Webサイト
脆弱性調査レポート

非破壊（パッシブ）調査による外部公開情報のセキュリティ点検

調査対象	https://example.co.jp
調査日	2026年7月17日
調査手法	非破壊（GETによる閲覧・存在確認のみ）
実施範囲	許可取得済みのWebサイト
取扱区分	社外秘 / Confidential

調査サマリー

本調査はサイトへの通常アクセス（ページ閲覧・レスポンスヘッダー取得・公開エンドポイントの存在確認）での試験を実施しており攻撃的アクセスを実施しておりません。サイトはWordPress（バージョン7.0.1）で構築されています。検出事項は下表のとおりで、特に「管理者ユーザー名の露出」と「サポート終了PHPの使用」への優先対応を推奨します。

深刻度	件数	主な内容
高〜中	3	管理者名露出／PHPサポート終了／内部URL漏洩
中	2	WordPressバージョン露出／セキュリティヘッダー未設定
軽微	2	xmlrpc有効／readme.html公開
良好	6	機密ファイル非公開／ディレクトリー覧無効／開発環境保護 等

■ 高〜中リスク（優先対応）

1. 管理者ユーザー名の露出＋ユーザー列挙が可能

高〜中

/wp-json/wp/v2/users および /wp-json/wp/v2/users/1 から、管理者のログイン名が「admin」であることを誰でも取得できます。また ?author=1 も /author/admin/ へリダイレクトし、同じ情報が漏れます。攻撃者はパスワードを破るだけでよい状態となり、ブルートフォース（総当たり）の成功率が大きく上がります。

→ 対策: ①ログイン名を「admin」以外へ変更、②REST APIのユーザー列挙を無効化、③ログイン試行回数制限（Limit Login Attempts等）を導入。

2. PHP 8.0 のサポート終了（EOL）＋バージョン露出

高〜中

全ページのHTTPヘッダーに X-Powered-By: PHP/8.0.30 が出力されています。PHP 8.0は2023年11月でセキュリティ更新が終了しており、以降に発見された脆弱性は修正されないまま残存している可能性があります。

→ 対策: PHP 8.2以降（推奨は8.3）へアップグレード。あわせて expose_php = Off でバージョンヘッダーを非表示に。

3. 内部・テスト環境URLの漏洩

中

REST APIの管理者プロフィール情報内に、以下の非公開URLが露出しています。

- http://test.www.example.co.jp/wordpress（http=平文。稼働していれば通信傍受のリスク）
- https://staging.example.net（開発環境）

※ 開発環境 staging はBasic認証で保護済み（401応答）で、この点は適切でした。ただしURL自体が本番サイトから漏れている状態は望ましくありません。

→ 対策: 管理者プロフィールの「サイト（URL）」欄から内部URLを削除する。

■ 中リスク

4. WordPressバージョンの露出

中

トップページに <meta name="generator" content="WordPress 7.0.1">
が出力され、また読み込みスクリプトのクエリ ?ver=7.0.1
からもバージョンが判明します。特定バージョンの既知脆弱性を狙われやすくなります。

→ 対策: generatorタグとバージョンクエリ文字列の除去（functions.php
または専用プラグイン）。本体の最新版追従も継続する。

5. 公開ページにセキュリティヘッダーが未設定

中

トップ等の公開ページで、以下の主要なセキュリティヘッダーがすべて欠落していました。

ヘッダー	状態	防げる主な攻撃
Strict-Transport-Security (HSTS)	未設定	通信の平文への降格
Content-Security-Policy	未設定	クロスサイトスクリプティング(XSS)
X-Frame-Options	未設定	クリックジャッキング
X-Content-Type-Options	未設定	MIMEスニッフィング
Referrer-Policy	未設定	リファラ経由の情報漏洩

※ /wp-login.php には一部（X-Frame-Options
等）が設定済みでしたが、これはWordPress既定によるもので、サイト全体には及んでいません。

