MBSD Cybersecurity Challenges
調査結果報告書

学校名：麻生情報ビジネス専門学校

チーム名：AIEL

2025年11月

1-1. 調査手法に関する説明

ツールを使った・作った、チームメンバの役割分担、調査手法など、脆弱性発見を網羅的・効率的に実施するために行った工夫があれば、アピールしてください。

目的：

mbsdEC サイトに対し、外部アクセスからアクセス可能な範囲における脆弱性を検出し、リスクを明確化する。

- 静的解析＋動的検証＋自動化を並行し、網羅性と再現性を両立
- システムの主要機能ごとにどのような脅威があり得るかを体系的に整理することで抜け漏れ抑止
- 最終的にすべて手動で取得・確認し、精度と再現性を確保する。

診断対象

ドメイン：http://172.16.97.128

環境：HTTP(port80)

対象 URL：アプリケーション全体

HTTP メソッド　GET/POST

検証端末：chrome/Firefox/ Burp embedded browser/Dev Tools

診断手法：ブラックボックステスト

- 診断ツール：OWASP ZAP、Burp Suite、独自クローラ

診断期間：2025/11/10~11/17

調査手法（手順）

1. マッピングを行う：
   (ア) ルーティング/機能一覧化
   (イ) 各機能ログインの可否を整理
   (ウ) 画面ごとに入力→検証→処理→出力のデータを可視化
2. 静的解析：
   (ア) DevTools で取得できる 最終出力 HTML／JavaScript／レスポンス内容 を対象に、脆弱につながる不備を機械的に抽出。
3. 動的検証:
   (ア) 実際に操作し、攻撃が成立するかを検証
4. 逆引き確認:

(ア) 画面上の挙動・レスポンスから内部処理の責務・前提条件を推測し、入力制御や認可ロジックの有無を間接的に確認。

5. 優先度付け:

(ア) 悪用難易度×影響度×露出度で評価し、危険度をスコアリングする

役割分担：

- 内野：レポート、資料収集
- 白神：レポート、全体設計
- 関：レポート、危険度スコアリング
- 東：レポート、レビュー

効率化ポイント

- 自動クローラで全ページの URL を収集し、見落としを防止
- 同一内容の検証を複数人で再現し、結果の信頼性を担保
- 画面キャプチャ・通信ログを自動保存し、再現性を高めた
- 攻撃検証（PoC）をテンプレート化し、短時間で再試行可能にした

危険度基準

各脆弱性に付与している危険度のレベルは、以下の基準に則って提示した。

重要度	基準	被害例
High（高）	被害者ユーザの関与がなくても、攻撃者が直接アプリケーションに対して攻撃可能である**能動的な脆弱性**。攻撃を受けると、大量の情報漏洩やデータ改ざんなど重大な被害を生じる可能性がある。	SQL インジェクション、認証回避、権限昇格、セッション固定化など
Medium（中）	攻撃成功には**被害者ユーザの関与（例：罠リンククリック、ページ閲覧）が必要な受動的な脆弱性**。または、能動的な脆弱性であっても大規模な漏洩・改ざんには至らないもの。	CSRF、クリックジャッキング、オープンリダイレクト、反射型 XSS など
Low（低）	攻撃成功の確率が低い、または成功しても**被害が軽微**と考えられる脆弱性。ただし、条件が整えば被害に発展する可能性はある。	エラーメッセージ列挙、オートコンプリート設定不備、ログアウト CSRF など
Info（情報）	それ単体では直接的な被害を引き起こさないが、他の脆弱性攻撃の足がかりとなる情報露出や設定上の問題。	サーバ時刻の露出、管理者 URL のコメント残存など

診断対象脆弱性の結果

診断対象脆弱性の結果は以下のとおりである。

High	Medium	Low	Info	合計
19	14	9	9	51

これらは 攻撃者が直接悪用可能で、情報漏洩やデータ改ざんに直結する重大な問題 であるため、すべて要修正として扱う必要がある。

被害発生の可能性が高く、早急な対応が求められる。以下のページに今回調査したリクエストの一覧を記す。

No	リクエスト名	URL
1	ログイン（画面）	GET http://{HOST}/login.php
2	ログイン（送信）	POST http://{HOST}/login.php → 302 /index.php
3	新規登録（画面）	GET http://{HOST}/register.php
4	新規登録（送信）	POST http://{HOST}/register.php → 302 /index.php
5	ログアウト	GET http://{HOST}/logout.php?to={PATH}
6	トップページ	GET http://{HOST}/index.php
7	ハートビート（AJAX）	POST http://{HOST}/hb.php（ヘッダ X-HeartBeat）
8	アカウント設定（画面）	GET http://{HOST}/account.php
9	アカウント情報変更 確認	POST http://{HOST}/accountInfoConfirm.php
10	アカウント情報変更 完了	POST http://{HOST}/accountInfoComplete.php
11	パスワード変更 確認	POST http://{HOST}/accountPasswordConfirm.php
12	パスワード変更 完了	POST http://{HOST}/accountPasswordComplete.php
13	ユーザー一覧（検索・ページング）GET	GET http://{HOST}/userlist.php?uname={q}&offset={n}
14	ユーザー一覧（検索・ページング）POST	POST http://{HOST}/userlist.php
15	ユーザ詳細（画面）	GET http://{HOST}/user.php?id={uid}
16	フォロー操作	POST http://{HOST}/user.php （type=1, to_id, from_id, csrf_token）
17	アンフォロー操作	POST http://{HOST}/user.php （type=0, to_id, from_id, csrf_token）
18	メール送信（画面）	GET http://{HOST}/sendmail.php
19	メール送信（送信）	POST http://{HOST}/sendmail.php
20	メール閲覧（画面）	GET http://{HOST}/readmail.php?id={mid}
21	メール削除	POST http://{HOST}/readmail.php（mode=delete, csrf_token）
22	商品一覧/検索（画面）	GET http://{HOST}/product.php?title={q}
23	商品詳細（画面）	GET http://{HOST}/product.php?mode=show&id={pid}
24	商品購入（画面）	GET http://{HOST}/product.php?mode=buy&id={pid}
25	商品購入（確定）	POST http://{HOST}/product.php?mode=buy&id={pid}（state=commit, price, csrf_token）
26	商品出品（新規/編集 画面）	GET `http://{HOST}/product.php?mode=make&id=new
27	商品出品（確認）	POST `http://{HOST}/product.php?mode=make&id=new
28	商品出品（確定）	POST `http://{HOST}/product.php?mode=make&id=new
29	自分の出品一覧（画面）	GET http://{HOST}/product.php?mode=own
30	自分の出品 CSV 出力	POST http://{HOST}/product.php?mode=own（output=csv）

