INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-05 06:56 UTC

budibase budibase - budibase budibase Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection')

CVE VULNERABILITY | **CRITICAL** | CVSS 9.8 | **CISA KEV**

SCC Item ID	SCC-CVE-2026-0265
Type	CVE Vulnerability
CVE ID	CVE-2026-31816
Severity	CRITICAL
CVSS Base Score	9.8
EPSS Score	0.1695 (95th percentile)
KEV Status	Yes — CISA Known Exploited Vulnerability
Affected Products	Budibase <= 3.31.4
Published	2026-06-05T00:00:00Z
Discovery Source	Vulncheck Kev

Executive Summary

A critical authentication bypass vulnerability (CVE-2026-31816, CVSS 9.8) in Budibase, a low-code platform used to build internal business tools, allows unauthenticated attackers to access any API endpoint without credentials. A public reverse shell exploit (CVE-2026-31816-rshell) is available on GitHub and CISA has confirmed active exploitation in the wild. Any organization running Budibase 3.31.4 or earlier with internet-facing deployments faces immediate risk of full system compromise.

Technical Analysis

CVE-2026-31816 affects Budibase <= 3.31.4 (CWE-288: Authentication Bypass, CWE-863: Incorrect Authorization, CWE-74: Injection). The vulnerability resides in the server-side authorized() middleware, which enforces authentication, role-based access control, and CSRF protection across all API endpoints. The isWebhookEndpoint() function uses an unanchored regular expression matched against ctx.request.url, which in the Koa framework includes the full URL string with query parameters. An attacker appending a webhook path pattern (e.g., ?/webhooks/trigger) to any API request URL causes the regex to match on the injected query string segment, triggering an immediate return next() and bypassing all security controls. No authentication is

required. MITRE ATT&CK mapping: T1190 (Exploit Public-Facing Application), T1078 (Valid Accounts, bypassed entirely), T1059 (Command and Script Interpreter, via published reverse shell). A public reverse shell exploit is published on GitHub (repository: imjdl/CVE-2026-31816-rshell). The vulnerability is listed on the CISA Known Exploited Vulnerabilities catalog and tracked by VulnCheck KEV, confirming active exploitation. EPSS score: 0.16947 (95th percentile). Patch status: upgrade to a version beyond 3.31.4; verify with the Budibase project release notes directly at https://github.com/budibase/budibase/releases.

