INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-31 13:38 UTC

Gradio Absolute Path Traversal on Windows (Python 3.13+), CVE-2026-28414

CVE VULNERABILITY | HIGH | CVSS 7.5 | CISA KEV

SCC Item ID	SCC-CVE-2026-0243
Type	CVE Vulnerability
CVE ID	CVE-2026-28414
Severity	HIGH
CVSS Base Score	7.5
EPSS Score	0.0421 (89th percentile)
KEV Status	Yes — CISA Known Exploited Vulnerability
Affected Products	gradio_project/gradio < 6.7 (Windows, Python 3.13+)
Published	2026-05-31T00:00:00Z
Discovery Source	Vulncheck Kev

Executive Summary

A path traversal vulnerability in Gradio, a widely used Python library for building AI/ML web interfaces, allows unauthenticated attackers to read arbitrary files from the server's file system on Windows hosts running Python 3.13 or later. Organizations deploying Gradio-based AI applications on Windows are at direct risk of sensitive file exposure, including credentials, configuration files, and proprietary model data. CISA has added this to the Known Exploited Vulnerabilities catalog, confirming active exploitation; immediate patching to Gradio 6.7 is required.

Technical Analysis

CVE-2026-28414 is a path traversal vulnerability (CWE-22) affecting Gradio versions prior to 6.7 on Windows systems running Python 3.13 or later. The root cause is a breaking behavioral change in Python 3.13's os.path.isabs() function: root-relative paths such as /windows/win.ini are no longer treated as absolute paths on Windows. Gradio's path validation logic depended on this behavior to block unsafe file access requests. An unauthenticated remote attacker can craft requests using root-relative path patterns to bypass Gradio's authentication controls entirely and retrieve arbitrary files from the server's file system. MITRE techniques T1083 (File and Directory Discovery) and T1552.001 (Credentials in Files) describe the likely exploitation pattern. CVSS base score is 7.5 (High); EPSS score is 0.042 at the 89th percentile, indicating elevated exploitation likelihood relative to the broader CVE population. CISA KEV listing confirms active exploitation in the wild. The

fix is available in Gradio 6.7. GitHub Advisory: GHSA-39mp-8hj3-5c49. NVD reference: https://nvd.nist.gov/vuln/detail/CVE-2026-28414.