→ 対策: Apacheの設定または .htaccess で、サイト全体に上記ヘッダーを付与する。

■ 軽微

6. xmlrpc.php が有効

軽微

/xmlrpc.php が稼働中（405応答）です。ブルートフォースの増幅（system.multicall）やpingback悪用の入口になり得ます。

→ 対策: 外部連携で使用していなければ無効化する。

7. /readme.html がアクセス可能

軽微

/readme.html が閲覧可能です。WordPress採用の裏付け情報となります。

→ 対策: ファイルを削除する。

■ 問題なし（適切だった点）

✓ .env / wp-config.php.bak / .git/config / バックアップ・SQLファイル等はすべて404（非公開）
✓ /wp-content/uploads/ ・/wp-includes/ のディレクトリー覧は403で無効化済み
✓ /wp-content/plugins/ ・/themes/ の一覧も空応答で内容は非表示
✓ 外部プラグインはトップページから検出されず、攻撃対象領域が小さい
✓ 開発環境（staging）はBasic認証で保護済み
✓ カスタムテーマ利用で既知テーマの脆弱性リスクが低い

■ 対応優先度のまとめ

順位	対応内容
1	「admin」ユーザー名の変更＋ユーザー列挙の無効化＋ログイン試行回数制限
2	PHP 8.2以降へのアップグレード
3	内部URLの削除／WordPressバージョンの隠蔽／セキュリティヘッダーの付与
4	xmlrpc.php・readme.html の整理

付録：対策の設定例

以下は各対策の代表的な実装例です。必ず検証環境で動作確認のうえ、バックアップを取得してから本番へ適用してください。サーバー構成（Apache/Nginx、権限、既存プラグイン）により調整が必要な場合があります。

① セキュリティヘッダーの付与（Apache / .htaccess）

```
<IfModule mod_headers.c>
  Header always set X-Content-Type-Options "nosniff"
  Header always set X-Frame-Options "SAMEORIGIN"
  Header always set Referrer-Policy "strict-origin-when-cross-origin"
  Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
  # CSP: adjust progressively to match your loaded resources
  Header always set Content-Security-Policy "default-src 'self'; img-src 'self' data: https:;"
  Header always unset X-Powered-By
</IfModule>
```

② PHPバージョンの隠蔽（php.ini）

```
expose_php = Off
```

③ WordPressバージョン露出の抑制（テーマの functions.php）

```
// Remove the generator meta tag
remove_action('wp_head', 'wp_generator');

// Strip ?ver= from styles and scripts
function id_remove_ver($src){
  return $src ? remove_query_arg('ver', $src) : $src;
}
add_filter('style_loader_src', 'id_remove_ver', 9999);
add_filter('script_loader_src', 'id_remove_ver', 9999);
```

④ REST APIのユーザー列挙を無効化（functions.php）

```
// Block /wp-json/wp/v2/users for unauthenticated access
add_filter('rest_endpoints', function($endpoints){
  if (isset($endpoints['/wp/v2/users'])) unset($endpoints['/wp/v2/users']);
  if (isset($endpoints['/wp/v2/users/(?P<id>[\\d]+)']))
    unset($endpoints['/wp/v2/users/(?P<id>[\\d]+)']);
  return $endpoints;
});
```

⑤ ?author= によるユーザー列挙のブロック（functions.php）

```
add_action('template_redirect', function(){
  if (!is_admin() && isset($_GET['author'])) {
    wp_redirect(home_url(), 301); exit;
  }
});
```

⑥ xmlrpc.php の無効化（.htaccess）

```
<Files xmlrpc.php>
  Require all denied
</Files>
```

⑦ readme.html への外部アクセス遮断（.htaccess）

```
<Files readme.html>
  Require all denied
</Files>
```

注：④⑤のユーザー列挙対策は専用のセキュリティプラグイン（例：SiteGuard WP Plugin
等）でも同等の制御が可能です。既存の運用に合わせて、コード実装かプラグインかを選択してください。