31	CSV ダウンロード	GET http://{HOST}/csvdownload.php?filename={userId_YYYYMMDDhhmmss}.csv
32	クーポン（画面）	GET http://{HOST}/coupon.php
33	クーポン適用	POST http://{HOST}/coupon.php（code, csrf_token）
34	管理：クーポン発行（画面）	GET http://{HOST}/____generate_coupon.php
35	管理：クーポン発行（送信）	POST http://{HOST}/____generate_coupon.php（amount）
36	管理：管理メニュー（画面）	GET http://{HOST}/____specific_administrative_functions.php
37	管理：商品画像差替	POST http://{HOST}/____specific_administrative_functions.php（mode=product, id, img, csrf_token）
38	管理：ユーザ削除	POST http://{HOST}/____specific_administrative_functions.php（mode=deluser, id, csrf_token）
39	画像取得（本番画像）	GET http://{HOST}/img.php?id={pid}
40	画像取得（一時画像）	GET http://{HOST}/img.php?tmp={key}
41	管理：セキュリティコード計算	GET http://{HOST}/secode.php?card={nnnn-nnnn-nnnn-nnnn}

1-2. 発見した脆弱性詳細

[1] クリックジャッキング

- 対象

  ログイン（画面）
  http://172.16.97.128/login.php

- 危険度

  Medium

- 解説

- 想定される被害・影響

  意図せずにページを操作してしまう。

- 対策

  ポリシーを設定し、iframeへの埋め込みを禁止にする。

- 備考

  なし

[2] クロスサイトリクエストフォージェリー

・対象

ログイン（画面）
http://172.16.97.128/login.php
新規登録（画面）
http://172.16.97.128/register.php

・危険度

Medium

・解説

ログインフォーム等にCSRFトークンがなく、他サイトから自動POSTされると、被害者を攻撃者アカウントでログインさせるなどのCSRFが成立する。成立すると、以後の操作・閲覧が攻撃者アカウントで行われ、意図せぬ情報閲覧や操作誤認につながる。
例えば、以下のようなフォームを用意して、被害者がアクセスすることで発生する。

```
<form id=f method=POST action="http://172.16.97.128/login.php">
    <input name="loginid" value="attacker_id">
    <input name="password" value="attacker_pass">
</form>
<script>f.submit()</script>
```

実際に攻撃者のアカウントでログインされている。

・想定される被害・影響

被害者が攻撃者アカウントで操作

・対策

- ログインPOSTにCSRFトークンを導入
- SameSite=Strict/Lax のCookie、Origin/Referer検証の併用

・備考

なし

[3] ブルートフォースおよびクレデンシャルスタッフィング対策不備

・対象

ログイン（画面）
http://172.16.97.128/login.php
新規登録（画面）
http://172.16.97.128/register.php

・危険度

Medium

・解説

短時間に多数のログイン試行および新規アカウント登録を行っても、アカウントロック、遅延、CAPTCHAなどの抑制が行われない。

・想定される被害・影響

- アカウント乗っ取り増加
- 認証基盤・DB負荷増による可用性低下

・対策

アカウントロックの導入

・備考

なし

[4] タイミング攻撃による登録済みユーザ ID の推測

- 対象

  ログイン（画面）
  http://172.16.97.128/login.php

- 危険度

  info

- 解説

  存在しないユーザ ID でログインした場合、存在するユーザ ID でログイン試行した場合に比べて応答時間が遅くなっている。

  存在しない ID の場合 27ms

  存在する ID の場合 108ms

  結果的に「存在するユーザーID」で試行した場合の応答時間のほうが長いことが分かる

- 想定される被害・影響

  総当たりでログイン試行するよりも、初めに有効なログインIDを入手することでより効率的に攻撃を行うことができる。

- 対策

  応答時間を一定にする。

- 備考

  ネットワーク状況によって応答時間が大きく変わるため、環境差異が大きくなりやすい。

[5] ログイン ID 重複エラーメッセージによる列挙

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php

- 危険度

  info

- 解説

  既存IDを登録した際に「エラー：既に登録されているIDです。」という明確なメッセージが返り、有効な loginid を機械的に収集できる。

  このように表示される

- 想定される被害・影響

  有効IDリストの作成効率が向上

- 対策

  エラーメッセージの曖昧化

- 備考

  なし

[6] パスワードのオートコンプリート機能の不備

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php
  アカウント設定（画面）
  http://172.16.97.128/account.php

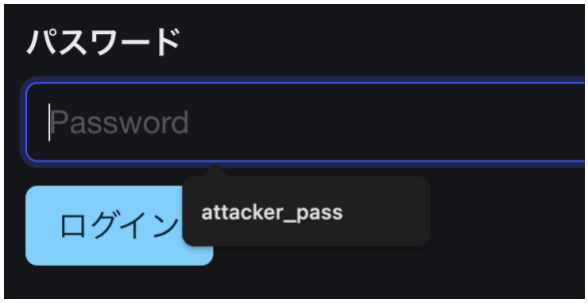
- 危険度

  low

- 解説

  パスワード入力欄に autocomplete="new-password" 等の指定がなく、ブラウザ保存や共有端末での覗き見のリスクが高まる。

  オートコンプリート機能によってパスワードが漏洩



- 想定される被害・影響

  パスワードの不要な保存・露出

- 対策

  新パス入力欄に autocomplete="new-password"、現パスにautocomplete="current-password" を指定する

- 備考

  なし

[7] 弱いパスワードの利用制限の不足

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php
  アカウント設定（画面）
  http://172.16.97.128/account.php

