



BGD e-GOV CIRT

TLP: CLEAR



CYBER THREAT ADVISORY

**Unpatched PHP Remote Code
Execution Vulnerability in Session
Upload Progress
CVE-2026-115560**

**TLP:** CLEAR**Distribution:** Public**Advisory on:** Unpatched PHP Remote Code Execution Vulnerability in Session Upload Progress — CVE-2026-115560**Severity:** High**Vulnerability Type:** Remote Code Execution (RCE)**CVE IDs:** CVE-2026-115560**Affected Technology:** PHP session handling and session upload progress**Patch Status:** No fix announced at the time of this advisory**Date:** 27 October 2026

Executive Summary

BGD e-GOV CIRT alerts organizations to a newly disclosed PHP remote code execution vulnerability, tracked as **CVE-2026-115560**. The issue concerns an interaction between PHP session serialization and the session upload progress feature that may create an RCE condition under a specific and unusual application configuration.

The published technical guidance indicates that the vulnerable condition requires all of the following:

- The application configures `session.serialize_handler` as `php_binary` or `php_serialize`.
- Application code subsequently switches the handler to `php` at runtime using `ini_set()`.
- The handler change occurs before `session_start()` is called.
- Session upload progress remains enabled.

PHP's published guidance describes this as an unlikely exploitation path and recommends disabling session upload progress until a regular bug-fix release becomes available.

At the time of publication, a complete technical advisory, a confirmed affected-version range, and a final severity score had not been published. No exploitation of CVE-2026-115560 had been reported in the sources reviewed for this advisory. Organizations should nevertheless apply the recommended mitigation, verify their application configuration, and monitor for further security updates.

Immediate action: Set `session.upload_progress.enabled=0` through an administrator-controlled PHP configuration, restart all relevant PHP workers, and test applications that rely on upload-progress functionality..

Vulnerability Overview

CVE-2026-115560 involves PHP's session handling behavior when an application changes the session serialization handler at runtime while session upload progress is enabled.

PHP supports multiple session serialization handlers. These handlers determine how session data is represented and processed. Session upload progress allows an application to track the progress of an uploaded file, typically to provide progress information to a user.

According to the published guidance, a specific combination of session-handler configuration and runtime behavior can create an unlikely RCE vector.

The exact low-level exploitation mechanism has not been fully documented in the public informational notice. Accordingly, this advisory does not assume a particular exploit payload, memory-corruption primitive, or post-exploitation outcome beyond the reported potential for remote code execution.

Key characteristics

Attribute	Assessment
Vulnerability identifier	CVE-2026-115560
Affected component	PHP session handling / session upload progress
Potential impact	Remote code execution under the required configuration
Authentication requirement	Not conclusively established in the available public technical details
Affected PHP versions	Not yet confirmed
CVSS score	Not yet published in the sources reviewed
Patch availability	No fix announced at the time of publication
Recommended mitigation	Disable session upload progress and restart relevant PHP workers
Reported exploitation	No exploitation reported for this CVE in the sources reviewed

Technical Details

Required configuration conditions

The published guidance identifies a particular sequence of application behavior that may expose the vulnerable path.

- The application initially configures `session.serialize_handler` to `php_binary` or `php_serialize`.

- Application code changes the handler to php using `ini_set()`.
- The handler change occurs before `session_start()`.
- PHP session upload progress remains enabled.

The combination of these conditions not simply the presence of PHP or an enabled upload-progress feature, is the reported risk scenario.

Applications that never change the session serialization handler at runtime are not exposed through this particular configuration path, according to the published guidance. However, administrators should validate the actual configuration and application code rather than assuming that a default setting alone establishes exposure.