Action Checklist

1. Step 1: Containment. Identify all Windows servers running Gradio < 6.7 with Python 3.13+ using your asset inventory (CIS 1.1). Immediately restrict inbound network access to Gradio endpoints via firewall rules (CIS 4.4) or place them behind an authenticated reverse proxy (D3-PBWSAM) until patching is complete. If internet-facing, take the service offline or block public access entirely pending remediation.
2. Step 2: Detection. Confirm Event Logging (NIST AU-2) controls are enabled and configured to capture HTTP requests and file access events. Verify log retention meets or exceeds 90 days (NIST AU-4). Query web server and application logs for HTTP requests containing root-relative path patterns targeting known sensitive Windows file paths: /windows/win.ini, /windows/system32/drivers/etc/hosts, /../../../ sequences, or any request path starting with a forward slash followed by a Windows directory name. Cross-reference NIST AU-6 review procedures for anomalous file access patterns in Gradio process logs. Alert on HTTP 200 responses to these path patterns as high-confidence exploitation indicators.
3. Step 3: Eradication. Upgrade Gradio to version 6.7 or later on all affected Windows hosts. Follow the official Gradio upgrade path: pip install --upgrade gradio. (On Windows, ensure your Python virtual environment is activated before running pip; if using conda, use: conda update -c conda-forge gradio.) Confirm the installed version post-upgrade. If Python 3.13 is not required, downgrading to Python 3.12 removes the triggering condition but does not substitute for patching. Reference GitHub Advisory GHSA-39mp-8hj3-5c49 for vendor remediation confirmation.
4. Step 4: Recovery. After patching, verify Gradio version is 6.7+ on all hosts. Conduct file access log review (NIST AU-6) for the window between Python 3.13 deployment and patch application to determine whether exploitation occurred. Rotate any credentials, API keys, or secrets stored in files accessible from the Gradio server's file system (D3-CRO). Re-enable external access only after patch verification. Monitor for anomalous outbound connections or lateral movement that may follow credential exposure.
5. Step 5: Post-Incident. Document the control gap: Gradio's path validation relied on Python runtime behavior rather than an independent, version-stable validation library. Review all internally deployed AI/ML web applications for similar dependency on runtime-specific path handling (NIST SI-4 coverage gap). Implement NIST AC-6 (Least Privilege) for Gradio server process accounts to limit the files reachable even if path traversal recurs. Add Gradio and similar AI/ML serving libraries to your software inventory (CIS 2.1) with active patch monitoring. Evaluate whether file access monitoring controls can detect unauthorized reads on sensitive file paths going forward.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO, legal counsel, and privacy officer immediately if retrospective log review confirms HTTP 200 responses to traversal paths targeting files containing PII, PHI, credentials, or proprietary model weights, as CISA KEV listing combined with confirmed exfiltration likely triggers breach notification obligations under HIPAA, state privacy laws, or contractual SLAs; escalate to IR retainer or external DFIR firm if the Gradio host shows post-exploitation indicators such as new outbound connections, lateral movement, or evidence that harvested credentials have been used.
Recovery Notes	After applying the Gradio 6.7+ patch, maintain enhanced logging on the previously exposed hosts for a minimum of 30 days, specifically monitoring for authentication attempts using credentials co-located with the Gradio application at the time of exposure — these may surface days or weeks after the initial traversal if credentials were staged for later use. Verify patch integrity by confirming the Gradio routes.py or path validation module hash matches the published 6.7 release before re-enabling public access. If any cloud provider credentials (AWS, Azure, GCP keys in .env files) were accessible during the exposure window, treat them as fully compromised regardless of log evidence and rotate them before recovery is declared complete.
Forensic Artifacts	IIS W3C access logs or uvicorn/Gradio stdout log file: Filter for HTTP GET requests where the cs-uri-stem field contains '/windows/', 'win.ini', 'system32', 'drivers/etc', or URL-encoded traversal sequences (%2F%2E%2E, %5C); HTTP 200 response codes on these patterns are confirmed exploitation indicators specific to CVE-2026-28414's unauthenticated file read mechanism. | Windows Security Event Log — Event ID 4663 (Object Access): Filter on the Gradio Python process executable (python.exe or the venv python binary) as the Subject, with Object Name falling outside the Gradio application directory — reads of C:\Windows\win.ini, C:\Windows\System32\drivers\etc\hosts, or any .env/.cfg file confirm successful traversal file reads at the OS layer independent of HTTP log completeness. | Windows Security Event Log — Event ID 4688 (Process Creation) and Event ID 4624/4625 (Logon Events): Document the Gradio service account's process lineage and any authentication events from external IPs that correlate temporally with traversal requests, establishing whether exploitation was followed by credential use or lateral movement. | Pip install history and Python package directory timestamps: C:\Users\\AppData\Local\pip\pip.log and filesystem timestamps on C:\Users\\AppData\Local\Programs\Python\Python313\Lib\site-packages\gradio\ — these establish the precise Python 3.13 and Gradio version deployment dates, which define the start of the CVE-2026-28414 exposure window for regulatory breach notification timeline calculations. | Application working directory file enumeration: Snapshot all files in the Gradio app root and subdirectories (Get-ChildItem -Recurse) including .env, *.key, *.pem, *.cfg, *.json, *.yaml files present during the exposure window — this defines the exact set of secrets and data an attacker could have retrieved via traversal and is required to scope any credential rotation and breach notification assessment.

Per-Action IR Details

Step 1: Containment — Identify all Windows servers running Gradio < 6.7 with Python 3.13+ using your asset inventory (CIS 1.1). Immediately restrict inbound network access to Gradio endpoints via firewall rules (CIS 4.4) or place them behind an authenticated reverse proxy (D3-PBWSAM) until patching is complete. If internet-facing, take the service offline or block public access entirely pending remediation.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-4 (Information Flow Enforcement), NIST AC-17 (Remote Access), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Run the following PowerShell one-liner across Windows hosts to identify Python 3.13+ installs and Gradio version simultaneously: `Get-ChildItem -Path 'C:\Users\*\AppData\Local\Programs\Python\Python313*','C:\Python313*' -ErrorAction SilentlyContinue | ForEach-Object { & "$($_.FullName)\python.exe" -c "import gradio; print(gradio.__version__)" 2>$null }`. For network block without enterprise firewall management, use Windows Firewall via netsh: `netsh advfirewall firewall add rule name='Block Gradio Port' dir=in action=block protocol=tcp localport=7860` (default Gradio port). For hosts that cannot be taken offline, front the Gradio process with an nginx reverse proxy requiring HTTP Basic Auth as an immediate unauthenticated-access break.

