Apache를 통한 보안설정

- [SECURITY] http header의 cookie
- [Security] Content-Security-Policy
- [SECURITY] Header 보안설정하기

[SECURITY] http header의 cookie

쿠키 이름	역할	**Secure** 플래그	**HttpOnly** 플래그
PHPSESSID	PHP 기본 세션 쿠키	적용 안됨	적용 안됨
XSRF-TOKEN	CSRF 보호를 위한 토큰	적용됨	적용 안됨
Laravel 세션 쿠키:	Laravel 세션 쿠키	적용됨	적용됨

XSRF-ROKEN과 Laravel 세션 쿠키는 프론트에서 처리 가능

> “
>
> **PHPSESSID**의 **Secure** 플래그 추가 방법

- **php.ini** 설정 수정

  session.cookie_secure = On

- **PHP** 코드에서 설정

  session_start()를 호출하기 전에 아래와 같이 설정합니다:

```
ini_set('session.cookie_secure', 1);
ini_set('session.cookie_httponly', 1); // HttpOnly 추가
session_start();
```

- **.htaccess** 파일에서 설정
- .htaccess를 사용하여 PHP 설정을 수정할 수 있습니다:

```
php_value session.cookie_secure On
php_value session.cookie_httponly On
```

[Security] Content-Security-Policy

Content-Security-Policy header is not set. It restricts resources the page can load and prevents XSS attacks.

seo 진단시 위와 같은 진단내용이 잡힌다면 아래 내용들을 확인해주세요.

> “
> **Content-Security-Policy(CSP)란?**

CSP는 HTTP 응답 헤더 중 하나로, 브라우저가 페이지에 어떤 리소스를 로드할 수 있는지 제어하는 보안 정책을 정의합니다.
이를 통해 아래와 같은 보안 위협을 방지할 수 있습니다:

1. **XSS(**크로스 사이트 스크립팅**)**
   공격자가 악성 JavaScript를 주입하는 것을 방지합니다.
2. 데이터 주입 공격
   악성 리소스(URL, CSS, 폰트 등)의 로드를 제한합니다.
3. **Clickjacking**
   악성 iframe 삽입 등을 방지합니다.

> “
> **Content-Security-Policy**가 중요한 이유

CSP가 없으면 다음과 같은 문제가 발생할 수 있습니다:

- 외부에서 [illegible] 주입하여 사
- 비인가된 리소스가 로드되어 정보 유출 가능.
- HTTPS 환경에서도 보안이 취약한 HTTP 리소스가 로드될 가능성.

해결 방법: Content-Security-Policy 헤더 설정

Apache 설정을 통해 CSP 추가

Apache[Content-Security-Policy 헤 .htaccess 파일[httpd.conf , apache2.conf)에 다음 규칙을 추가합니다.

```
<IfModule mod_headers.c>
Header set Content-Security-Policy "default-src 'self'; script-src 'self' https://www.googletagmanager.com
https://d3js.org https://cdnjs.cloudflare.com https://cdn.jsdelivr.net 'unsafe-inline' 'unsafe-eval'; style-src 'self'
https://fonts.googleapis.com https://cdn.jsdelivr.net 'unsafe-inline'; font-src 'self' https://fonts.gstatic.com
https://cdn.jsdelivr.net data:; connect-src 'self' https://www.google-analytics.com; img-src 'self' data:
https://cdn.jsdelivr.net;"
</IfModule>
```

default-src 'self'

현재 사이트 외의 다른 출처에서 리소스(스크립트, 스타일, 이미지)를 불러오지 못하게 방지합니다.

self

현재 사이트 자체에서 제공되는 스크립트만 허용합니다.

unsafe-inline

인라인 스크립트를 허용합니다 (보안상 추천되지 않음).

unsafe-eval

eval() [setTimeout(string) 같은 동적 코드 실행을 허용합니다.

data:

데이터 URI를 통해 폰트, 이미지 등의 데이터를 허용합니다.

