INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-22 19:01 UTC

# ShapedPlugin Build Pipeline Compromised, Backdoor Injected into Three WordPress Pro Plugins via Official Update Channel

**THREAT CAMPAIGN** | **CRITICAL** | CVSS 9.5

| | |
|---|---|
| **SCC Item ID** | SCC-CAM-2026-0538 |
| **Type** | Threat Campaign |
| **CVE ID** | CVE-2026-49777, CVE-2026-10735 |
| **Severity** | CRITICAL |
| **CVSS Base Score** | 9.5 |
| **EPSS Score** | 0.0124 (65th percentile) |
| **Affected Products** | ShapedPlugin Product Slider Pro for WooCommerce (before 3.5.4), Real Testimonials Pro (3.2.5), Smart Post Show Pro (before 4.0.2); distributed via Easy Digital Downloads / account.shapedplugin.com |
| **Published** | 2026-06-22T14:00:48 |
| **Discovery Source** | Rss |

## Executive Summary

Unknown threat actors compromised ShapedPlugin's software build and distribution pipeline, injecting backdoor code into three paid WordPress plugins delivered through the vendor's own licensed update system. Any WordPress site that updated Product Slider Pro for WooCommerce (before 3.5.4), Real Testimonials Pro (3.2.5), or Smart Post Show Pro (before 4.0.2) through official channels may have received a backdoored version capable of stealing credentials, exfiltrating database configuration secrets, deploying web shells, and bypassing two-factor authentication. The attack targets paying customers who followed best practices by updating through an authenticated vendor channel, making standard update-trust assumptions unreliable for this vendor until the pipeline is confirmed clean.

## Technical Analysis

ShapedPlugin's Easy Digital Downloads build and distribution pipeline was compromised by an unknown threat actor who injected malicious code into Pro versions of three WordPress plugins before delivery through account.shapedplugin.com. Affected versions: Product Slider Pro for WooCommerce (before 3.5.4), Real Testimonials Pro (3.2.5), Smart Post Show Pro (before 4.0.2). The backdoor capabilities documented in source

material include: credential harvesting (T1056.001, T1552.001), wp-config.php exfiltration including database credentials (T1041), web shell deployment (T1505.003), 2FA bypass (T1556), JavaScript keylogging (T1059.007), and log tampering for defense evasion (T1070.004). The supply chain compromise vector (T1195.002) is the critical distinction, malicious code arrived via the vendor's authenticated update mechanism, not through a vulnerability in the plugins themselves. Relevant CWEs: CWE-494 (Download of Code Without Integrity Check), CWE-506 (Embedded Malicious Code), CWE-312 (Cleartext Storage of Sensitive Information), CWE-522 (Insufficiently Protected Credentials). CVE attribution carries LOW confidence: CVE-2026-49777 does not conform to a recognized indexed format; CVE-2026-10735 is cited via Tenable but cannot be independently verified against NVD; the NVD source URL in the pipeline references CVE-2026-10777, indicating a possible identifier mismatch in upstream data. Treat CVE IDs as provisional pending NVD confirmation. No confirmed threat actor attribution is available.

