INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-27 14:07 UTC

CVE-2026-9552, CVE-2026-9551, CVE-2026-9550: Critical/Severe Security Advisories in Das Parking Management System

CVE VULNERABILITY | **CRITICAL** | CVSS 9.8

SCC Item ID	SCC-CVE-2026-0231
Type	CVE Vulnerability
CVE ID	CVE-2026-9552, CVE-2026-9551, CVE-2026-9550
Severity	CRITICAL
CVSS Base Score	9.8
EPSS Score	0.0001 (1th percentile)
Affected Products	Das Parking Management System (■■■■■■■) 6.2.0
Published	2026-05-26
Discovery Source	Gemini

Executive Summary

Three unpatched vulnerabilities in Das Parking Management System 6.2.0 expose the platform to potential operating system-level command execution through its Search API and a separate API endpoint that interacts with SQL Server's xp_cmdshell function. Organizations running this system face risk of full server compromise, including data theft, ransomware deployment, and operational disruption to parking operations. CVSS 9.8 and active exploitation status are not yet confirmed by NVD; however, the attack class (SQL/argument injection leading to OS command execution) warrants immediate containment action.

Technical Analysis

Three CVEs (CVE-2026-9552, CVE-2026-9551, CVE-2026-9550) affect Das Parking Management System version 6.2.0. Two distinct attack surfaces are identified: (1) a Search API Endpoint susceptible to injection (CWE-89: SQL Injection, CWE-88: Argument Injection), and (2) an API endpoint that invokes xp_cmdshell, a SQL Server extended stored procedure that executes OS-level commands directly from within SQL Server. Exploitation of xp_cmdshell via injection flaws maps to MITRE ATT&CK T1190 (Exploit Public-Facing Application) and T1059.003 (Command and Scripting Interpreter: Windows Command Shell). Successful exploitation could yield unauthenticated remote code execution at the SQL Server service account privilege

level. CVE records were not fully populated in NVD at analysis time; the CVSS 9.8 base score is sourced from discovery analysis and is unconfirmed by NVD or the vendor. EPSS score is 0.009% (9e-05), placing this in the 1st percentile for exploitation likelihood; no active exploitation has been reported. No official vendor patch or advisory has been identified in available sources. Affected version: 6.2.0 only (other versions unconfirmed).