Action Checklist

1. Step 1: Containment, Immediately isolate all internet-facing Budibase instances running version 3.31.4 or earlier. Block external access at the network perimeter (firewall or WAF) by restricting inbound requests to Budibase API ports. If the instance cannot be taken offline, configure WAF rules to reject requests containing webhook path patterns (e.g., '/webhooks/trigger') appearing in query string parameters.
2. Step 2: Detection, Review web server and application logs for requests to any API endpoint containing '/webhooks/' in the query string (e.g., GET /api/anything?/webhooks/trigger). Look for HTTP 200 responses to API endpoints from unauthenticated sources. Audit logs for unexpected command execution, new user account creation, or privilege escalation consistent with MITRE T1059 and T1078. Check for outbound connections from the Budibase host to unknown IPs, which may indicate reverse shell activity from the published PoC on GitHub. Apply NIST AU-6 (Audit Record Review, Analysis, and Reporting) and CIS 8.2 (Collect Audit Logs).
3. Step 3: Eradication, Upgrade Budibase to a version beyond 3.31.4 as soon as a patched release is available from the official Budibase project. Verify patch availability at the official Budibase repository (https://github.com/budibase/budibase/releases; confirm via official domain budibase.com). Until a patch is confirmed, enforce network-level controls that prevent direct external access to the Budibase API. Remove or rotate all API keys, service account credentials, and session tokens that may have been accessible via the bypass window, per NIST IA-4 (Identifier Management) and IA-5 (Authenticator Management).
4. Step 4: Recovery, After patching, verify the isWebhookEndpoint() regex is anchored and no longer matches query string injection. Conduct authenticated and unauthenticated API requests against representative endpoints and confirm 401/403 responses are returned without the webhook query string trick. Re-enable external access only after validation. Monitor application and network logs for 72 hours post-fix for any signs of persistence or lateral movement consistent with T1059. Apply NIST AC-2 (Account Management) to detect residual unauthorized account activity.
5. Step 5: Post-Incident, Conduct a control gap review against NIST AC-3 (Access Enforcement) and NIST AC-6 (Least Privilege) to assess whether API authentication controls are consistently applied across all middleware. Review regex-based security logic across the codebase for unanchored patterns that could be exploited similarly. Establish a process to track CISA KEV additions and require response within CISA's defined remediation windows. Reference CIS 7.1 (Establish and Maintain a Vulnerability Management Process) and CIS 7.2 (Establish and Maintain a Remediation Process) to formalize this as a repeatable workflow.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO, legal counsel, and breach notification review if log analysis confirms HTTP 200 responses to unauthenticated API requests during the exposure window, any attacker-created accounts are discovered in the Budibase user database, or if Budibase apps have data source connections to systems containing PII, PHI, or financial data — all of which indicate potential data breach triggering regulatory notification obligations.
Recovery Notes	After applying the Budibase patch beyond 3.31.4, validate that both the `isWebhookEndpoint()` fix and any WAF rules are operating correctly by running unauthenticated bypass attempts against all API route families (`/api/global/`, `/api/apps/`, `/api/automations/`) and confirming uniform 401/403 responses. Maintain enhanced logging at DEBUG level on the Budibase application for a minimum 72-hour post-recovery window, retaining logs externally (not on the Budibase host) to detect any persistence mechanisms — particularly scheduled automations or webhook triggers the attacker may have created through the unauthenticated API before containment. Re-inventory all downstream systems and credentials that Budibase automation workflows connected to during the exposure window and treat them as potentially compromised until verified otherwise.
Forensic Artifacts	Budibase application access logs with full query strings (default path: /home/budibase/.budibase/logs/ or Docker volume mount) — the auth bypass leaves a distinctive fingerprint of HTTP 200 responses to /api/* endpoints where the query string contains '/webhooks/trigger' or similar webhook path fragments; logs must include the %q (query string) component or the bypass requests will be invisible. | Budibase backend database user table (CouchDB /_users endpoint or PostgreSQL bb-users table depending on deployment type) — attacker with unauthenticated API access could create persistent admin accounts; compare pre- and post-incident snapshots to identify rows inserted during the exploitation window. | OS-level process execution logs from auditd (Linux /var/log/audit/audit.log, event type EXECVE) or Sysmon Event ID 1 (Process Creation) filtered to the Budibase/Node.js process user — the public CVE-2026-31816-rshell PoC executes OS commands via the unauthenticated API, which will appear as child processes (sh, bash, python) spawned under the Budibase service account. | Network flow or connection state logs showing outbound TCP connections from the Budibase host process — the published reverse shell PoC establishes a persistent outbound callback to an attacker-controlled IP; capture via `ss -tnp` snapshots, netflow records, or Sysmon Event ID 3 (Network Connection) filtered to the Node.js PID, noting destination IP, port, and connection duration. | Budibase app and automation configuration exports (accessible via /api/automations and /api/apps endpoints when authenticated) — an attacker with unauthenticated API access may have modified existing automation workflows to embed persistent command execution or data exfiltration logic; compare current automation configs against version-controlled backups or prior exports to detect unauthorized modifications.

Per-Action IR Details

Step 1: Containment — Immediately isolate all internet-facing Budibase instances running version 3.31.4 or earlier. Block external access at the network perimeter (firewall or WAF) by restricting inbound requests to Budibase API ports. If the instance cannot be taken offline, configure WAF rules to reject requests containing webhook path patterns (e.g., '/webhooks/trigger') appearing in query string parameters.