Evidence: Before isolating, snapshot the current Gradio process network connections using `netstat -anob | findstr ` and capture the full process tree with `Get-WmiObject Win32_Process | Where-Object { $_.CommandLine -like '*gradio*' }` to document the running service account, bound interface, and port. Preserve IIS or Python HTTP server access logs (default locations: IIS — `C:\inetpub\logs\LogFiles\W3SVC*\*.log`; uvicorn/Gradio stdout — captured via Windows Event Log if running as a service, or the console redirect file if launched manually) prior to any service restart that might flush in-memory log buffers.

Step 2: Detection — Query web server and application logs for HTTP requests containing root-relative path patterns targeting known sensitive Windows file paths: /windows/win.ini, /windows/system32/drivers/etc/hosts, /../../../ sequences, or any request path starting with a forward slash followed by a Windows directory name. Review AU-2 (Event Logging) coverage to confirm these request patterns are captured. Cross-reference NIST AU-6 review procedures for anomalous file access patterns in Gradio process logs. Alert on HTTP 200 responses to these path patterns as high-confidence exploitation indicators.

NIST Phase: Detection Analysis

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

Controls: NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-3 (Content of Audit Records), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, run the following PowerShell grep against IIS logs or Gradio stdout log file to surface exploitation attempts: `Select-String -Path 'C:\inetpub\logs\LogFiles\W3SVC*\*.log' -Pattern '(/windows/|win\.ini|system32|drivers/etc|%2F%2E%2E|\.\./)' | Where-Object { $_ -match ' 200 ' }`. For uvicorn-served Gradio, parse the stdout log file (redirect output to a file if not already: `python app.py >> gradio.log 2>&1`). Deploy Sysmon with the SwiftOnSecurity config and query Event ID 4663 (File System Auditing — Object Access) on `C:\Windows\win.ini`, `C:\Windows\System32\drivers\etc\hosts`, and the Gradio working directory to catch file reads that bypassed HTTP-layer logging. A Sigma rule targeting Gradio process reads of `C:\Windows\*` paths via Sysmon Event ID 4663 provides durable detection.

Evidence: The primary forensic signal for CVE-2026-28414 exploitation on Windows is HTTP 200 responses to GET requests containing absolute Windows path patterns in the Gradio file-serving endpoint (typically `/file=` or the Gradio static route). Capture: (1) full IIS or uvicorn access logs with response codes and full request URIs for the period from Python 3.13 deployment to present; (2) Windows Security Event Log Event ID 4663 (An attempt was made to access an object) filtered to the SYSTEM or Gradio service account accessing files outside the Gradio working directory; (3) Windows Security Event ID 4688 (Process Creation) showing the Gradio Python process lineage; (4) any network proxy or WAF logs showing the source IP, user-agent, and timing of anomalous file requests — repeated requests for `win.ini` or `/etc/hosts` from a single external IP within a short window are a high-confidence exploitation signature.

Step 3: Eradication — Upgrade Gradio to version 6.7 or later on all affected Windows hosts. Follow the official Gradio upgrade path: pip install --upgrade gradio. Confirm the installed version post-upgrade. If Python 3.13 is not required, downgrading to Python 3.12 removes the triggering condition but does not substitute for patching. Reference GitHub Advisory GHSA-39mp-8hj3-5c49 for vendor remediation confirmation.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

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), CIS 2.2 (Ensure

Authorized Software is Currently Supported)

Compensating: Execute the upgrade in a Python virtual environment to avoid breaking other application dependencies: `python -m pip install --upgrade gradio` then verify with `python -c "import gradio; print(gradio.__version__)"` — output must be 6.7 or higher. If pip upgrade is blocked by network policy, download the wheel from PyPI on an internet-connected host, transfer via approved channel, and install offline: `pip install gradio-6.7.0-py3-none-any.whl`. Run this verification command across all affected hosts using a simple PowerShell remoting loop if WinRM is available: `Invoke-Command -ComputerName (Get-Content hosts.txt) -ScriptBlock { python -c "import gradio; print($env:COMPUTERNAME, gradio.__version__)" }`. Document the pre- and post-upgrade version and the executing account in your change record.