Action Checklist

1. Step 1: Containment, Immediately restrict network access to the Das Parking Management System 6.2.0 API endpoints (Search API and the xp_cmdshell-adjacent endpoint) at the perimeter firewall and any internal network segments. If the system is internet-facing, take it offline or place it behind an IP allowlist until a vendor patch is available. Apply NIST AC-4 (Information Flow Enforcement) to isolate the SQL Server instance from untrusted networks.
2. Step 2: Detection, Review SQL Server logs and application logs for anomalous calls to xp_cmdshell, unexpected stored procedure executions, or unusual query patterns against the Search API (look for SQL metacharacters: single quotes, UNION, EXEC, xp_cmdshell in request parameters). Check Windows Event Logs (Event ID 4688, process creation) for child processes spawned by the SQL Server service account. Query IDS/IPS logs for T1190 signatures. Enable CIS 8.2 (Collect Audit Logs) if SQL Server and application logging are not already centralized.
3. Step 3: Eradication, Disable xp_cmdshell on the SQL Server instance immediately: execute 'sp_configure xp_cmdshell, 0; RECONFIGURE;' in SQL Server Management Studio. Verify the change with 'SELECT value FROM sys.configurations WHERE name = xp_cmdshell'. Apply input validation and parameterized queries at the API layer. Monitor the vendor (Das Parking Management System) for an official patch to version 6.2.0; no confirmed patch was available at analysis time. Apply NIST SI-10 (Information Input Validation) principles when evaluating any vendor-supplied fix.
4. Step 4: Recovery, After disabling xp_cmdshell and restricting API access, confirm SQL Server agent jobs and application functions dependent on xp_cmdshell have been replaced or disabled without breaking legitimate operations. Conduct a post-containment review of SQL Server audit logs (NIST AU-6: Audit Record Review, Analysis, and Reporting) to determine if exploitation occurred prior to containment. Validate firewall rules are enforced and no lateral movement artifacts exist on the hosting system (check scheduled tasks, new local accounts, modified startup configs per NIST SI-7: Software, Firmware, and Information Integrity).
5. Step 5: Post-Incident, Document the control gap: API endpoints with direct SQL Server stored procedure access represent an architectural risk. Require the vendor to provide a security assessment of all API endpoints before redeployment. Implement NIST AC-6 (Least Privilege) to ensure the SQL Server service account operates under least-privilege principles, preventing OS-level damage even if injection occurs. Add this vulnerability class to recurring threat-hunt hypotheses and update detection rules for xp_cmdshell invocation patterns.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to senior IR leadership and legal/compliance if Windows Security Event Log (Event ID 4688) or SQL Server default trace confirms cmd.exe, powershell.exe, or net.exe was spawned by the SQL Server service account (indicating confirmed xp_cmdshell exploitation), or if any outbound network connections from the SQL Server host to external IPs are identified during the compromise window, as these conditions indicate active post-exploitation and potential data exfiltration from the Das Parking Management System database — which may contain PII (vehicle registration, payment records) triggering breach notification obligations under applicable data protection regulations.
Recovery Notes	Before returning Das Parking Management System 6.2.0 to production, verify through SQL Server sys.configurations that xp_cmdshell value_in_use is 0, confirm firewall ACLs blocking direct external access to the API and SQL Server port 1433 are persistent across reboots, and validate no attacker-created SQL logins, scheduled tasks, or startup registry entries remain on the host. Monitor SQL Server ERRORLOG and Windows Security Event Log (Event ID 4688 filtered on sqlservr.exe parent) continuously for a minimum of 30 days post-recovery, as threat actors who achieved OS-level execution via CVE-2026-9552 may have implanted secondary persistence mechanisms that survive xp_cmdshell disablement. Do not redeploy the Das Parking Management System API to internet-facing infrastructure until the vendor provides a patched release and a written attestation that parameterized queries replace direct stored procedure invocation in the Search API and xp_cmdshell-adjacent endpoint.
Forensic Artifacts	SQL Server default trace files (.trc) located via 'SELECT path FROM sys.traces WHERE is_default = 1' — retain all .trc files from the Das Parking Management System database server for the 30 days preceding containment; these record xp_cmdshell enablement/execution events, DDL changes (new stored procedures, new logins), and failed login attempts that map directly to CVE-2026-9552/9551 exploitation attempts | Das Parking Management System application/web server access logs (IIS logs at C:\inetpub\logs\LogFiles\ or Apache/Nginx equivalent) — filter on the Search API endpoint URI for requests containing URL-encoded SQL metacharacters (%27 for single-quote, UNION, EXEC, xp_cmdshell) which represent the injection payload delivery specific to CVE-2026-9550/9551/9552 | Windows Security Event Log (Event ID 4688 — Process Creation) from the SQL Server host, filtered on ParentProcessName matching sqlservr.exe — cmd.exe, powershell.exe, net.exe, certutil.exe, or mshta.exe as child processes are direct forensic indicators of successful OS command execution via xp_cmdshell exploitation of this vulnerability | SQL Server sys.server_principals and sys.sql_logins snapshots — query 'SELECT name, create_date, modify_date, is_disabled FROM sys.server_principals ORDER BY create_date DESC' to identify any backdoor SQL or Windows logins created by an attacker who achieved sysadmin-level access through xp_cmdshell elevation following exploitation of the Das Parking Management System Search API injection vulnerability | Windows Scheduled Tasks export ('schtasks /query /fo CSV /v > schtasks_export.csv') and HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run registry hive from the Das Parking Management System server — attackers with OS-level command execution via xp_cmdshell commonly establish persistence through scheduled tasks or run-key entries, and these artifacts would survive a SQL Server service restart or xp_cmdshell disablement

Per-Action IR Details

Step 1: Containment — Immediately restrict network access to the Das Parking Management System 6.2.0 API endpoints (Search API and the xp_cmdshell-adjacent endpoint) at the perimeter firewall and any internal network segments. If the system is internet-facing, take it offline or place it behind an IP allowlist until a vendor patch is available. Apply NIST AC-4 (Information Flow Enforcement) to isolate the SQL Server instance

from untrusted networks.