- 危険度

  Medium

- 解説

  パスワードの制限がなく、弱いパスワードでも登録可能になっている。

- 想定される被害・影響

  弱いパスワードを登録した場合、簡単に攻撃者にログインされてしまう。

- 対策

  パスワードチェックをサーバー側で行う。

- 備考

  なし

[8] ユーザ名の文字数制限の不備

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php

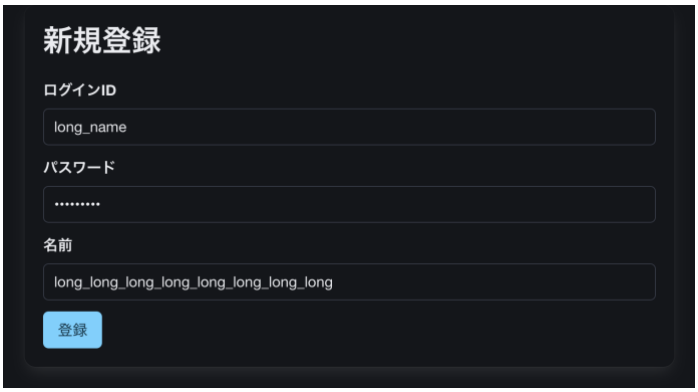
- 危険度

  info

- 解説

  ユーザ名は、**4文字以上20文字以内** でなければならないが、バリデーションの不備によって任意の長さのユーザ名で登録することができる。

  例えば、**Long_long_long_long_ …** のような20文字を超えた名前を入力する。



  登録後、アカウントページで名前を確認すると確かに登録されている。

- 想定される被害・影響。

  文字数制限がないことでXSSなどのスクリプト攻撃が行いやすくなる。

- 対策

  バリデーションを厳密に行う。

- 備考

  なし

[9] ユーザ名の重複登録

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php

- 危険度

  low

- 解説

  同一の名前でアカウントを登録することができる。ユーザ検索やメール送信などで、ユーザの誤認識を狙った攻撃や、なりすましといった攻撃につながる可能性がある。

  以下のように同一のユーザ名が複数ヒットする。

- 想定される被害・影響

  ユーザの誤認識や、なりすまし被害につながる危険性がある。

- 対策

  同一の名前では、登録できないようにする。

- 備考

  なし

[10] 登録直後のセッション ID 未ローテーション

- 対象

  新規登録（画面）
  http://172.16.97.128/register.php

- 危険度

  Medium

- 解説

  登録成功時にセッションIDが再生成されず、攻撃者が事前に配布したセッションIDでそのまま
  ログイン状態にされるセッション固定化のリスクがある。

  ログイン前

  ```
  Cookie PHPSESSID=p8hg8n2tv1p80teamqb36cnsik
  ```

  ログイン後

  ```
  Cookie PHPSESSID=p8hg8n2tv1p80teamqb36cnsik
  ```

- 想定される被害・影響

  登録直後のセッション乗っ取り

- 対策

  ログイン成功後にセッションを再生成する

- 備考

  なし

[11] 推測可能な CSRF トークン

・対象

新規登録（画面）
http://172.16.97.128/register.php

・危険度

low

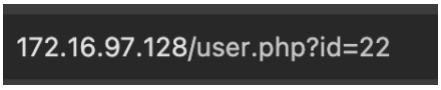
・解説

セキュリティコードが、あまりに単純なルールで作られており、部外者に偽造されてしまう可能性が非常に高い。 攻撃者はユーザ ID と時刻さえ分かれば、本人になりすまして偽のリクエストを送りつけることができる。

【再現方法】

1, ユーザ検索画面で自身の名前を検索して、選択

2, URL に表示された自分自身のユーザーID を確認



3, ログイン後にフォームがあるページでソースを表示して実際の CSRF トークンを取得

```
<input type="hidden" name="csrf_token" value="0b17de7eee10a064dfd5a59db8e488dd205afc4d">
```

4, 集めた情報をもとに、トークンを総当たりで生成するスクリプトを実行

```php
<?php
if ($argc < 3) {
    echo "使用法: php generate_tokens.php <ユーザーID> <ログイン時刻のUNIXタイムスタンプ>\n";
    exit(1);
}

$userId = $argv[1];
$loginTime = (int)$argv[2];
$timeWindow = 10; // ログイン時刻の誤差を考慮し、前後10秒間を探索

echo "ユーザーID '{$userId}'、時刻 '{$loginTime}' の前後{$timeWindow}秒でトークンを生成します。\n\n";

for ($i = -$timeWindow; $i <= $timeWindow; $i++) {
    $timestamp = $loginTime + $i;
    $token = sha1($timestamp . $userId);
    echo "時刻: {$timestamp} | トークン: {$token}\n";
}
```

5, ターミナル内でユーザーID とログイン時の UNIX タイムスタンプを実行

```
PS C:\Users\rui04> php "C:\Users\rui04\OneDrive\デスクトップ\generate_tokens.php" 23 1762914965
ユーザーID '23'、時刻 '1762914965' の前後10秒でトークンを生成します。
```

【結果】

```
時刻: 1762914955 | トークン: 31262802690c6b087dab343a61156a250913ad01
時刻: 1762914956 | トークン: fc101486bce9104ea13d7e160ac1ae07148f78d3
時刻: 1762914957 | トークン: f05b9eb29b4904163d444bd4e4c2704e6e8b0592
時刻: 1762914958 | トークン: d1697d414074e816d46105cfed603b57aa29492c
時刻: 1762914959 | トークン: 763514984a392a86c4b3434f80823c6329495003
時刻: 1762914960 | トークン: 5766b61a209c537692263d4cdb86e42c9b9d3c5a
時刻: 1762914961 | トークン: 03fd1511990702bc626c08dc4fff8717ea1a812e
時刻: 1762914962 | トークン: d6ec6ce9361f0f1cdc75a01f10c1648323bd120d
時刻: 1762914961 | トークン: 03fd1511990702bc626c08dc4fff8717ea1a812e
時刻: 1762914962 | トークン: d6ec6ce9361f0f1cdc75a01f10c1648323bd120d
時刻: 1762914963 | トークン: 4bf8c0b0746b1a6c387e503f991f0a85e8d66b37
時刻: 1762914964 | トークン: ef47d01d7c6617ac3f365abcced554a17f3a0bf1
```

・想定される被害・影響

強制的な不正操作: ユーザーが罠サイトを閲覧しただけで、意図せず「退会処理」「パスワード変更」「不正な投稿」などをさせられる恐れがある。

対策の無効化: 実装されている CSRF 対策が事実上機能していないため、実質的に「対策なし」と同等のリスク状態にある。

・対策

トークンの生成には、予測不可能な「暗号論的に安全な乱数」を使用する。

時刻や ID を使わず、専用の関数でランダムなバイト列を生成する。

・備考

フレームワーク（Laravel 等）を使用している場合は、フレームワーク標準の CSRF 保護機能を有効にするだけで、安全なトークン管理が自動的に行われる。

hb の脆弱性と組み合わせることで被害が拡大する。

[12] オープンリダイレクト

・対象

ログアウト
http://172.16.97.128/logout.php?to={PATH}

・危険度

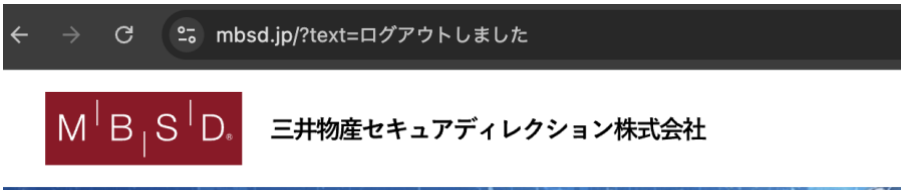
Medium

・解説

ログアウト時に to パラメータを受け取り、その値を Location にそのまま使用しているため、プロトコル相対URL（例: //mbsd.jp）を渡すと外部サイトへ遷移する。

