; provided as search-retrieved reference, not from verified IOC feed.	LOW

Framework Mappings

MITRE-ATTACK

- **T1059** — Command and Scripting Interpreter
- **T1190** — Exploit Public-Facing Application
- **T1552** — Unsecured Credentials

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
- **IA-5** — Authenticator Management

OWASP-TOP10-2021

- **A03:2021** — Injection
- **A04:2021** — Insecure Design
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **16.10**
- **5.2**
- **6.3** — Require MFA for Externally-Exposed Applications
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.8.26** — Application security requirements
- **A.8.8** — Management of technical vulnerabilities
- **A.5.23** — Information security for use of cloud services

HIPAA-SECURITY

- **164.308(a)(5)(ii)(D)** — Password Management
- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1059	Command and Scripting Interpreter	Execution
T1190	Exploit Public-Facing Application	Initial-Access
T1552	Unsecured Credentials	Credential-Access

Sources

Source	URL	Tier
The Hacker News	https://thehackernews.com/2026/03/critical-n8n-flaws-allow-remote-c...	T3

Source	URL	Tier
Critical N8n Vulnerabilities Allowed Server Takeover - SecurityWeek	https://www.securityweek.com/critical-n8n-vulnerabilities-allowed-s...	T3
Simon J Smith's Post - LinkedIn	https://www.linkedin.com/posts/simonsmithinvestigator_critical-n8n-...	T3
POC available for critical n8n remote command execution flaw	https://fieldeffect.com/blog/poc-available-critical-n8n-remote-comm...	T3
Critical n8n Flaws Enable Remote Code Execution and Credential ...	https://greatis.com/unhackme/help/news/critical-n8n-flaws-enable-re...	T3

Generated 2026-03-29 18:34 UTC by TJS Security Command Center