## Action Checklist

1. Step 1: Containment, immediately disable or quarantine all installations of Product Slider Pro for WooCommerce (versions before 3.5.4), Real Testimonials Pro (version 3.2.5), and Smart Post Show Pro (versions before 4.0.2) on any WordPress site updated through account.shapedplugin.com. If deactivation is not immediately possible, block outbound connections from affected WordPress servers to unknown external IPs at the perimeter while investigation proceeds.
2. Step 2: Detection, audit WordPress file integrity on affected sites: compare plugin file hashes against known-good versions from ShapedPlugin's official repository. Search web server access logs and WordPress debug logs for unexpected outbound POST requests, access to wp-config.php from plugin contexts, creation of new PHP files in plugin directories, and admin account creation events not initiated by authorized personnel. Review authentication logs for 2FA bypass indicators and unexpected admin-level logins (per NIST AU-6 for audit record analysis). Ensure WordPress debug logging is enabled per NIST AU-2 (Event Logging) to capture plugin-level activity. Check for new or modified files in wp-content/plugins/product-slider-pro/, wp-content/plugins/real-testimonials-pro/, and wp-content/plugins/smart-post-show-pro/ with timestamps matching the update window.
3. Step 3: Eradication, update to clean versions: Product Slider Pro for WooCommerce 3.5.4 or later, Smart Post Show Pro 4.0.2 or later. For Real Testimonials Pro 3.2.5 (listed as affected with no clean version specified in source data), contact ShapedPlugin directly to confirm a remediated release is available. If no clean version is available, remove the plugin entirely and evaluate alternative WordPress testimonial solutions. Do not reinstall until a verified clean version is confirmed by the vendor. Remove all plugin files from the compromised versions and reinstall from a verified clean source. Rotate all WordPress admin credentials and database passwords found in wp-config.php on affected sites. Revoke and regenerate any API keys or secrets stored in the WordPress configuration. Apply credential rotation across all accounts that may have been exposed.
4. Step 4: Recovery, after installing clean plugin versions, verify file integrity of the full WordPress installation, not only the three affected plugins, as web shell deployment (T1505.003) may have introduced persistence elsewhere in the file system. Re-enable the site only after confirming no unauthorized admin accounts exist (CIS 5.1, Establish and Maintain an Inventory of Accounts), no unexpected PHP files are present outside normal plugin structures, and outbound traffic patterns have returned to baseline. Monitor authentication logs for 30 days post-remediation for signs of persistent access using credentials harvested before remediation. Reference NIST IR controls for formal incident closure criteria.

**5.** Step 5: Post-Incident, this attack exposed a gap in third-party plugin update verification: sites had no mechanism to validate the integrity of updates delivered through an authenticated vendor channel. Implement file integrity monitoring for all WordPress plugin directories. Evaluate deployment of a WordPress security plugin or server-side WAF rule set that alerts on new PHP file creation in plugin directories. Review vendor update trust policies: consider staging updates in a non-production environment before deploying to production for all paid plugins. Reference NIST CM-3 (Configuration Change Management) for vendor update staging procedures and CIS 4.1 (Establish and Maintain a Software Inventory) for third-party plugin risk assessment. Document lessons learned per NIST IR-4 (Incident Handling).

## IR / Forensic Enrichment

| | |
|---|---|
| **Triage Priority** | IMMEDIATE |
| **Escalation Criteria** | Escalate immediately to legal counsel and data protection officer if forensic analysis of web server logs or the WordPress database confirms that the backdoor successfully exfiltrated wp-config.php credentials, customer PII, or WooCommerce order/payment data, as this may trigger breach notification obligations under GDPR, CCPA, or PCI DSS depending on the site's data scope; also escalate if unauthorized admin accounts are discovered that predate the containment action, indicating the attacker achieved persistent access before isolation. |
| **Recovery Notes** | After reinstalling clean plugin versions (Product Slider Pro for WooCommerce 3.5.4+, Smart Post Show Pro 4.0.2+, and a ShapedPlugin-confirmed clean build of Real Testimonials Pro), perform a full WordPress core checksum verification via `wp core verify-checksums --path=/var/www/html` and a full plugin directory file integrity scan before returning the site to production. Monitor wp-login.php authentication logs, WordPress admin account creation events, and outbound HTTP POST traffic from the web server process daily for a minimum of 30 days, as credentials harvested by the backdoor prior to containment may be leveraged in delayed credential-stuffing or database attacks. Confirm with ShapedPlugin that CVE-2026-49777 and CVE-2026-10735 are addressed in the clean releases before closing the incident. |
| **Forensic Artifacts** | wp-content/plugins/product-slider-pro/, real-testimonials-pro/, and smart-post-show-pro/ directory trees: PHP files with modification timestamps falling within the account.shapedplugin.com update delivery window are the primary indicator of injected backdoor code; preserve full directory archives with inode metadata before any eradication action. \| WordPress database wp_users and wp_usermeta tables: the backdoor's admin account creation capability would insert unauthorized administrator-role users; dump and preserve the full table with row creation timestamps to establish whether attacker-controlled accounts were provisioned before containment. \| Web server access logs (Apache access.log / Nginx access.log) filtered for outbound POST requests to non-ShapedPlugin external IPs originating from PHP processes, and for requests accessing wp-config.php directly from plugin execution context — these indicate active credential exfiltration by the backdoor. \| wp-config.php snapshot with file modification timestamp and SHA-256 hash: the backdoor's described capability to steal database configuration secrets means the file's last-modified time establishes whether it was read or altered during the compromise window; preserve before any credential rotation. \| WordPress debug.log (if WP_DEBUG_LOG is enabled) and PHP error logs: injected backdoor PHP executing in plugin context will generate error or notice entries referencing the malicious file paths inside product-slider-pro/, real-testimonials-pro/, or smart-post-show-pro/ directories, providing execution evidence independent of the web server access log. |