遷移前のURL

遷移後の画面



・想定される被害・影響

- ログアウトと同時に任意の外部サイトへ誘導（フィッシング/マルウェア配布/トラッキング）
- SSO/多段遷移のある環境ではトークンや状態遷移の混乱

・対策

- 許可リスト方式で内部相対パスのみ許可（例: /login.php 等を固定）
- 正規表現で ^/(?!/) のように // を厳格に拒否し、.. も拒否
- もしくはサーバ側で固定の遷移先のみ使用し、クエリによる制御を廃止

・備考

その他上記に当てはまらないが報告すべきことがあれば記載

[13] GET メソッドによるログアウト CSRF

・対象

ログアウト
http://172.16.97.128/logout.php

・危険度

info

・解説

ログアウトが GET 要求で成立し、CSRFトークン検証が無いため、攻撃者サイトに埋め込まれた画像/iframe等の読み込みだけでユーザを強制的にログアウトさせられる。

・想定される被害・影響

- 強制ログアウトによる可用性低下・業務妨害
- 直後にオープンリダイレクトでフィッシングサイトへ誘導される複合攻撃

・対策

ログアウトは POST＋CSRFトークン必須

・備考

なし

[14] 管理者 URL の露出

- 対象

  トップページ
  http://172.16.97.128/index.php

- 危険度

  High

- 解説

  ソースコード内のコメントに、管理者用のパスが載っている。
  /____specific_administrative_functions.php にアクセスすることで、一般ユーザーでも管理者機能にアクセスすることができる。

- 想定される被害・影響

  管理者ページに不正にアクセスされる。

- 対策

  ソースコード内のコメントを削除

- 備考

  なし

[15] サーバ時刻の情報露えい

- 対象

  ハートビード
  http://172.16.97.128/hb.php

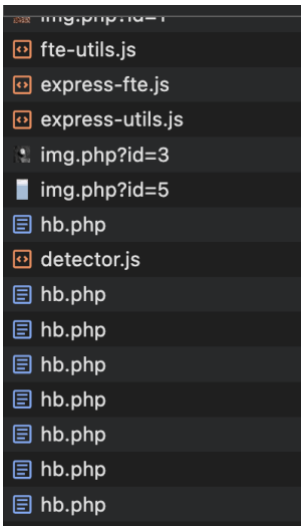
- 危険度

  Info

- 解説

  ログイン時に POST /hb.php へ定期送信し、応答に正確なサーバ時刻が含まれる。これは単体では重大ではないが、システム内に時刻依存のトークンや検証がある場合、推測を容易にする。

  定期送信の様子



  このようにサーバー時刻が返却される

```
HTTP/1.1 200 OK
Date: Wed, 12 Nov 2025 12:40:24 GMT
Server: Apache/2.4.58 (Ubuntu)
X-XSS-Protection: 0
```

- 想定される被害・影響

  - 時刻依存トークンの推測補助
  - 内部情報（サーバ時刻）の外部露出

- 対策

  - 不要ならサーバ時刻の返却をやめる
  - どうしても必要な場合でも精密な時刻や内部情報は返さない

- 備考

  本API単体では直接の侵入点になりにくいが、**[7]推測可能なCSRFトークン** などの他の欠陥と組み合わせると悪用余地が広がる

[16] パスワード変更時の再認証不足

- 対象

  アカウント設定（画面）
  http://172.16.97.128/account.php

- 危険度

  Medium

- 解説

  パスワード変更フォームに「現在のパスワード」入力欄が存在しない。そのため、セッションを保持した第三者が、本人の意思なく新パスワードに変更できる。

- 想定される被害・影響

  - アカウントの恒久的乗っ取り（パスワードを攻撃者任意に変更）
  - CSRF/クリックジャック等と組み合わせた強制変更

- 対策

  パスワード変更時は「現在のパスワード」を必須にする

- 備考

  なし

[17] パスワードの HTML 出力

・対象

アカウン設定（画面）
http://172.16.97.128/accountInfoConfirm.php

・危険度

High

・解説

前ページで入力したパスワードが、HTMLのvalueに平文のまま出力されている。

・想定される被害・影響

- アカウントの恒久的乗っ取り（パスワードを攻撃者任意に変更）
- CSRF/クリックジャック等と組み合わせた強制変更

HTML上に書かれたパスワードの漏洩。

・対策

パスワードをそのままHTML上に出力しない。

・備考

なし

[18] CSRF 検証の UI 依存

・対象

アカウント情報変更 確認
http://172.16.97.128/accountInfoConfirm.php

・危険度

low

・解説

CSRFトークンが不正・欠落でもサーバ側で即時エラーにならず画面が表示され、ボタンがdisabledになるだけである。確認ステップは直接POSTに対する防御が必要で、UI側のdisabledだけではCSRFを防げない。

・想定される被害・影響

直接POSTメソッドを送られるとCSRFの被害に遭う可能性がある。

・対策

サーバ側でもエラー扱いにする。

・備考

今回の場合、完了ページに遷移しても処理は実行されない。そのため現時点では影響はない。

[19] 権限値のクライアント委譲

・対象

アカウント情報変更 確認
http://172.16.97.128/accountInfoConfirm.php
アカウント情報変更 完了
http://172.16.97.128/accountInfoComplete.php

・危険度

High

・解説

確認画面のフォームに priv が hidden で含まれており、利用者が任意に書き換え可能。次の完了ステップ（/accountInfoComplete.php） がこの値をそのまま採用する実装であり、一般ユーザが priv=1 へ改ざんして権限昇格できる。