NIST Phase: Containment

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

Controls: NIST AC-4 (Information Flow Enforcement), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: On Linux hosts, immediately apply iptables rules to drop inbound traffic on the Budibase API port (default 4001/tcp) from non-trusted CIDRs: `iptables -I INPUT -p tcp --dport 4001 ! -s -j DROP`. For WAF-less

environments, deploy NGINX as a reverse proxy and add a `location` block rejecting any request URI or query string matching `(?i)/webhooks/` with a 403 return code. Verify with `curl -v 'http://:4001/api/test?/webhooks/trigger'` confirming 403 response before re-testing application functionality.

Evidence: Before isolating, capture a full netstat/ss snapshot (`ss -tnp`) to document any established outbound connections from the Budibase process — active reverse shell sessions from the CVE-2026-31816-rshell PoC will appear as outbound TCP connections from the Node.js/Budibase PID to attacker-controlled IPs on non-standard ports. Also capture running process list (`ps auxf`) and active Docker network connections if Budibase is containerized (`docker inspect ` and `docker exec ss -tnp`). Preserve these before isolation severs attacker visibility.

Step 2: Detection — Review web server and application logs for requests to any API endpoint containing '/webhooks/' in the query string (e.g., GET /api/anything?/webhooks/trigger). Look for HTTP 200 responses to API endpoints from unauthenticated sources. Audit logs for unexpected command execution, new user account creation, or privilege escalation consistent with MITRE T1059 and T1078. Check for outbound connections from the Budibase host to unknown IPs, which may indicate reverse shell activity from the published PoC (CVE-2026-31816-rshell on GitHub). Apply NIST AU-6 (Audit Record Review, Analysis, and Reporting) and CIS 8.2 (Collect Audit Logs).

NIST Phase: Detection Analysis

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

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-2 (Event Logging), NIST AU-3 (Content Of Audit Records), CIS 8.2 (Collect Audit Logs)

Compensating: Parse Budibase application logs (typically at `/home/budibase/.budibase/logs/` or Docker volume mount) using grep: `grep -E 'GET|POST.*/api/.*\?.*\/webhooks\/' access.log | grep ' 200 '` to identify successful authentication bypass requests. For reverse shell detection without EDR, deploy Sysmon on the Budibase host with EventID 3 (Network Connection) monitoring outbound connections from the Budibase/Node.js process; query with: `Get-WinEvent -LogName 'Microsoft-Windows-Sysmon/Operational' | Where-Object {$_.Id -eq 3 -and $_.Message -match 'node'}`. On Linux, use auditd with a rule targeting execve syscalls from the Budibase process user: `auditctl -a always,exit -F arch=b64 -S execve -F uid= -k budibase_exec` and review with `ausearch -k budibase_exec`.

Evidence: Capture the following before log rotation: (1) Full Budibase application access logs covering the past 90 days showing raw request URIs including query strings — the bypass uses the query string path `/webhooks/trigger` appended to legitimate API routes, so standard access logs that truncate query strings will miss it; confirm your log format includes `%q` or equivalent. (2) Budibase database (CouchDB or PostgreSQL depending on deployment) user table exports — attacker may have created admin accounts via the unauthenticated API. (3) OS-level auth logs (`/var/log/auth.log` or `/var/log/secure`) for SSH logins or sudo usage by unexpected users following the exploitation window.

Step 3: Eradication — Upgrade Budibase to a version beyond 3.31.4 as soon as a patched release is available from the official Budibase project (verify at the official Budibase GitHub repository or release channel). Until a patch is confirmed, enforce network-level controls that prevent direct external access to the Budibase API. Remove or rotate all API keys, service account credentials, and session tokens that may have been accessible via the bypass window, per D3-CRO (Credential Rotation).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST AC-2 (Account Management), NIST AC-3 (Access Enforcement), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 5.2 (Use Unique Passwords)

Compensating: If a patched Budibase release is not yet available, implement a mandatory authentication proxy in front of all `/api/` routes using NGINX `auth_request` directive pointing to an internal auth service, forcing token validation before any request reaches the Budibase application layer. For credential rotation without a PAM system: (1) Export all API keys from the Budibase admin panel (`/builder/portal/settings/apikeys`), invalidate all existing keys, and reissue only to verified service accounts; (2) Reset the Budibase admin password and any service account passwords

stored in the Budibase user database via the CouchDB admin interface or `docker exec ./cli.js reset-admin`; (3) Rotate any downstream credentials (database passwords, third-party API keys) that Budibase automation workflows may have had access to through app integrations.