### Per-Action IR Details

**Step 1: Containment — immediately disable or quarantine all installations of Product Slider Pro for WooCommerce (versions before 3.5.4), Real Testimonials Pro (version 3.2.5), and Smart Post Show Pro (versions before 4.0.2) on any WordPress site updated through account.shapedplugin.com. If deactivation is not immediately possible, block outbound connections from affected WordPress servers to unknown external IPs at the perimeter while investigation proceeds.**

**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)

**Compensating:** On Linux hosts, use iptables to block all non-essential outbound traffic immediately: `iptables -A OUTPUT -m owner --uid-owner www-data -j DROP` (adjust UID to the web server user). On WordPress sites without server access, deactivate all three plugins via wp-cli: `wp plugin deactivate product-slider-pro real-testimonials-pro smart-post-show-pro --path=/var/www/html`. Document the exact timestamp of deactivation for forensic chain of custody.

**Evidence:** Before deactivating or blocking: capture active outbound network connections from the web server process using `ss -tulnp | grep php` or `netstat -anp | grep -E 'php|apache|nginx'` to identify any live C2 channels the backdoor may have already established. Capture running PHP process list (`ps aux | grep php`) and any PHP-FPM worker state. Export current WordPress option table entries via `wp option list --path=/var/www/html > wp_options_snapshot.txt` — backdoor code may have written callback URLs or encoded payloads into wp_options. These connection states and option values are volatile and lost once the plugin is deactivated or the server process is interrupted.

**Step 2: Detection — audit WordPress file integrity on affected sites: compare plugin file hashes against known-good versions from ShapedPlugin's official repository. Search web server access logs and WordPress debug logs for unexpected outbound POST requests, access to wp-config.php from plugin contexts, creation of new PHP files in plugin directories, and admin account creation events not initiated by authorized personnel. Review authentication logs for 2FA bypass indicators and unexpected admin-level logins. Check for new or modified files in wp-content/plugins/product-slider-pro/, wp-content/plugins/real-testimonials-pro/, and wp-content/plugins/smart-post-show-pro/ with timestamps matching the update window. Reference NIST AU-6 (Audit Record Review, Analysis, and Reporting) and AU-2 (Event Logging) for log scope guidance.**

**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:** Run `find /var/www/html/wp-content/plugins/product-slider-pro /var/www/html/wp-content/plugins/real-testimonials-pro /var/www/html/wp-content/plugins/smart-post-show-pro -type f -name '*.php' -newer /var/www/html/wp-config.php -exec ls -la {} \;` to surface PHP files modified after the last known-good state. Generate SHA-256 hashes of all plugin files: `find /var/www/html/wp-content/plugins/ -name '*.php' | xargs sha256sum > current_hashes.txt` and diff against hashes from a clean reference copy. Search Apache/Nginx access logs for outbound indicators: `grep -E 'POST.*(wp-admin|wp-login|xmlrpc)' /var/log/apache2/access.log | grep -v '192\.168\.'`. Use wp-cli to audit admin accounts: `wp user list --role=administrator --path=/var/www/html`.