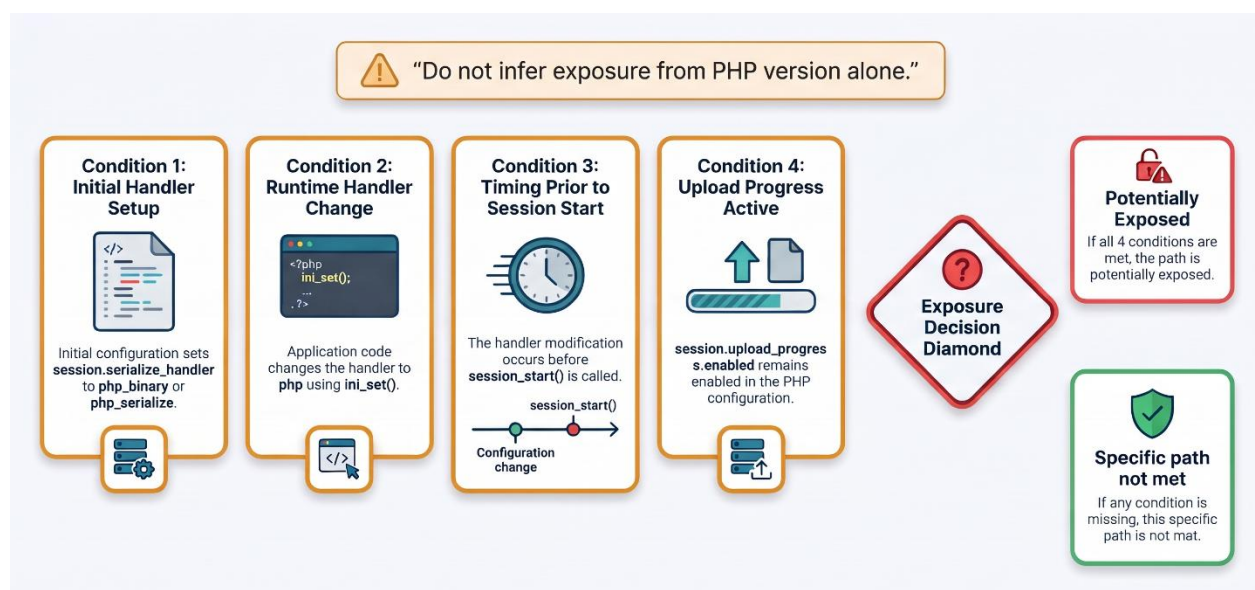


Fig: The Four Conditions Required for the Reported Vulnerability Path

Potential attack path

A conceptual attack path is:

External Request → Vulnerable Application Session Flow → Runtime Session-Handler Switch → Session Upload Progress Processing → Potential RCE

This is a high-level representation of the reported risk, not a confirmed exploit sequence or proof that every PHP deployment can be exploited remotely.

Whether an attacker can reach the relevant code path depends on application behavior, session configuration, request handling, and the deployment environment.



Fig: CVE-2026-115560: Vulnerability Overview and Potential Attack Path

Relationship to PHP2Shell

CVE-2026-115560 is a separate issue from the two vulnerabilities addressed by the PHP2Shell emergency updates:

- CVE-2026-109618 — remote code execution associated with out-of-memory handling.
- CVE-2026-109619 — remote code execution associated with stale request-related global state.
- CVE-2026-115560 — an additional RCE vector involving session upload progress and a specific runtime session-handler configuration.

The mitigation of disabling session upload progress was already included in guidance for CVE-2026-109619. This setting also mitigates the newly disclosed CVE-2026-115560 attack path.

Important: Installing the PHP2Shell fixed releases does not, by itself, establish that CVE-2026-115560 is fixed. The available guidance states that a separate fix is expected in a subsequent regular bug-fix release.

Affected Environments

Potentially relevant environments include PHP applications that use session handling and session upload progress, such as:

- Government and public-sector web applications.
- Public-facing portals and online service platforms.
- File-upload and document-submission systems.
- Applications that display upload-progress indicators.
- PHP applications using custom session-management logic.
- Applications that modify `session.serialize_handler` dynamically.
- PHP deployments using PHP-FPM, CGI, or web-server-integrated PHP runtimes, where the required application-level conditions are present.