* 줄바꿈이 포함되면 500 server error가 발생할 수 있습니다.

2. 프론트에서 미들웨어 추가 (라라벨 기준)

1. app/Http/Middleware에 새로운 class를 생성합니다.

2. 아래와 같이 handle function을 추가합니다. $csp 변수 안에 내용은 직접 변경하셔야합니다.

```
public function handle(Request $request, Closure $next)
    {
        $response = $next($request);



        $csp = "default-src 'self'; " .
       "script-src 'self' https://www.googletagmanager.com https://d3js.org https://cdnjs.cloudflare.com
https://cdn.jsdelivr.net 'unsafe-inline' 'unsafe-eval'; " .
       "style-src 'self' https://fonts.googleapis.com https://cdn.jsdelivr.net 'unsafe-inline'; " .
       "font-src 'self' https://fonts.gstatic.com https://cdn.jsdelivr.net data:; " .
       "connect-src 'self' https://www.google-analytics.com; " .
       "img-src 'self' data: https://cdn.jsdelivr.net;";



        $response->headers->set('Content-Security-Policy', $csp);



        return $response;
    }
```

3. `app/Http/Kernel.php` 에 미들웨어를 추가합니다.

```
protected $middleware = [ \App\Http\Middleware\ContentSecurityPolicy::class, ];
```

권장사항

엄격한 정책으로 전환

- [`unsafe-inline` 을 사용할 수 있지만, 최종 배포시 제거하는 것이 좋습니다.
- `nonce` [`hash` 기반 CSP를 사용하여 특정 스크립트만 허용하도록 설정하는 것이 좋습니다.

Nonce를 사용 (권장)

아래 코드는 nonce 적용에 대한 예시코드입니다.

- apache 설정

```
Header set Content-Security-Policy "default-src 'self'; script-src 'self' https://www.googletagmanager.com
https://d3js.org https://cdnjs.cloudflare.com 'nonce-abc123'; style-src 'self' https://fonts.googleapis.com
https://cdn.jsdelivr.net 'unsafe-inline'; font-src 'self' https://fonts.gstatic.com; connect-src 'self'
https://api.example.com; img-src 'self' data:;"
```

- script 사용부분

```
<script nonce="abc123"> console.log('Inline script with nonce'); </script>
```

- 동적 Nonce 생성

```
<?php
// php + javascript
$nonce = base64_encode(random_bytes(16));
header("Content-Security-Policy: script-src 'self' 'nonce-$nonce'");
?>

<script nonce="<?php echo $nonce; ?>">
  console.log('This script uses a nonce');
</script>
```

```
<script>
  // only javascript
  // Generate a random nonce
  const array = new Uint8Array(16);
```

```
  window.crypto.getRandomValues(array);
  const nonce = btoa(String.fromCharCode.apply(null, array));

  // Set the Content-Security-Policy header
  const meta = document.createElement('meta');
  meta.setAttribute('http-equiv', 'Content-Security-Policy');
  meta.setAttribute('content', `script-src 'self' 'nonce-${nonce}'`);
  document.head.appendChild(meta);

  // Dynamically create a script element with the nonce
  const script = document.createElement('script');
  script.setAttribute('nonce', nonce);
  script.textContent = "console.log('This script uses a nonce')";
  document.body.appendChild(script);
</script>
```