Evidence: Before applying the patch and rotating credentials, preserve: (1) A full export of the Budibase user/account table from the backend database (CouchDB `/_users` endpoint or equivalent PostgreSQL `bb-users` table) — this documents any attacker-created accounts that must be removed; (2) All active API keys from the Budibase admin interface with associated last-used timestamps — keys accessed during the exploitation window are presumed compromised; (3) File system snapshot of the Budibase application directory (`/home/budibase/.budibase/`) to preserve any webshells or modified files the attacker may have written via the reverse shell PoC before eradication removes evidence.

Step 4: Recovery — After patching, verify the isWebhookEndpoint() regex is anchored and no longer matches query string injection. Conduct authenticated and unauthenticated API requests against representative endpoints and confirm 401/403 responses are returned without the webhook query string trick. Re-enable external access only after validation. Monitor application and network logs for 72 hours post-fix for any signs of persistence or lateral movement consistent with T1059. Apply D3-LAM (Local Account Monitoring) to detect residual unauthorized account activity.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AC-2 (Account Management), NIST AC-7 (Unsuccessful Logon Attempts), CIS 5.3 (Disable Dormant Accounts), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: Validate patch effectiveness using a 2-step manual test: (1) Unauthenticated bypass attempt — `curl -v -X GET 'https:///api/global/users?/webhooks/trigger'` must return 401, not 200; repeat against `/api/apps`, `/api/automations`, and any custom API routes used by your Budibase apps. (2) Authenticated request — `curl -H 'x-budibase-api-key: ' https:///api/global/users` must return 200, confirming authenticated access still functions post-patch. For 72-hour persistence monitoring without SIEM, configure a cron job that runs `grep -c '200' /var/log/budibase/access.log` every 15 minutes and alerts via email if unauthenticated 200 responses reappear, and run `ss -tnp | grep node` hourly to flag any new outbound connections from the Budibase process.

Evidence: During recovery validation, preserve test request/response pairs with full headers as evidence of restored auth enforcement — these serve as the 'known-good' baseline for future comparison. Capture and retain the patched `isWebhookEndpoint()` function source from the updated Budibase package for documentation. Export a post-recovery snapshot of the Budibase user database and compare against the pre-eradication snapshot to confirm all attacker-created accounts were removed and no new unauthorized accounts appeared during the recovery window.

Step 5: Post-Incident — Conduct a control gap review against NIST AC-3 (Access Enforcement) and NIST AC-6 (Least Privilege) to assess whether API authentication controls are consistently applied across all middleware. Review regex-based security logic across the codebase for unanchored patterns that could be exploited similarly. Establish a process to track CISA KEV additions and require response within CISA's defined remediation windows. Reference CIS 7.1 (Establish and Maintain a Vulnerability Management Process) and CIS 7.2 (Establish and Maintain a Remediation Process) to formalize this as a repeatable workflow.

NIST Phase: Post Incident

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

Controls: NIST AC-3 (Access Enforcement), NIST AC-6 (Least Privilege), NIST AU-11 (Audit Record Retention), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: For regex security review without a commercial SAST tool: run `grep -rn 'new RegExp\|\.test(\|\.match(' ` across the Budibase source to enumerate all regex-based access control checks, then manually verify each uses anchored patterns (`^` and `$` or `\A`/`\z`) rather than substring matching. Subscribe to the

CISA KEV RSS feed (`https://www.cisa.gov/sites/default/files/feeds/known_exploited_vulnerabilities.json`) and create a weekly cron job that parses additions for products in your asset inventory and generates a ticket if a match is found — achievable with 20 lines of Python and the `requests` library. Document the Budibase version in CIS 2.1-aligned software inventory with EOL/patch dates tracked.