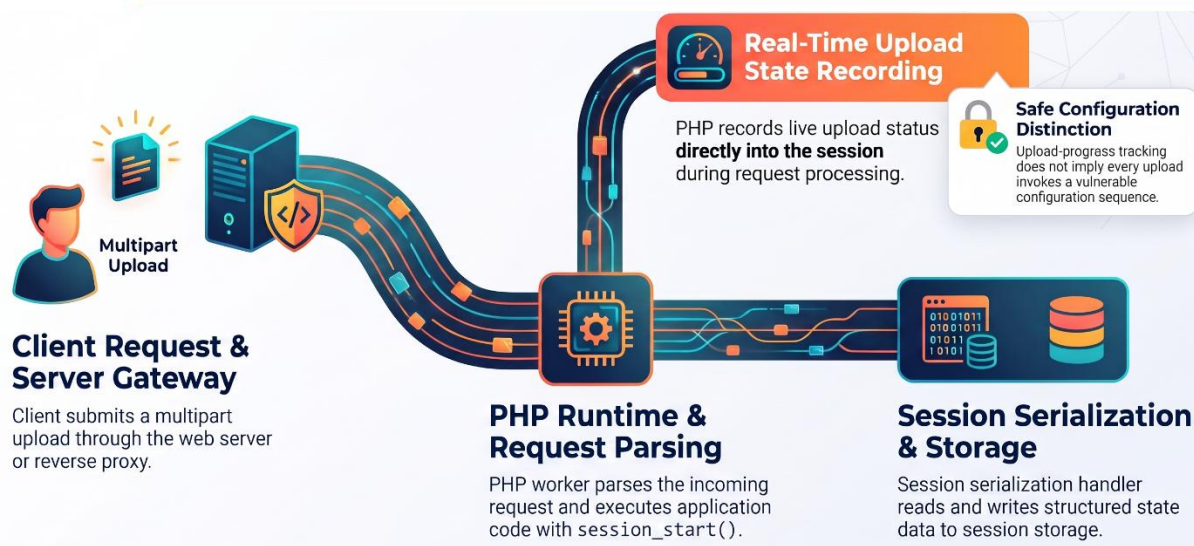


Fig: PHP Session Processing and Upload-Progress Architecture

Scope limitation: A PHP version banner, an Internet-facing web server, or an enabled upload-progress feature alone does not confirm exposure. The application configuration and runtime behavior must be examined. A definitive affected-version range has not been established in the public guidance reviewed.

Immediate Mitigation

Disable session upload progress

Set the following directive in an administrator-controlled configuration that takes effect before request parsing:

```
session.upload_progress.enabled=0
```

Suitable configuration locations depend on the deployment and may include:

- The applicable `php.ini`.
- An administrator-controlled PHP-FPM pool configuration.
- The relevant web-server PHP configuration.

After applying the change:

- Verify that the setting is effective for the actual production runtime.
- Restart all relevant PHP workers or services so that the change takes effect.
- Confirm that the setting cannot be overridden by application-level configuration.
- Test file-upload workflows and associated application functionality.
- Monitor PHP logs for configuration errors or unexpected application behavior.

Do not rely solely on a runtime `ini_set()` call or a per-directory `.user.ini` file for this mitigation. The setting must be applied early enough and consistently across the relevant workers.

Review session-handler changes

Search application code and configuration for:

```
ini_set('session.serialize_handler', ...);
```

Also review references to:

```
session.serialize_handler
session.upload_progress.enabled
```

Determine whether the application changes the session serialization handler at runtime, which handler values it uses, and whether the change occurs before `session_start()`.

Where possible, remove unnecessary runtime handler changes and adopt a consistent session configuration.

