🛡️ PHP 后端网页项目安全访问机制维护档案

文档创建人： Cui Shidong
创建日期： 2025-04-24
适用版本： 所有以 PHP 为后端的网页型数据库项目

📌 背景说明

miRTarBase 的多个 PHP 数据接口（如 `detail.php`, `search.php`, `showDatasetDetail.php` 等）遭遇了大量爬虫请求，主要特征包括：

- 高频 GET 请求；
- 动态参数构造（如 `miRNA`, `gene`, `dataset_id` 等）；
- 使用代理池、模拟 UA 等方式绕过基础防护。

为防止数据库过载和数据滥用，我们引入了一套轻量的防爬虫机制，确保只有“**真实网页用户**”能够访问核心数据内容。

✅ 实施内容

统一在 `configs/security.php` 中添加以下逻辑：

- 判断是否为主流浏览器 User-Agent；
- 判断是否执行 JavaScript 并设置 cookie（`js_check=1`）；
- 可选 Referer 检查；
- 未通过验证的请求：
  - 非浏览器：立即返回 403 Forbidden；
  - 浏览器首次访问：返回 412 Precondition Failed，并附带 JS reload。

⚙️ 技术细节

- **创建统一验证模块路径：** `/configs/security.php`
- **接入方式：** 每个需要保护的 PHP 页面顶部添加(以实际路径为准)：
  `include_once(../configs/security.php');`
- **验证失败返回状态码：** `403` 或 `412`
- **自定义响应头：** `X-Protect: JS-Check`
- **Cookie 名称：** `js_check`

🧩 实际使用的 security.php 代码

```php
<?php
if (session_status() === PHP_SESSION_NONE) {
    session_start();
}

// 1. Browser UA check
$user_agent = $_SERVER['HTTP_USER_AGENT'] ?? '';
$allowed_browsers = ['Chrome', 'Firefox', 'Safari', 'Edge', 'Opera', 'MSIE', 'Trident',
'Mozilla'];
$is_browser = false;
foreach ($allowed_browsers as $b) {
    if (stripos($user_agent, $b) !== false) {
        $is_browser = true;
        break;
    }
}

// 2. Referer check (optional, can be relaxed if needed)
$referer_ok = true;
if (isset($_SERVER['HTTP_REFERER'])) {
    $referer = $_SERVER['HTTP_REFERER'];
    $allowed_sources = ['awi.cuhk.edu.cn', '10.26.4.101'];
    $referer_ok = false;
    foreach ($allowed_sources as $src) {
        if (strpos($referer, $src) !== false) {
            $referer_ok = true;
            break;
        }
    }
}

// 3. Check if JS cookie is set
$js_cookie_set = isset($_COOKIE['js_check']) && $_COOKIE['js_check'] === '1';

// === Non-browser access, immediate 403 ===
if (!$is_browser) {
    http_response_code(403);
    echo "Access denied. Please use a real web browser.";
    exit;
}

// === Browser but cookie not set (first visit) → Set cookie, return 412 + JS reload ===
if (!$js_cookie_set) {
    header("X-Protect: JS-Check");
    http_response_code(412);  // Precondition Failed
    echo '<script>document.cookie="js_check=1; path=/"; window.location.reload();
</script>';
    exit;
}

// === Invalid referer, also denied ===
if (!$referer_ok) {
```

```
    http_response_code(403);
    echo "Access denied. Invalid referer.";
    exit;
}
?>
```

📈 日志状态码说明

- `200` – 真实浏览器访问已通过验证
- `403` – 非浏览器或伪装失败，立即拦截
- `412` – 浏览器首次访问，未设置 cookie，返回 reload 脚本

✅ 日志示例

```
40.77.167.20 - - "GET /php/detail.php?... HTTP/1.1" 403 45
40.77.167.20 - - "GET /php/detail.php?... HTTP/1.1" 412 80
```

说明：

- `412` 表示浏览器首次访问但尚未设置 cookie，返回设置 cookie 的 JS 脚本；
- `403` 表示请求来源不是浏览器（如 curl 或伪装爬虫）或 UA 检查失败；
- 响应体长度极小（如 80、45 字节），无实际数据或页面内容返回，确保数据安全。

可通过以下方式模拟验证行为：

```
# 模拟爬虫访问，期望返回 403
curl -A "curl/7.85.0" -i "https://awi.cuhk.edu.cn/~miRTarBase/..."

# 实时查看访问日志（推荐安装 ccze 彩色高亮）
sudo tail -f /var/log/httpd/ssl_access_log | ccze

# 或使用纯文本方式查看
sudo tail -f /var/log/httpd/ssl_access_log
```

🔄 后续建议（可扩展）

- 记录保护行为日志（IP、UA、路径）
- 引入访问频率限制（如 session 或 Redis）
- 自动封禁过于频繁命中 412 的 IP
- 可扩展为验证码验证机制