Evidence: Before executing the upgrade, preserve: (1) output of `pip show gradio` capturing the installed version, install location, and dependencies for the pre-patch state; (2) a filesystem snapshot or hash of the Gradio package directory (`C:\Users\\AppData\Local\Programs\Python\Python313\Lib\site-packages\gradio\`) to confirm which path validation code was in place — specifically the file `routes.py` or equivalent in the Gradio source, which contains the path traversal fix; (3) the running process list and open file handles for the Gradio process at time of eradication, to confirm no active sessions were mid-exploit during the upgrade window.

Step 4: Recovery — After patching, verify Gradio version is 6.7+ on all hosts. Conduct file access log review (NIST AU-6) for the window between Python 3.13 deployment and patch application to determine whether exploitation occurred. Rotate any credentials, API keys, or secrets stored in files accessible from the Gradio server's file system (D3-CRO). Re-enable external access only after patch verification. Monitor for anomalous outbound connections or lateral movement that may follow credential exposure.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST AC-2 (Account Management), NIST AC-3 (Access Enforcement), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-11 (Audit Record Retention), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 6.2 (Establish an Access Revoking Process)

Compensating: To enumerate credentials at risk, run the following against the Gradio working directory and common co-located paths: `Get-ChildItem -Path 'C:\' -Recurse -Include '*.env','*.cfg','*.ini','*.json','*.yaml','*.key','*.pem' | Select-String -Pattern '(password|secret|api_key|token|access_key)' -CaseSensitive:$false`. This surfaces `.env` files, HuggingFace tokens, OpenAI API keys, and cloud provider credentials that are commonly co-located with Gradio AI/ML applications and would be directly readable via this path traversal. Pipe output to a file for remediation tracking. For outbound lateral movement monitoring post-credential exposure, use Wireshark or `netstat -anob` polling every 60 seconds via a scheduled task to detect new outbound connections from the Gradio host.

Evidence: The exploitation window (Python 3.13 deployment date to patch date) defines your retrospective review scope. Collect: (1) IIS or uvicorn access logs for the full window, filtered for HTTP 200 responses to paths containing Windows directory names — these confirm successful file reads, not just attempts; (2) Windows Security Event Log for the Gradio service account showing file read events (Event ID 4663) outside the application directory, particularly reads of `C:\Windows\System32\config\SAM`, `C:\Windows\repair\SAM`, `.env` files, or `*.cfg` files in the app root; (3) any outbound DNS or HTTP connections from the Gradio host to external IPs in the same window, captured from firewall or proxy logs, which would indicate exfiltration following credential theft via this traversal.

Step 5: Post-Incident — Document the control gap: Gradio's path validation relied on Python runtime behavior rather than an independent, version-stable validation library. Review all internally deployed AI/ML web applications for similar dependency on runtime-specific path handling (NIST SI-4 coverage gap). Implement NIST AC-6 (Least Privilege) for Gradio server process accounts to limit the files reachable even if path traversal recurs. Add Gradio and similar AI/ML serving libraries to your software inventory (CIS 2.1) with active patch monitoring. Evaluate whether D3-SFA (System File Analysis) controls can detect unauthorized reads on sensitive file paths going forward.

NIST Phase: Post Incident

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

Controls: NIST IR-4 (Incident Handling), NIST AC-6 (Least Privilege), NIST SI-2 (Flaw Remediation), NIST SI-4 (System Monitoring), NIST RA-3 (Risk Assessment), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Run Gradio (and all AI/ML serving processes — Streamlit, FastAPI inference servers, Jupyter) under a dedicated low-privilege Windows service account using `sc.exe create GradioSvc binPath= '...' obj= '.\gradio_svc'` with explicit NTFS ACL denials on `C:\Windows\System32`, `C:\Users`, and any directory outside the application root: `icacls 'C:\Windows\System32' /deny gradio_svc:(RX)`. Add Gradio, Streamlit, and similar ML-serving packages to an osquery inventory query: `SELECT name, version, path FROM python_packages WHERE name IN ('gradio','streamlit','fastapi');` scheduled daily, with output diff alerting for version changes. Subscribe to the Gradio GitHub Security Advisories RSS feed (github.com/gradio-app/gradio/security/advisories) for zero-lag patch notification without requiring a commercial vulnerability management platform.