**Evidence:** Before any file modification or plugin removal: preserve a forensic copy of all three plugin directories using `tar czf plugin_evidence_$(date +%Y%m%d_%H%M%S).tar.gz /var/www/html/wp-content/plugins/product-slider-pro/ /var/www/html/wp-content/plugins/real-testimonials-pro/ /var/www/html/wp-content/plugins/smart-post-show-pro/` and record SHA-256 of the archive. Capture web server access logs and error logs in their current state before any log rotation occurs. Dump the WordPress database for offline analysis: `wp db export evidence_db_$(date +%Y%m%d).sql --path=/var/www/html`. The backdoor's credential-harvesting activity and any injected admin accounts will be visible in the database and logs but may be overwritten by normal site operation.

**Step 3: Eradication — update to clean versions: Product Slider Pro for WooCommerce 3.5.4 or later, Smart Post Show Pro 4.0.2 or later. For Real Testimonials Pro 3.2.5 (listed as affected with no clean version specified in source data), contact ShapedPlugin directly to confirm remediated release availability before reinstalling. Remove all plugin files from the compromised versions and reinstall from a verified clean source. Rotate all WordPress admin credentials and database passwords found in wp-config.php on affected sites. Revoke and regenerate any API keys or secrets stored in the WordPress configuration. Apply D3-CRO (Credential Rotation) across all accounts that may have been exposed.**

**NIST Phase:** Eradication

**Reference:** NIST 800-61r3 §3.4 — Eradication

**Controls:** NIST AC-2 (Account Management), NIST AC-3 (Access Enforcement), NIST IA (Identification and Authentication) — [no mapped control in knowledge base for credential rotation specifically; AC-2 governs account remediation], CIS 5.2 (Use Unique Passwords), CIS 6.2 (Establish an Access Revoking Process)

**Compensating:** Remove backdoored plugin files completely before reinstalling: `rm -rf /var/www/html/wp-content/plugins/product-slider-pro /var/www/html/wp-content/plugins/smart-post-show-pro` then reinstall only after verifying the downloaded package SHA-256 matches ShapedPlugin's published hash for version 3.5.4 / 4.0.2. Rotate the WordPress database password in wp-config.php and update it in MySQL: `ALTER USER 'wp_user'@'localhost' IDENTIFIED BY 'new_strong_password';`. Force-reset all admin passwords: `wp user update $(wp user list --role=administrator --field=ID --path=/var/www/html) --user_pass='new_secure_password' --path=/var/www/html`. Regenerate WordPress secret keys by replacing the AUTH_KEY, SECURE_AUTH_KEY, and LOGGED_IN_KEY values in wp-config.php with fresh values from https://api.wordpress.org/secret-key/1.1/salt/.

**Evidence:** Before rotating credentials or patching: confirm the forensic database dump from Step 2 is complete and intact — wp-config.php database credentials, any stored API keys, and the wp_users table (which may contain backdoor-injected admin accounts) must be preserved in evidence before overwriting. Verify that the memory and network snapshots from Step 1 containment are secured. If any PHP webshell files were identified in Step 2 detection, preserve their content and inode metadata (`stat `) before removal, as timestamp and ownership data establish when the backdoor first wrote persistence to disk.