[SECURITY] Header □□□□□

Header	OK	Notice	Warning	Critical	Recommendation
X-Frame-Options	0	0	77	0	X-Frame-Options header is not set. It prevents clickjacking attacks when set to 'deny' or 'sameorigin.'
X-Content-Type-Options	0	0	77	0	X-Content-Type-Options header is not set. It stops MIME type sniffing and mitigates content type attacks.
Referrer-Policy	0	0	77	0	Referrer-Policy header is not set. It controls referrer header sharing and enhances privacy and security.
Feature-Policy	0	0	77	0	Feature-Policy header is not set. It allows enabling/disabling browser APIs and features for security. Not important if Permissions-Policy is set.
Permissions-Policy	0	0	77	0	Permissions-Policy header is not set. It allows enabling/disabling browser APIs and features for security.
Server	0	0	77	0	Server header is set to known 'Apache.' It is better not to reveal used technologies.
Set-Cookie	70	65	70	0	Set-Cookie header for 'PHPSESSID' does not have 'SameSite' flag. Consider using 'SameSite=Strict' or 'SameSite=Lax.'
					Set-Cookie header for 'XSRF-TOKEN' does not have 'HttpOnly' flag. Attacker can steal the cookie using XSS. Consider using 'HttpOnly' when cookie is not used by JavaScript.

> “
> # X-Frame-Options

문제:

- X-Frame-Options 헤더가 설정되지 않음.

해결:

Apache 설정 파일에 다음을 추가:

```
Header always set X-Frame-Options "DENY"
```

- **DENY**: iframe 로딩을 완전히 차단.
- **SAMEORIGIN**: 같은 도메인에서만 iframe 로딩 허용.

> “
> # X-Content-Type-Options

문제:

- X-Content-Type-Options 헤더가 설정되지 않음.

해결:

Apache 설정 파일에 다음을 추가:

```
Header always set X-Content-Type-Options "nosniff"
```

- MIME 타입 스니핑 방지.

> # Referrer-Policy

문제:

- Referrer-Policy 헤더가 설정되지 않음.

해결:

Apache 설정 파일에 다음을 추가:

```
Header always set Referrer-Policy "no-referrer"
```

- **no-referrer**: 참조 헤더를 완전히 숨김.
- 다 `strict-origin`, `strict-origin-when-cross-origin`, `same-origin` 등.

> “
> # Feature-Policy / Permissions-Policy

문제:

- Feature-Policy 또는 Permissions-Policy 헤더가 설정되지 않음.

해결:

Apache 설정 파일에 다음을 추가:

```
Header always set Permissions-Policy "geolocation=(), camera=(), microphone=()"
```

- **Permissions-Policy**: 특정 브라우저 기능(API) 사용을 제한.
- 예: `geolocation`, `camera`, `microphone` 등.

Server

문제:

- Server 헤더가 Apache로 설정되어 서버 정보가 노출됨.

해결:

Apache 설정 파일에 다음을 추가:

```
ServerTokens Prod
ServerSignature Off
Header unset Server
```

- **ServerTokens Prod**: 최소한의 정보만 노출.
- **ServerSignature Off**: 오류 페이지에서 Apache 정보 숨김.
- **Header unset Server**: Server 헤더를 제거.

> “
Set-Cookie

문제:

- SameSite 및 HttpOnly 플래그가 없음.

해결:

Apache 설정 파일에 다음을 추가:

```
Header always edit Set-Cookie ^(.*)$ "$1; SameSite=Strict; HttpOnly; Secure"
```

- **SameSite=Strict**: 쿠키를 외부 요청에 전송하지 않음.
- **HttpOnly**: JavaScript에서 쿠키 접근을 차단.
- **Secure**: HTTPS 요청에서만 쿠키를 전송.

□□ □□□

```
# X-Frame-Options □□
Header always set X-Frame-Options "DENY"

# X-Content-Type-Options □□
Header always set X-Content-Type-Options "nosniff"

# Referrer-Policy □□
Header always set Referrer-Policy "no-referrer"

# Permissions-Policy □□
Header always set Permissions-Policy "geolocation=(), camera=(), microphone=()"

# Server □□ □□□
ServerTokens Prod
ServerSignature Off
Header unset Server

# Set-Cookie □□ □□
Header always edit Set-Cookie ^(.*)$ "$1; SameSite=Strict; HttpOnly; Secure"
```