Evidence: For the lessons-learned record and to support any regulatory notification assessment, compile: (1) the full HTTP access log extract showing all exploitation attempts and confirmed successful reads (HTTP 200 to traversal paths) with source IPs and timestamps; (2) the list of files confirmed or plausibly accessed based on traversal path patterns in the logs — this directly informs breach notification scope if PII, PHI, or regulated data resided on the filesystem; (3) the output of the credential sweep from Step 4 identifying secrets files present in the Gradio working directory during the exposure window; (4) the Python 3.13 deployment date from Windows Event Log (Event ID 11707, MSI install) or pip install history (`pip show python` or pip log at `C:\Users\\AppData\Local\pip\pip.log`) to precisely bound the exposure window for regulatory purposes.

Detection Guidance

Primary detection method: Parse web server or application access logs for HTTP requests where the URL path begins with a forward slash followed by a Windows directory name, or contains sequences that resolve to sensitive Windows file locations. High-confidence patterns include requests for /windows/win.ini, /windows/system32/, /users/, or /programdata/ with HTTP 200 responses. Secondary indicator: Gradio process file access events (Windows Security Event ID 4663, object access auditing) showing reads of files outside the designated Gradio working directory or allowed_paths configuration. Query example for SIEM (adapt to your log format): source=web_access | where uri_path matches regex '^/[a-zA-Z]+/' AND response_code=200 AND NOT uri_path matches '^/gradio_expected_path'. MITRE T1552.001 hunting: look for file access to known credential storage locations such as .env files, config.ini, application.properties, or AWS/GCP credential files. If Gradio is deployed with specific allowed_paths, any file read outside those paths is an anomaly worth escalating. EPSS at the 89th percentile and CISA KEV status mean opportunistic scanning is likely already underway; treat log gaps as a gap to remediate under NIST AU-5 (Response to Audit Logging Process Failures).

Indicators of Compromise

Type	Value	Context	Confidence
URL	`/windows/win.ini`	Root-relative path pattern used to test or exploit the traversal vulnerability on Windows Gradio servers; HTTP 200 response confirms exploitation success	HIGH

Type	Value	Context	Confidence
URL	`/windows/system32/drivers/etc/hosts`	This IOC is suspicious when observed in HTTP requests to a Gradio application running on Python 3.13+ on Windows, indicating an attempt to exploit path traversal to read the hosts file for reconnaissance or to verify arbitrary file access capability; legitimate applications do not request this system file via web requests, distinguishing this from normal file I/O patterns where the hosts file is accessed only by system processes or administrative tools with local file access.	MEDIUM

Framework Mappings

MITRE-ATTACK

- **T1083** — File and Directory Discovery
- **T1552.001** — Credentials In Files

OWASP-TOP10-2021

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

NIST-800-53R5

- **AC-3** — Access Enforcement
- **SI-10** — Information Input Validation

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **16.12** — Implement Code-Level Security Checks

ISO-27001-2022

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1083	File and Directory Discovery	Discovery
T1552.001	Credentials In Files	Credential-Access

Sources

Source	URL	Tier
vulncheck_kev	https://nvd.nist.gov/vuln/detail/CVE-2026-28414	T1
CVE-2026-28414 Detail - NVD	https://nvd.nist.gov/vuln/detail/cve-2026-28414	T1
CVE-2026-28414: Gradio Path Traversal Vulnerability - SentinelOne	https://www.sentinelone.com/vulnerability-database/cve-2026-28414/	T3
Understanding CVE-2026-28414: Protect Your Server Now - BitNinja	https://bitninja.com/blog/understanding-cve-2026-28414-protect-your...	T3
CVE-2026-28414 · GitHub Advisory Database	https://github.com/advisories/GHSA-39mp-8hj3-5c49	T3
CISA KEV	https://www.cisa.gov/known-exploited-vulnerabilities-catalog	T1

Generated 2026-05-31 13:38 UTC by TJS Security Command Center