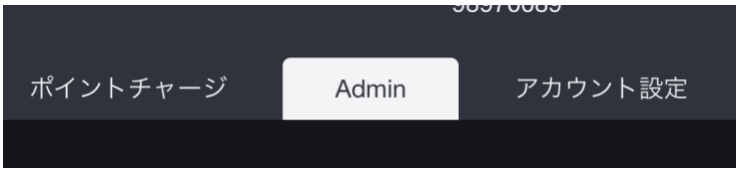
変更前のpriv

```
<input type="hidden" name="priv" value="0">
```

変更後のpriv

```
<input type="hidden" name="priv" value="1">
```

アカウント変更の成功後にAdminタブにアクセス可能になる。



・想定される被害・影響

一般ユーザによる管理者化

・対策

- 権限値はクライアントに渡さない/受け取らない
- 権限変更は管理者UI専用とし、サーバ側で操作者の権限を確認

・備考

なし

[20] 他端末セッション未失効

・対象

アカウント情報変更 完了
http://172.16.97.128/accountInfoComplete.php

・危険度

Medium

・解説

パスワード変更成功後も他ブラウザ/端末の既存セッションが無効化されない。

・想定される被害・影響

パスワードを変更しても、既に奪取されたセッションが継続利用される
個人情報漏洩、不正投稿、不正購入、設定変更などが可能
退会・設定変更などの破壊的操作もされ得る

・対策

パスワード変更後にサーバ側でユーザの全セッションを無効化

・備考

なし

[21] ユーザ ID 列挙・プロフィール収集の容易性

- 対象

  ユーザ詳細（画面）
  http://172.16.97.128/user.php?id={uid}

- 危険度

  info

- 解説

  id が連番で、任意の uid を指定してプロフィールを閲覧できるため、順番に走査するだけで全ユーザの公開情報を収集できる。

  例

- 想定される被害・影響

  取得したユーザ情報を元に次の攻撃の足掛かりとする。

- 対策

  ユーザIDをパラメータに表示しない。

- 備考

  [8]や[21]で悪用される

[22] 格納型 XSS 攻撃

・対象

ユーザ詳細（画面）
http://172.16.97.128/user.php?id={uid}

・危険度

High

・解説

ユーザ名にスクリプトを指定することで検索時に任意のスクリプトが実行される危険性がある。

・想定される被害・影響

任意スクリプトの実行

・対策

スクリプトに関する文字をユーザ名に利用できなくする。

・備考

なし

[23] 検索不能文字の処理の不備

- 対象

  ユーザ詳細（画面）
  http://172.16.97.128/user.php?id={uid}

- 危険度

  low

- 解説

  本来、ユーザ名検索は英数字のみで行われるべきにもかかわらず、不正な文字（英数字以外）を入力した場合でもエラーメッセージは表示されるが、検索処理そのものは実行されてしまう。

  @ で検索した場合の例

- 想定される被害・影響

  SQLインジェクションなどの入力値を使った攻撃のリスクの増加

- 対策

  バリデーションエラー時は検索処理を実行しない

- 備考

  なし

[24] フォロー/アンフォローの送信者なりすまし

- 対象

  フォロー操作
  http://172.16.97.128/user.php（type=1, to_id, from_id, csrf_token）
  アンフォロー操作
  http://172.16.97.128/user.php（type=0, to_id, from_id, csrf_token）

- 危険度

  Medium

- 解説

  脆弱性についての解説を記載

  POST パラメータ内のto_idおよびfrom_idをそのまま用いてフォロー/アンフォロー処理を行っているため、値を書き換えることで他人になりすましてフォロー/アンフォローを行うことができる。

  POST パラメータ内の to_id および from_id をクライアント側の入力値そのままで処理しているため、これらの値を書き換えることで 任意のユーザになりすまし、フォロー／アンフォロー操作を実行できてしまう。

  例えば、test1（id=31）が test2（id=32）をフォローする際、攻撃者がリクエスト中の from_id を test3（id=33）の値に書き換えて送信すると、test3 のアカウントが test2 をフォローしたことになる 挙動が確認できる。

  書き換え前

  ```
  type=1&to_id=32&from_id=31&csrf_token=72260d77131a711eebd56e5a381f6378bbf21055
  ```

  書き換え後

  ```
  type=1&to_id=32&from_id=33&csrf_token=72260d77131a711eebd56e5a381f6378bbf21055
  ```

  test3の画面を見るとtest2が勝手にフォロー状態にされてしまっている

・想定される被害・影響

任意の他人ユーザ間のフォロー/アンフォロー関係を改ざんできる

・対策

サーバ側で to_idとfrom_id を受け取らず、セッションのユーザIDを使用して処理する

・備考

なし

[25] 受信者制限のバイパス

- 対象

  メール送信（画面）
  http://172.16.97.128/sendmail.php

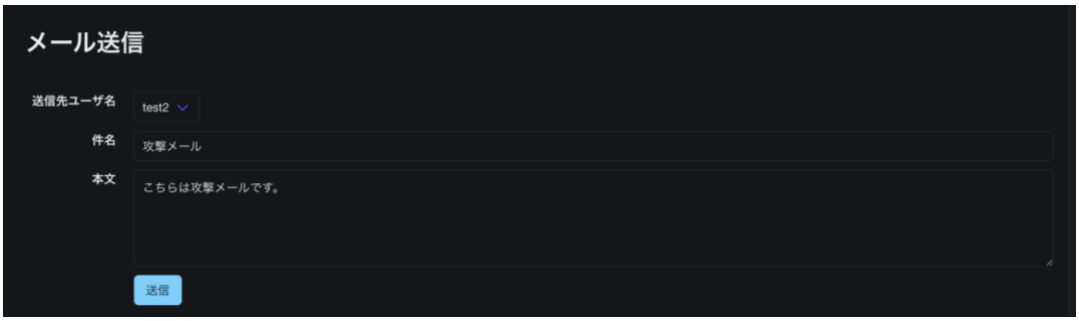
- 危険度

  low

- 解説

  本来は、相互フォローから送信先を選ぶ仕様となっているが、メッセージ送信処理において、サーバーで送信先ユーザーが相互フォロー関係にあるか確認していない。そのため **to_id** を書き換えることで任意のユーザにメールを送信できる。

