From "Language Switcher" to Full Server Control

How I exploited a simple web feature to gain RCE

02

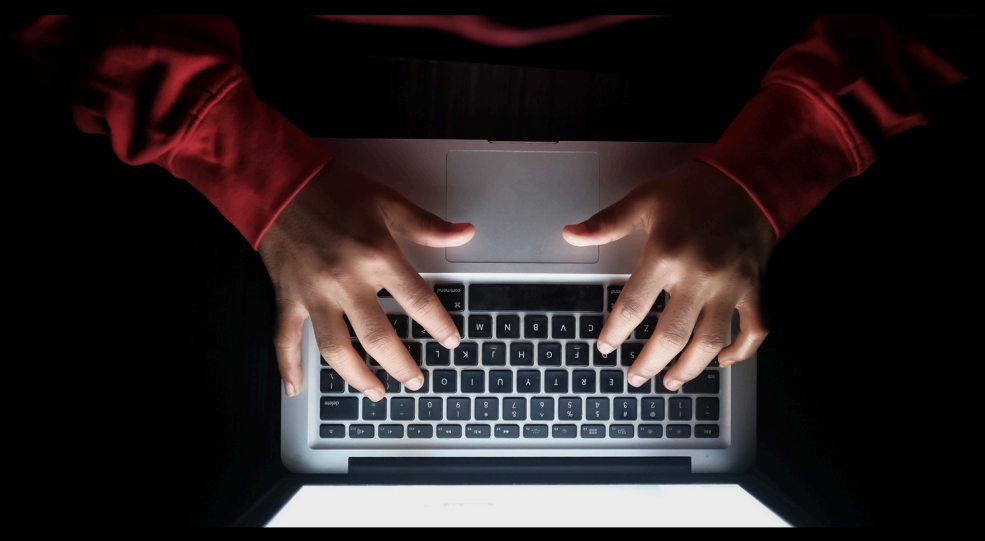
Hacker Mindset

- The app used a lang parameter: index.php?lang=EN.
- Behind the scenes: include("languages/" . $lang . ".php");
- The Flaw: The app trusted user input to build a file path without proper sanitization.

Why did it happen?

03

Cracking the Filters



1.The Null Byte (%00)

- The Trap: Server forces .php at the end of your input.
- The Hack: Add %00 to your path.
- The Result: It acts as a "Stop Sign." The server ignores the .php and opens your file.

2. The Nested Bypass (....//)

- The Trap: Security deletes every ../ it sees.
- The Hack: Use ....// instead.
- The Result: The filter deletes the middle ../, and the remaining characters collapse into a new ../ that slips through.
-

💡 Pro-Tip: "Find-and-replace" isn't real security. Use Allow-listing instead!

How to prevent this?

The "Defense"

04

- Never trust user input in file paths.
- Use Allow-listing: Only allow specific filenames (EN, ES, FR).
- Use basename(): Strip everything but the filename.
- Disable allow_url_include in PHP settings.

Thank YOU

Did you find these bypasses useful?

Repost to help a fellow researcher.

- @Mohak Agarwal