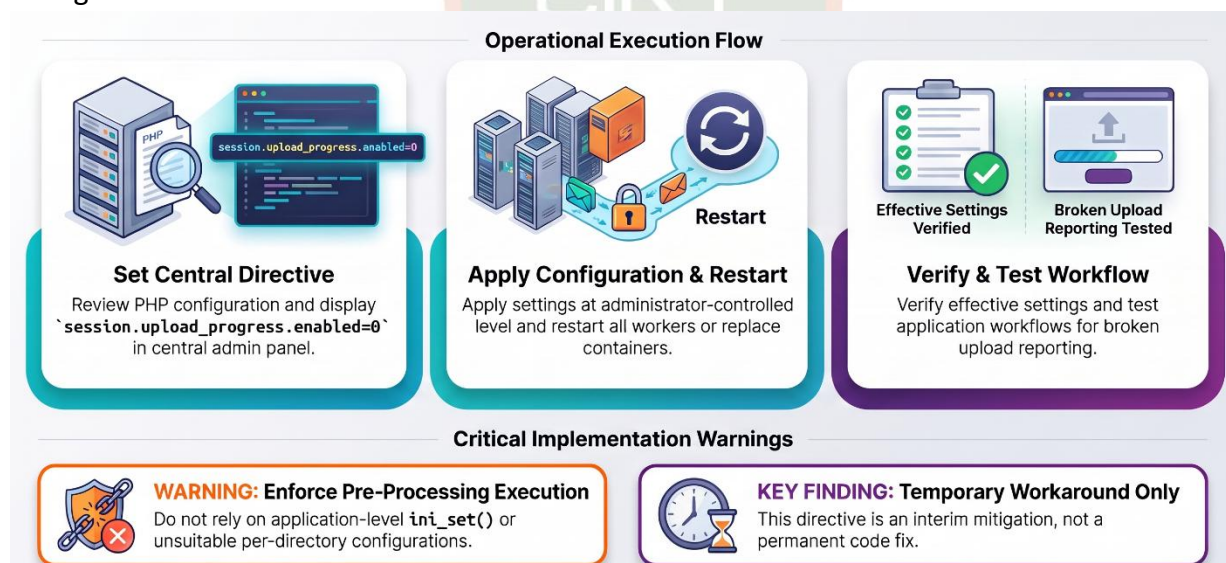


Fig: Immediate Mitigation - Disable Session Upload Progress

Review upload functionality

Disabling session upload progress may affect applications that rely on it to display file-upload progress. It should not normally prevent the underlying file upload from working, but application-specific behavior must be tested.

Validate:

- File-upload workflows.
- Document-submission forms.
- Session initialization and persistence.

- Upload-progress indicators.
- Background upload processing.
- Error handling and application logs.

Do not re-enable the feature solely to restore a progress indicator without first assessing the application's exposure and the available security updates.

Detection and Threat Hunting

The published information does not provide a reliable network signature or a confirmed exploit payload for CVE-2026-115560. Detection should therefore focus on configuration review, application behavior, and evidence of possible post-exploitation.

Configuration and code review

Identify applications that:

- Set `session.serialize_handler` to `php_binary` or `php_serialize`.
- Change the serialization handler to `php` at runtime.
- Perform the handler change before `session_start()`.
- Leave `session.upload_progress.enabled` enabled.

Prioritize applications where all the required conditions coexist.

Host and application monitoring

Investigate unusual activity involving PHP worker processes, including:

- Unexpected child processes or command interpreters.
- New or modified files in web roots and temporary directories.
- Unexpected outbound network connections.
- Unauthorized access to application configuration and secrets.
- Unusual authentication events or account changes.
- Unexpected application errors or PHP worker crashes.
- Unexplained changes to session-handling code or PHP configuration.

These are general indicators of possible compromise, not CVE-specific indicators. The absence of such indicators does not establish that an application is unaffected.

SIEM correlation recommendations

Correlate:

1. Requests reaching file-upload or session-related application endpoints.
2. Unusual PHP worker behavior or process creation.
3. Unexpected outbound connections.
4. Unauthorized file or configuration changes.