  例としてこのようなメールを送信する。



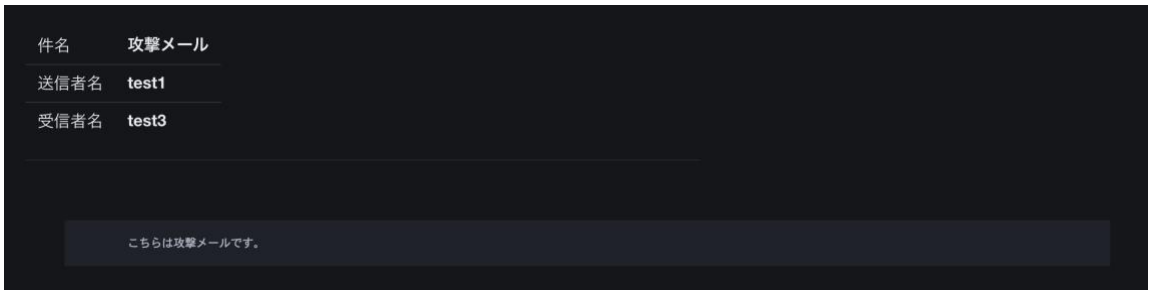
  test2というユーザに送信する場合、**to_id** は32になっている。

```
csrf_token=72260d77131a711eebd56e5a381f6378bbf21055&to_id=32&title=%E6%94%BB%E6%92%83%E3%83%A1%E3%83%BC%E3%83%AB&message=
%E3%81%93%E3%81%A1%E3%82%89%E3%81%AF%E6%94%BB%E6%92%83%E3%83%A1%E3%83%BC%E3%83%AB%E3%81%A7%E3%81%99%E3%80%82&csrf_token=
72260d77131a711eebd56e5a381f6378bbf21055
```

  test3というユーザに送信したいので、**to_id** を33に書き換える。

```
csrf_token=72260d77131a711eebd56e5a381f6378bbf21055&to_id=33&title=%E6%94%BB%E6%92%83%E3%83%A1%E3%83%BC%E3%83%AB&message=
%E3%81%93%E3%81%A1%E3%82%89%E3%81%AF%E6%94%BB%E6%92%83%E3%83%A1%E3%83%BC%E3%83%AB%E3%81%A7%E3%81%99%E3%80%82&csrf_token=
72260d77131a711eebd56e5a381f6378bbf210553
```

  メールを送信すると、test3が先ほどのメールを受信する結果となった。



- 想定される被害・影響

  相互フォローしていないユーザに対しても任意のメールを送ることができる。そのため、攻撃者は任意の相手に攻撃メールを送信できる。

・対策

サーバ側で **to_id** の値を検証する。

・備考

なし

[26] メールの入力バリデーション不足

・対象

メール送信（画面）
http://172.16.97.128/sendmail.php

・危険度

low

・解説

メッセージ送信機能において、タイトルや本文の長さや送信回数に制限がない。

・想定される被害・影響

悪意のあるユーザーが何度も大量のメッセージを送ると、サーバやデータベースに過剰な負荷がかかる。また一種のメール爆弾のような攻撃が可能。

・対策

- 文字数を制限する
- 送信回数の制限を設ける

・備考

その他上記に当てはまらないが報告すべきことがあれば記載

[27] 存在しないユーザ ID に対してのメール送信の不具合

・対象

メール送信（画面）
http://172.16.97.128/sendmail.php

・危険度

low

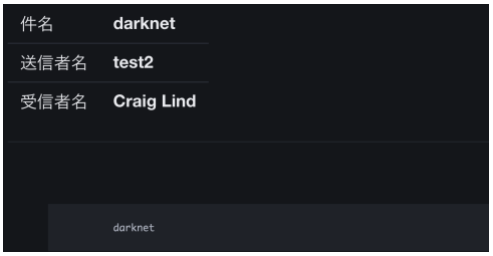
・解説

メッセージを送信する際に、送信先のユーザIDが実際に存在するかどうか確認していない。そのため、存在しないユーザIDに対してメールを送信することができる。

例として、**to_id** に存在しないユーザーID **9999** を入力して送信する。

```
csrf_token=d29a20f663f514be7e067e048e7f7a020e012131&to_id=9999&title=dar
```

ユーザID9999のユーザが存在しないにも関わらず、送信が成立し、受信者名「Craig Lind」と表示される。



これはユーザIDが最も小さいユーザに送信されており、今回の場合ユーザID1と2のユーザが削除済みだったため、ユーザID3の「Craig Lind」にメールが送信された。

・想定される被害・影響

意図しない相手へメールが送信され、機密情報が漏洩する。

・対策

送信先ユーザIDの存在確認を実施し、存在しない場合はメール送信を不可にする。

・備考

なし

[28] メール送信による格納型 XSS 攻撃

- 対象

  メール送信（画面）
  http://172.16.97.128/sendmail.php
  メール閲覧（画面）
  http://172.16.97.128/readmail.php?id={mid}

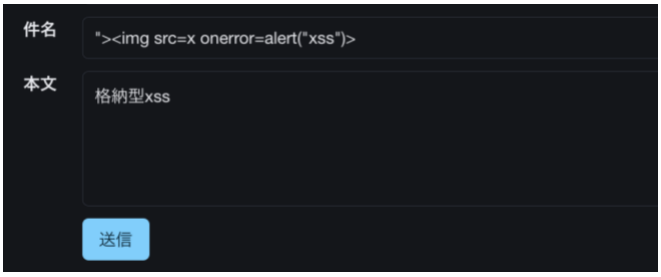
- 危険度

  High

- 解説

  メール送信および返信時のタイトルに "><img src=x onerror=alert(1)> などのスクリプトを入力し送信した後に、トップページ及び当該の受信メールを開くことでスクリプトが実行される。

  例えば、このような内容を入力して送信する。



  ページ読み込み次にスクリプトが実行される。

- 想定される被害・影響

  攻撃者の任意のスクリプトを相手ブラウザで実行することができる。

- 対策

  エスケープ処理を行う。

・備考

その他上記に当てはまらないが報告すべきことがあれば記載

[29] 他人のメールの無断閲覧および削除

- 対象

  メール閲覧（画面）
  http://172.16.97.128/sendmail.php
  メール削除（画面）
  http://172.16.97.128/sendmail.php（mode=delete, csrf_token）