Evidence: Retain the following for post-incident review and potential regulatory reporting: (1) Complete access logs covering the full exploitation window (first malicious request to containment) with attacker source IPs and request URIs intact — minimum 12-month retention per NIST AU-11 (Audit Record Retention); (2) Timeline reconstruction document mapping first observed bypass request, any attacker-created accounts or exfiltrated data, lateral movement indicators, and containment timestamp — this is required evidence if PII or regulated data accessible through Budibase apps triggers breach notification obligations; (3) Documented test results from Step 4 validation demonstrating the patch resolves the bypass, retained as evidence of due diligence.

Detection Guidance

Query web access logs for any request URL containing the pattern '/webhooks/' outside of a leading path position, specifically where the pattern appears after a '?' character (e.g., GET /api/users?/webhooks/trigger). Alert on HTTP 200 or 2xx responses to protected API endpoints from sources with no session cookie or Authorization header. Look for anomalous API activity from IPs with no prior authentication history in the Budibase application logs. Monitor the Budibase host for outbound TCP connections on non-standard ports initiated by the application process, which may indicate reverse shell establishment from the published PoC on GitHub. Cross-reference source IPs against threat intelligence feeds for known scanning or exploitation infrastructure. Apply NIST SI-7 (Software, Firmware, and Information Integrity) to detect modifications to Budibase configuration files or authentication databases that may indicate post-exploitation persistence. NIST AU-6 and CIS 8.2 require that these log sources be reviewed and alerting be configured proactively.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`https://github.com/imjdl/CVE-2026-31816-rshell`	Public proof-of-concept reverse shell exploit for CVE-2026-31816, published on GitHub. Presence of requests matching this exploit pattern in logs indicates active exploitation attempt.	HIGH
URL	`?/webhooks/trigger`	Query string injection pattern used to bypass Budibase authorized() middleware. Appearance of this pattern in API request URLs is a strong indicator of exploitation attempt.	HIGH

Framework Mappings

MITRE-ATTACK

- **T1078** — Valid Accounts
- **T1190** — Exploit Public-Facing Application
- **T1059** — Command and Scripting Interpreter

NIST-800-53R5

- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-7** — Software, Firmware, and Information Integrity
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **AC-3** — Access Enforcement
- **SI-10** — Information Input Validation
- **IR-5** — Incident Monitoring

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A03:2021** — Injection

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.8** — Define and Maintain Role-Based Access Control
- **16.10** — Apply Secure Design Principles in Application Architectures
- **6.3** — Require MFA for Externally-Exposed Applications

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

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

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities

NIST-CSF-2

- **DE.AE-08** — Incidents are declared when adverse events meet the defined incident criteria

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1078	Valid Accounts	Defense-Evasion
T1190	Exploit Public-Facing Application	Initial-Access
T1059	Command and Scripting Interpreter	Execution

Sources

Source	URL	Tier
vulncheck_kev	https://nvd.nist.gov/vuln/detail/CVE-2026-31816	T1
CVE-2026-31816: Budibase Auth Bypass Vulnerability - SentinelOne	https://www.sentinelone.com/vulnerability-database/cve-2026-31816/	T3
CVE-2026-31816 - Exploits & Severity - Feedly	https://feedly.com/cve/CVE-2026-31816	T3
CVE-2026-31816 Reverse Shell Exploit - GitHub	https://github.com/imjdl/CVE-2026-31816-rshell	T3
Known Exploited Vulnerabilities Catalog | CISA	https://www.cisa.gov/known-exploited-vulnerabilities-catalog	T1

Generated 2026-06-05 06:56 UTC by TJS Security Command Center