NIST Phase: Containment

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

Controls: NIST AC-4 (Information Flow Enforcement), NIST SC-7 (Boundary Protection), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 12.2 — Establish and Maintain a Secure Network Architecture (block inbound access to SQL Server port 1433 and the parking system API port from untrusted zones)

Compensating: On the Windows host running Das Parking Management System 6.2.0, use Windows Firewall (netsh advfirewall) to immediately block inbound TCP on the API listener port and SQL Server port 1433: 'netsh advfirewall firewall add rule name="Block DasPark API" dir=in action=block protocol=tcp localport=' and 'netsh advfirewall firewall add rule name="Block MSSQL External" dir=in action=block protocol=tcp localport=1433'. On the perimeter, add an ACL or iptables rule blocking all source IPs except explicitly allowlisted management IPs. A 2-person team can execute both within 10 minutes without enterprise tooling.

Evidence: Before modifying firewall rules, capture current netstat output ('netstat -anob > netstat_precontainment.txt') to preserve evidence of any active connections to the Das Parking Management System API or SQL Server port 1433 at time of containment. Collect the Windows Firewall log (C:\Windows\System32\LogFiles\Firewall\pfirewall.log) for inbound connection history to the API and SQL ports. Export IDS/IPS connection logs filtered on the parking system server IP for the 30 days preceding containment to establish a pre-compromise connection baseline.

Step 2: Detection — Review SQL Server logs and application logs for anomalous calls to xp_cmdshell, unexpected stored procedure executions, or unusual query patterns against the Search API (look for SQL metacharacters: single quotes, UNION, EXEC, xp_cmdshell in request parameters). Check Windows Event Logs (Event ID 4688 — process creation) for child processes spawned by the SQL Server service account. Query IDS/IPS logs for T1190 signatures. Enable CIS 8.2 (Collect Audit Logs) if SQL Server and application logging are not already centralized.

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 SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs), MITRE ATT&CK T1190 (Exploit Public-Facing Application), MITRE ATT&CK T1059.003 (Windows Command Shell via xp_cmdshell), MITRE ATT&CK T1505.001 (SQL Stored Procedures)

Compensating: Enable SQL Server Audit or C2 audit trace if not active: execute 'EXEC sp_configure "c2 audit mode", 1; RECONFIGURE;' — this logs all T-SQL statements including xp_cmdshell invocations to the SQL Server ERRORLOG directory. Install Sysmon (free, Sysinternals) with a config that captures ProcessCreate events (Event ID 1) filtering on ParentImage matching the SQL Server executable (sqlservr.exe) to catch cmd.exe, powershell.exe, or net.exe spawned as child processes. Use this PowerShell one-liner to scan existing Security Event Log for SQL Server service account spawning shells: 'Get-WinEvent -LogName Security | Where-Object {$_.Id -eq 4688 -and $_.Message -match "sqlservr"}'. Deploy a Sigma rule for xp_cmdshell detection against Windows Event Log (Sigma rule 'proc_creation_win_mssql_xp_cmdshell_spawn' from SigmaHQ) converted to a PowerShell query or Winlogbeat format.

Evidence: Collect SQL Server ERRORLOG files (default path: C:\Program Files\Microsoft SQL Server\MSSQL.MSSQLSERVER\MSSQL\Log\ERRORLOG*) and SQL Server default trace files (.trc) from the Das Parking Management System database server — these record stored procedure executions including xp_cmdshell calls with timestamps and source SPIDs. Export Das Parking Management System web/application server access logs covering all requests to the Search API endpoint, filtering on URI patterns containing SQL metacharacters (%27, UNION, EXEC, xp_cmdshell). Pull Windows Security Event Log (Event ID 4688) from the SQL Server host for the prior 30 days filtered on processes spawned by the SQL Server service account (typically NT SERVICE\MSSQLSERVER or a dedicated domain account) — cmd.exe, powershell.exe, net.exe, or certutil.exe as child processes are high-fidelity indicators of successful xp_cmdshell exploitation.