- 危険度

  High

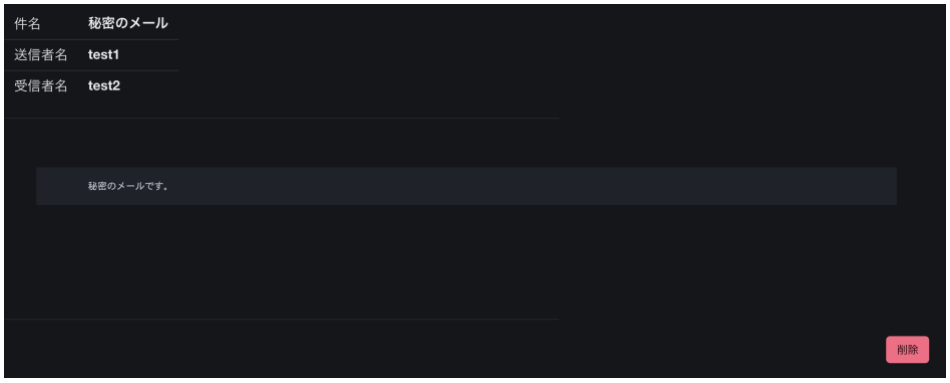
- 解説

  メール閲覧画面のURLのパラメータに任意のiidを指定することで、指定したメールを閲覧および削除することができる。

  例えば test3ユーザでログインしている状態で、このようにidを指定する。

  ```
  172.16.97.128/readmail.php?id=5
  ```

  すると本来、test1とtest2のユーザしか閲覧できないメールをtest3が閲覧することができる。
  また、画像右下の削除ボタンを押すことで対象ユーザ以外でもメールの削除が可能である。

- 想定される被害・影響

  他人のメールの無断閲覧および削除

- 対策

  サーバで閲覧者が送受信者のどちらかであることを確認する

- 備考

  なし

[30] 商品購入価格の改変

- 対象

  商品購入
  http://172.16.97.128/product.php?mode=buy&id=<商品ID>

- 危険度

  High

- 解説

  商品購入フォームのhiddenのpriceプロパティを書き換えることで任意の価格で商品を購入することができる。

  例えば、100ポイントを所持している状態で、こちらの300ポイントの商品を購入する場合、

  Valueの値を300から0に書き換える。

  変更前

```
<input type="hidden" name="price" value="300">
```

  変更後

```
<input type="hidden" name="price" value="0">
```

  価格を書き換えたことでポイントを消費することなく商品の購入が成功する。

・想定される被害・影響

商品の不正購入

・対策

サーバ側で商品価格のチェックを行う。

・備考

なし

[31] 商品検索欄の反射型 XSS 攻撃

- 対象

  商品検索
  http://172.16.97.128/product.php?title={q}

- 危険度

  High

- 解説

  商品検索フォームに入力したスクリプトがそのまま解釈されてしまい、任意のスクリプトを実行することができる。

  例えば、商品検索フォームに以下のようなスクリプトを入力し検索する。

  ```
  "><script>alert(xss)</script>
  ```

  すると、スクリプトが実行されてポップアップに xss と表示される。

- 想定される被害・影響

  任意のスクリプトの実行

- 対策

  入力値のエスケープ処理を行う

- 備考

  特になし。

[32] 新規出品の商品名欄の反射型 XSS 攻撃

・対象

商品出品

http://172.16.97.128/product.php?mode=make&id=new

・危険度

High

・解説

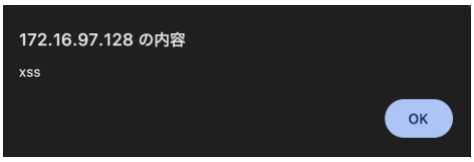
商品名欄に以下のようなスクリプトを入力し出品する。

```
"><script>alert(xss)</script>
```

また、画像を添付した場合属性値インジェクションを行える。

```
a onload=alert("xss")
```

確認画面で以下のようにスクリプトが実行される。



・想定される被害・影響

任意のスクリプトの実行

・対策

入力のエスケープ処理を行う。

・備考

なし

[33] 商品検索時のワイルドカード利用による Dos 攻撃

- 対象

  商品検索
  http://172.16.97.128/product.php

- 危険度

  info

- 解説

  商品検索欄の入力に含めた % と _ がそのまま LIKE のワイルドカードとして働く。
  全件の商品を取得することができる。

- 想定される被害・影響

  DBの負荷

- 対策

  文字長・文字種のサーバ側バリデーション

- 備考

  特になし。

[34] CSRF トークンの認証不備

- 対象

  商品出品
  http://172.16.97.128/product.php?mode=make&id=new

- 危険度

  Medium

- 解説

  商品出品時にCSRFトークンを改竄してリクエストを送信しても、問題なくリクエストが通る。

  例えば商品出品時に以下のようなcsrf_tokenがある場合、

```
<input type="hidden" name="csrf_tokon" value="ced7bcd9d17f64ed042a5f726563f46e6e6bb4b3">
```

  値をこのように空に書き換えて送信する。

```
<input type="hidden" name="csrf_tokon" value="">
```

  CSRFトークンがからにも関わらず、以下のように商品の出品が成功する。

- 想定される被害・影響

  なりすまし出品

- 対策

  CSRFを検証

- 備考

  なし

[35] ユーザ操作のレースコンディション（並列実行時の整合性欠如）

- 対象

  商品購入
  http://172.16.97.128/product.php?mode=buy&id=<商品id>
  クーポン適用
  http://172.16.97.128/coupon.php

- 危険度

  Medium

- 解説

  ユーザ操作時にレースコンディションが発生する。例えば、同一商品を複数のユーザーが同時に購入した場合、どちらも購入が成功する。

- 想定される被害・影響

  誤った操作が行われる。

- 対策

  トランザクションや行ロックの導入

- 備考

  2 人のユーザが誤差 1 秒程度の間に操作した場合に発生した。

  同じクーポンコードの並列使用などあらゆる操作で重複が起こる。

[36] SQL インジェクション攻撃

- 対象

  商品一覧/検索（画面）
  http://172.16.97.128/product.php?title={q}
  自分の出品一覧（画面）
  http://172.16.97.128/product.php?mode=own&title={q}
  ユーザ詳細（画面）
  http://172.16.97.128/user.php

- 危険度

  High

- 解説

  検索欄に入力したSQL文がそのまま解釈されてしまい、攻撃者の任意のSQL文を実行することができる。

  例えば、idが8の商品を取得したい場合、**' AND id=8 –** と入力して検索することで以下のようにIDが8の商品が結果に表示される。

- 想定される被害・影響

  任意のSQL文の実行

- 対策

  入力値のプレスホルダー処理を行う。

- 備考

  なし

[37] CSV ファイル名の情報露えいに伴う無断ダウンロード

・対象

CSVダウンロード
http://172.16.97.128//csvdownload.php?filename={uid}_YYYYMMDDhhmmss.csv