5. Authentication anomalies or access to sensitive application resources.

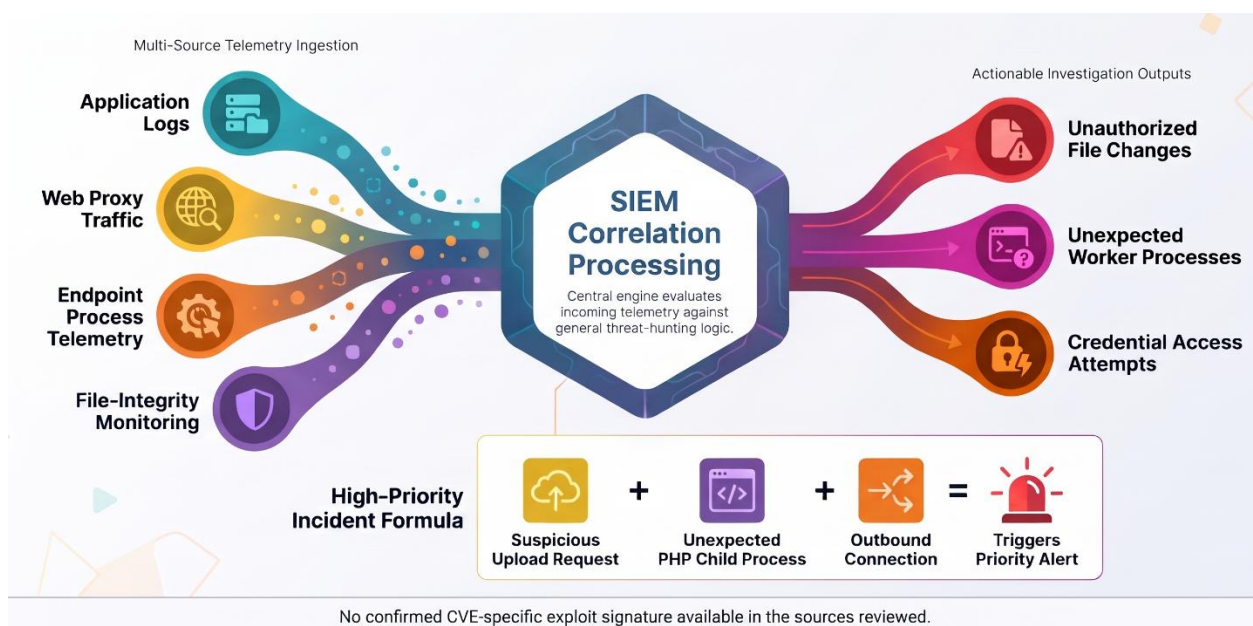


Fig: Detection and Threat-Hunting Architecture

Escalate findings where multiple independent indicators occur within the same application or server context.

Incident Response

If exploitation is suspected:

Contain: Restrict public access or isolate the affected application where operationally feasible.

Preserve evidence: Collect web-server, PHP, application, authentication, process, and network logs.

Validate exposure: Review the session-handler configuration and determine whether the required application behavior exists.

Investigate: Examine process execution, file changes, outbound connections, and access to credentials or sensitive data.

Protect credentials: Rotate potentially exposed application secrets, database credentials, API keys, and other affected credentials.

Remediate: Apply the recommended configuration mitigation, restart PHP workers, and remove unauthorized artifacts.

Patch: Install the applicable security fix when it is published and verified for the deployed runtime.

Recover and monitor: Restore trusted application components where needed and monitor for renewed suspicious activity.

Applying the mitigation or a future security update does not automatically remove persistence or reverse a compromise that may have occurred earlier.



Fig: Incident Response Workflow for Suspected PHP Compromise

Recommendations

CTI Unit of BGD e-GOV CIRT recommends that government agencies, critical-infrastructure operators, financial institutions, telecommunications providers, educational institutions, and private-sector organizations:

- Identify Internet-facing PHP applications and their session configurations.
- Apply the recommended session upload progress mitigation where operationally appropriate.
- Review application code for runtime changes to session.serialize_handler.
- Verify that mitigation settings apply to every production worker and backend instance.
- Test file-upload and session-related functionality after changes.
- Review security logs for suspicious activity and investigate credible indicators of compromise.
- Maintain an inventory of PHP runtimes, application owners, and remediation status.
- Coordinate with hosting and application providers where configuration is managed externally.
- Track official security updates and apply the applicable fix once available.

Organizations should prioritize applications that exhibit the complete reported configuration sequence rather than treating every PHP installation as equally exposed to this specific vulnerability.

At the time of this advisory:

- A public informational notice identifies CVE-2026-115560 and recommends disabling session upload progress.
- The reported risk depends on an unusual combination of session-handler configuration and application behavior.
- A confirmed affected-version range and final severity score have not been established in the sources reviewed.
- A complete technical advisory and security fix are expected in a subsequent regular bug-fix release.
- No exploitation of this specific CVE was reported in the sources reviewed.

Organizations should retain the mitigation until the relevant security update and accompanying guidance are available. The status should be reassessed when the full technical advisory is published.



Fig: Organizational Exposure and Remediation Workflow

CVE-2026-115560 introduces a potential remote code execution risk in PHP applications that use a specific runtime session-handler configuration while session upload progress remains enabled.

Although the published guidance describes the configuration as unusual, organizations should not rely on that assessment without checking their own applications. Disabling session upload progress, reviewing runtime session-handler changes, and monitoring for suspicious activity are appropriate immediate measures while awaiting a complete security fix.

BGD e-GOV CIRT urges organizations operating PHP-based services to review their session configurations, apply the recommended mitigation, validate application behavior, and monitor for the forthcoming security update.

References

- <https://www.php.net>
- <https://thehackernews.com/2026/10/urgent-php-warns-of-another-unpatched.html>
- <https://www.bleepingcomputer.com/news/security/php-shares-mitigation-for-new-and-unpatched-rce-vulnerability>
- <https://branddefense.io/knowledge-hub/vulnerabilities/cve-2026-115560>



BGD e-GOV CIRT