Step 3: Eradication — Disable xp_cmdshell on the SQL Server instance immediately: execute 'sp_configure xp_cmdshell, 0; RECONFIGURE;' in SQL Server Management Studio. Verify the change with 'SELECT value FROM sys.configurations WHERE name = xp_cmdshell'. Apply input validation and parameterized queries at

the API layer. Monitor the vendor (Das Parking Management System) for an official patch to version 6.2.0 — no confirmed patch was available at analysis time. Apply NIST SI-10 (Information Input Validation) principles when evaluating any vendor-supplied fix.

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), NIST AC-6 (Least Privilege), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Beyond disabling xp_cmdshell, harden the SQL Server instance by revoking EXECUTE permission on xp_cmdshell from all non-sysadmin logins: 'REVOKE EXECUTE ON xp_cmdshell TO PUBLIC;'. Additionally, rename or disable the SQL Server Agent if it is not operationally required, as it provides an alternative OS command execution path. For the Das Parking Management System API layer — if source access is unavailable — deploy a free WAF rule using ModSecurity (CRS ruleset) or Nginx with a Lua filter that rejects any Search API request body or query parameter containing: single-quote sequences, UNION SELECT patterns, EXEC(, or the literal string xp_cmdshell. Document all changes with timestamps for the post-incident report, as no vendor patch for version 6.2.0 was confirmed available at time of analysis.

Evidence: Before disabling xp_cmdshell, capture the current SQL Server configuration state: 'SELECT name, value, value_in_use FROM sys.configurations WHERE name IN (''xp_cmdshell'', ''Ole Automation Procedures'', ''clr enabled'')' — document all enabled OS-interaction features, as attackers may have enabled additional execution paths (Ole Automation, CLR) alongside xp_cmdshell. Review SQL Server startup stored procedures ('SELECT * FROM sys.objects WHERE type = ''P'' AND is_ms_shipped = 0') for any attacker-created persistence mechanisms that auto-execute on service restart. Check for new SQL Server logins created during the suspected compromise window: 'SELECT name, create_date, modify_date, type_desc FROM sys.server_principals WHERE create_date > '.

Step 4: Recovery — After disabling xp_cmdshell and restricting API access, confirm SQL Server agent jobs and application functions dependent on xp_cmdshell have been replaced or disabled without breaking legitimate operations. Conduct a post-containment review of SQL Server audit logs (NIST AU-6: Audit Record Review, Analysis, and Reporting) to determine if exploitation occurred prior to containment. Validate firewall rules are enforced and no lateral movement artifacts exist on the hosting system (check scheduled tasks, new local accounts, modified startup configs per D3-SICA).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST CP-10 (System Recovery and Reconstitution), NIST CM-6 (Configuration Settings), NIST IR-4 (Incident Handling), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 5.3 (Disable Dormant Accounts)

Compensating: Run the following PowerShell commands on the Das Parking Management System host to enumerate lateral movement artifacts without EDR: (1) Scheduled tasks added after the suspected exploitation date: 'Get-ScheduledTask | Where-Object {$_.Date -gt ""} | Select TaskName, TaskPath, Date'. (2) New local accounts: 'Get-LocalUser | Select Name, Enabled, LastLogon, PasswordLastSet'. (3) Startup registry modifications: 'Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'. (4) New services: 'Get-WinEvent -LogName System | Where-Object {$_.Id -eq 7045}' (Event ID 7045 = new service installed). Cross-reference all findings against a known-good baseline from backup or change management records for the parking system server.

Evidence: Prior to restoring normal operations, acquire a full forensic image or at minimum a targeted triage collection from the Das Parking Management System server using a free tool such as KAPE (Kroll Artifact Parser and Extractor) targeting the MSSQL log directory, Windows Event Logs, scheduled tasks, startup locations, and the Das Parking Management System application directory. Collect the SQL Server default trace (SELECT * FROM fn_trace_getinfo(NULL) to locate .trc files) which retains a rolling history of DDL changes, login events, and stored procedure executions — this is the primary artifact for determining whether xp_cmdshell was invoked before containment. Hash all collected artifacts (SHA-256) and store offline before proceeding with system changes.