**Step 4: Recovery — after installing clean plugin versions, verify file integrity of the full WordPress installation, not only the three affected plugins, as web shell deployment (T1505.003) may have introduced persistence elsewhere in the file system. Re-enable the site only after confirming no unauthorized admin accounts exist (CIS 5.1 — Establish and Maintain an Inventory of Accounts), no unexpected PHP files are present outside normal plugin structures, and outbound traffic patterns have returned to baseline. Monitor authentication logs for 30 days post-remediation for signs of persistent access using credentials harvested before remediation. Reference NIST IR controls for formal incident closure criteria.**

**NIST Phase:** Recovery

**Reference:** NIST 800-61r3 §3.5 — Recovery

**Controls:** CIS 5.1 (Establish and Maintain an Inventory of Accounts), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AC-2 (Account Management)

**Compensating:** Run a full-site PHP file audit before re-enabling: `find /var/www/html -name '*.php' -newer /var/www/html/wp-includes/version.php -not -path '*/product-slider-pro/*' -not -path '*/real-testimonials-pro/*' -not -path '*/smart-post-show-pro/*' | xargs ls -la` to surface webshells dropped outside the known plugin directories. Verify the wp_users table contains only known accounts: `wp user list --role=administrator --path=/var/www/html`. For 30-day post-recovery monitoring without a SIEM, configure a cron job to run `wp user list --role=administrator` daily and diff against the known-good account list, alerting on any new entries. Monitor Nginx/Apache access logs for authentication attempts against wp-login.php from previously unseen IPs.

**Evidence:** Before re-enabling public access, verify that all volatile evidence from Steps 1–3 (network connection captures, plugin directory archives, database dump, access log snapshots) has been transferred to offline, write-protected storage. Confirm that the wp-config.php credential snapshot is secured separately — credentials exfiltrated by the backdoor before containment may be used in delayed attacks against the database host or wp-admin interface during the recovery window, making the 30-day monitoring period forensically significant.

**Step 5: Post-Incident — this attack exposed a gap in third-party plugin update verification: sites had no mechanism to validate the integrity of updates delivered through an authenticated vendor channel. Implement file integrity monitoring (D3-SFA — System File Analysis) for all WordPress plugin directories. Evaluate deployment of a WordPress security plugin or server-side WAF rule set that alerts on new PHP file creation in plugin directories. Review vendor update trust policies: consider staging updates in a non-production environment before deploying to production for all paid plugins. Reference NIST CM controls for configuration change management and CIS 7.1 (Establish and Maintain a Vulnerability Management Process) for vendor update risk assessment procedures. Document lessons learned per NIST IR-4 (Incident Handling).**

**NIST Phase:** Post Incident

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

**Controls:** NIST IR-4 (Incident Handling), NIST AU-2 (Event Logging), 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:** Implement file integrity monitoring for WordPress plugin directories using the free AIDE tool (`aide --init` on the clean post-recovery state, then `aide --check` via daily cron) or inotifywait: `inotifywait -m -r -e create,modify /var/www/html/wp-content/plugins/ --format '%T %w%f' --timefmt '%Y-%m-%d %H:%M:%S' >> /var/log/plugin_changes.log 2>&1 &`. For future plugin updates from account.shapedplugin.com or any paid EDD-delivered plugin, download the zip to a staging server, run `sha256sum` against the vendor-published hash before deployment, and maintain a hash log per plugin version. Author a Sigma rule triggering on new PHP file creation in wp-content directories and pipe inotify output through a local syslog forwarder.

**Evidence:** Compile the complete incident timeline from evidence collected in Steps 1–4: first update timestamp from wp_options `auto_updater.lock` or plugin update logs, first observed malicious outbound connection, first unauthorized admin account creation event, and credential rotation timestamp. This timeline constitutes the post-incident record required for lessons learned documentation under NIST 800-61r3 §4 and supports any regulatory breach notification assessment if wp-config.php database credentials or customer PII were in scope for the exfiltration capability described in the advisory.