Step 5: Post-Incident — Document the control gap: API endpoints with direct SQL Server stored procedure access represent an architectural risk. Require the vendor to provide a security assessment of all API endpoints before redeployment. Implement D3-UAP (User Account Permissions) to ensure the SQL Server service account operates under least privilege (NIST AC-6), preventing OS-level damage even if injection occurs. Add this vulnerability class to recurring threat-hunt hypotheses and update detection rules for xp_cmdshell invocation patterns.

NIST Phase: Post Incident

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

Controls: NIST AC-6 (Least Privilege), NIST IR-4 (Incident Handling), NIST RA-3 (Risk Assessment), NIST CA-7 (Continuous Monitoring), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Create a persistent Sysmon + Windows Event Forwarding (WEF) detection for xp_cmdshell re-enablement: configure a WEF subscription on the SQL Server host to forward Event ID 4688 (process creation) where ParentImage contains sqlservr.exe to a central log collector (even a simple Windows Server running WEF can serve as a no-cost SIEM substitute). Write a scheduled SQL Server Agent alert (or a Windows Scheduled Task using sqlcmd) that queries 'SELECT value_in_use FROM sys.configurations WHERE name = "xp_cmdshell"' every 15 minutes and emails the DBA team if the result is non-zero — this detects re-enablement by an attacker or accidental re-activation. Add a Sigma rule for this specific vulnerability class (SQL injection leading to xp_cmdshell) to your detection backlog using the community rule 'proc_creation_win_mssql_xp_cmdshell_spawn' from the SigmaHQ repository.

Evidence: Compile the complete post-incident evidence package for the lessons-learned review: (1) timeline of all SQL Server ERRORLOG and default trace entries showing first and last xp_cmdshell invocation; (2) full list of Search API requests from application logs containing SQL metacharacters sorted by source IP; (3) diff of SQL Server configuration before and after incident (sys.configurations snapshot); (4) inventory of any new OS accounts, scheduled tasks, or services created on the parking system host during the compromise window; (5) network flow data showing any outbound connections from the SQL Server host to external IPs during the exploitation window — these represent potential data exfiltration or C2 callback artifacts specific to post-exploitation activity following CVE-2026-9552/9551/9550 exploitation.

Detection Guidance

Primary detection target: unauthorized or anomalous xp_cmdshell execution within SQL Server. Query SQL Server error log and default trace for xp_cmdshell calls: 'SELECT * FROM sys.traces' and review trace files for stored procedure executions. Enable SQL Server Audit if not active (NIST AU-2, AU-12). In SIEM, alert on Windows Event ID 4688 where ParentProcessName contains 'sqlservr.exe' and NewProcessName is cmd.exe or powershell.exe. Search web/application access logs for API requests to the Search endpoint containing SQL metacharacters (', --, ;, EXEC, UNION, xp_). For system file analysis, monitor for new files written to directories writable by the SQL Server service account. Severity is critical based on impact (unauthenticated remote code execution, full server compromise) not current exploitation likelihood. EPSS 0.009% (1st percentile) indicates no known active exploitation, but vulnerability class warrants high detection priority. In the absence of confirmed IOCs (no threat actor attribution, no public PoC identified at analysis time), behavioral detection on xp_cmdshell invocation is the highest-confidence signal.

Framework Mappings

MITRE-ATTACK

- **T1190** — Exploit Public-Facing Application
- **T1059.003** — Windows Command Shell

NIST-800-53R5

- **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
- **SI-10** — Information Input Validation
- **IR-5** — Incident Monitoring

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures

ISO-27001-2022

- **A.8.28** — Secure coding
- **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
T1190	Exploit Public-Facing Application	Initial-Access
T1059.003	Windows Command Shell	Execution

Sources

Source	URL	Tier
gemini	https://cve.report/CVE-2026-9552	T3
CVE-2026-9552 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-9552	T1

Source	URL	Tier
CVE Threat Database | Real-Time Security Insights	https://cve.akaoma.com/	T3
CVE-2026-9550 Detail - NVD	https://nvd.nist.gov/vuln/detail/cve-2026-9550	T1
CVE-2026-5435 - Red Hat Customer Portal	https://access.redhat.com/security/cve/cve-2026-5435	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-9552, CVE-2026-9551, CVE-...	T1

Generated 2026-05-27 14:07 UTC by TJS Security Command Center