## Detection Guidance

Primary detection focus is file integrity and anomalous outbound traffic. Check plugin directories for PHP files modified or created during the update window for the three affected plugins. Look for unexpected outbound HTTP/S POST requests from the web server process to external IPs, particularly those carrying wp-config.php content or credential data (NIST AU-6, AU-12). In WordPress access logs, flag requests to plugin PHP files that are not standard plugin endpoints, especially those accepting POST parameters associated with credential submission or file writes. Search authentication logs for new WordPress admin account creation, role escalation events, or successful admin logins from IPs not associated with known administrators, these may indicate T1098 (Account Manipulation) post-exploitation. For sites with EDR or file integrity monitoring, alert on new .php file creation under wp-content/ outside of a scheduled maintenance window. Monitor DNS and network egress for connections to newly registered or low-reputation domains originating from web server processes. If a WAF is in place, review logs for anomalous plugin-path requests returning 200 responses with large response bodies, which may indicate web shell interaction (T1505.003). No confirmed IOC hashes or C2 infrastructure are available in source material at this time, behavioral detection is the primary viable method.

## Indicators of Compromise

| Type | Value | Context | Confidence |
|---|---|---|---|
| URL | `account.shapedplugin.com` | Official ShapedPlugin licensed update distribution endpoint — updates delivered through this channel during the compromise window may contain backdoor code | HIGH |

## Framework Mappings

### MITRE-ATTACK

- **T1552.001** — Credentials In Files
- **T1195.002** — Compromise Software Supply Chain
- **T1041** — Exfiltration Over C2 Channel
- **T1190** — Exploit Public-Facing Application
- **T1098** — Account Manipulation
- **T1070.004** — File Deletion
- **T1059.007** — JavaScript
- **T1505.003** — Web Shell
- **T1056.001** — Keylogging
- **T1059.004** — Unix Shell
- **T1556** — Modify Authentication Process

### NIST-800-53R5

- **CM-7** — Least Functionality
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-4** — System Monitoring
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SI-2** — Flaw Remediation
- **CM-2** — Baseline Configuration
- **SI-3** — Malicious Code Protection
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CM-3** — Configuration Change Control
- **AT-2** — Literacy Training and Awareness

### OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures
- **A04:2021** — Insecure Design
- **A07:2021** — Identification and Authentication Failures

### CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **5.2** — Use Unique Passwords
- **6.3** — Require MFA for Externally-Exposed Applications
- **14.2** — Train Workforce Members to Recognize Social Engineering Attacks

### 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
- **CC9.2** — Manages risks associated with vendors and business partners

### ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain

## MITRE ATT&CK Mapping

| Technique ID | Technique Name | Tactic |
|---|---|---|
| T1552.001 | Credentials In Files | Credential-Access |
| T1195.002 | Compromise Software Supply Chain | Initial-Access |
| T1041 | Exfiltration Over C2 Channel | Exfiltration |
| T1190 | Exploit Public-Facing Application | Initial-Access |
| T1098 | Account Manipulation | Persistence |
| T1070.004 | File Deletion | Defense-Evasion |
| T1059.007 | JavaScript | Execution |
| T1505.003 | Web Shell | Persistence |
| T1056.001 | Keylogging | Collection |
| T1059.004 | Unix Shell | Execution |

| Technique ID | Technique Name | Tactic |
|---|---|---|
| T1556 | Modify Authentication Process | Credential-Access |

## Sources

| Source | URL | Tier |
|---|---|---|
| **Security News** | https://thehackernews.com/2026/06/shapedplugin-wordpress-pro-plugin... | T3 |
| **CVE-2026-10777 Detail - NVD** | https://nvd.nist.gov/vuln/detail/CVE-2026-10777 | T1 |
| **CVE-2026-10735 \| Tenable®** | https://www.tenable.com/cve/CVE-2026-10735 | T3 |
| **CVE-2026-23377 - Red Hat Customer Portal** | https://access.redhat.com/security/cve/cve-2026-23377 | T3 |
| **CVE-2026-34977: Aperi'Solve Steganalysis RCE Vulnerability** | https://www.sentinelone.com/vulnerability-database/cve-2026-34977/ | T3 |
| **NVD** | https://nvd.nist.gov/vuln/detail/CVE-2026-49777, CVE-2026-10735 | T1 |



Generated 2026-06-22 19:01 UTC by TJS Security Command Center