・危険度

info

・解説

CSVのファイル名が ユーザーID_YYYYMMDDhhmmss の形式になっているため、他のユーザが簡単に推測することができる。
以下のように推測したファイル名をパラメータに含めてアクセスすることで他のユーザのCSVファイルを勝手にダウンロードすることができる。

例 ユーザIDが **32** で **2025年11月12日 20:05:08** に出力した場合

```
http://172.16.97.128/csvdownload.php?filename=32_20251112200508.csv
```

・想定される被害・影響

他のユーザのCSVファイルをダウンロードできる。

・対策

- ファイル名を推測困難なものに変更する
- 特定のユーザのみファイルのダウンロードが可能にする

・備考

なし

[38] CSV インジェクション

・対象

商品出品
http://172.16.97.128/product.php?mode=make&id=new
CSVダウンロード
http://172.16.97.128//csvdownload.php?filename={uid}_YYYYMMDDhhmmss.csv

・危険度

High

・解説

CSV内のセルが「=」「+」「-」「@」「¥t」等で始まると、Excel等が数式として実行し、任意の数式、関数を実行させることができる。商品タイトルなどに悪性ペイロードを含めておき、ユーザがダウンロードしたCSVを開くと実行される。

例えば =SUM(3+1) という商品名で商品を出品しておく。

次に、出品商品一覧をCSV出力ボタンを押した後に、該当ファイルをダウンロードする。

ダウンロードしたファイルを開くと以下のように商品名がSUM関数として評価されている。

	A	B	C	D	E
1	商品名	商品説明	販売ポイント	購入制限	状態
2	=SUM(3+1)		¥100	制限なし	販売中

・想定される被害・影響

任意の数式、関数を実行させることができる

・対策

エクスポート時に先頭が [= + - @ ¥t] のセルを適切にエスケープ処理する

・備考

[35]CSVファイル名の情報漏洩に伴う無断ダウンロード と組み合わせることで任意の数式を含んだCVSファイルを他のユーザにダウンロードさせ、実行させることができる。

[39] CSV ファイルのダウンロードによるディレクトリトラバーサル

・対象

CSVダウンロード
http://172.16.97.128/csvdownload.php?filename={uid}_YYYYMMDDhhmmss.csv

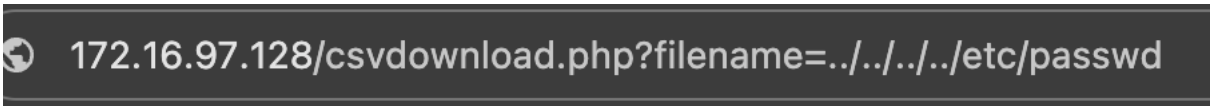
・危険度

High

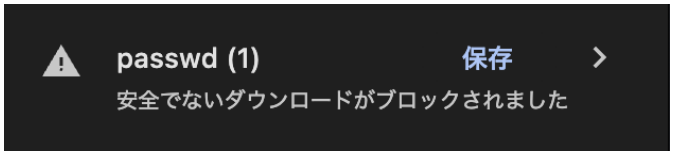
・解説

URLにパスを含めることでサーバ上の任意のファイルをダウンロードすることができる。

例えば以下のようにパスを入力する。



以下のようなURLにアクセスすることで password情報のファイルを取得することができた。



・想定される被害・影響

任意のサーバファイルのダウンロードによる情報漏洩

・対策

エスケープ処理を行う。

・備考

なし

[40] CSV ファイルの大量生成

- 対象

  自分の出品CSV出力
  http://172.16.97.128/product.php?mode=own'（output=csv）

- 危険度

  info

- 解説

  CSRF 防御が存在しない。同一オリジンから勝手に叩けるため、多量の CSV を生成してストレージを消費させる DoS の足掛かりになる。

  以下のようなコードを実行することで大量のCSVファイルを作成することができる。

```
for i in {1..10}; do
        curl -s -H 'Cookie: PHPSESSID=<ブラウザの値>' -d 'output=csv' \
        'http://172.16.97.128/product.php?mode=own' > /dev/null
        sleep 1.2
done
```

- 想定される被害・影響

  Dos攻撃の足掛かり

- 対策

  CSRFを検証

- 備考

  なし

[41] クーポン自動入力リンクの悪用

- 対象

  クーポン（画面）
  http://172.16.97.128/coupon.php?coupon.code=XXXX&coupon.amount=YYYY

- 危険度

  info

- 解説

  URLのパラメータの値が、そのままHTMLおよび入力欄に埋め込まれる。もしもURLに危険なスクリプトが含まれているかつそれをそのまま解釈してしまう場合、URLをクリックするだけで実行される危険性がある。

  例えばこのようなURLにアクセスする。

  アクセスした時に、パラメータがそのまま埋め込まれてしまっている。

- 想定される被害・影響

  意図しないスクリプトの自動実行の可能性

- 対策

  パラメータの値をページ上に埋め込まない。

- 備考

  現時点ではスクリプトは実行されず直接的な影響はない。

[42] 管理者機能の認可不備

・対象

クーポン（画面）
http://172.16.97.128/____generate_coupon.php
管理メニュー（画面）
http://172.16.97.128/____specific_administrative_functions.php

・危険度

High

・解説

一般ユーザが管理者機能にアクセスすることができる。例えば、クーポン（画面）にアクセスすることでユーザは任意のクーポンを発行することができる。

・想定される被害・影響

管理者機能への不正アクセス

・対策

管理者機能にアクセス権限をつける

・備考

なし

[43] 不適切なクーポンの発行

- 対象

  クーポン（画面）
  http://172.16.97.128/____generate_coupon.php

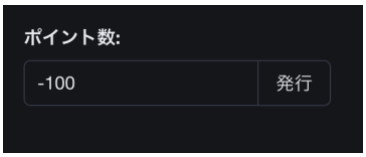
- 危険度

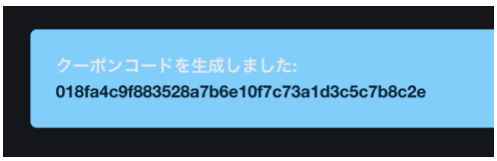
  High

- 解説

  負数や少数など不適切な値を入力しても、クーポンを発行することができる。負数で発行したクーポンを利用することで利用者のポイントを減少させることができる。

  例として -100 を入力してクーポンを発行する。





  次に、所持ポイントが 100 の状態で発行したクーポンを利用する。

  クーポンを利用後、所持ポイントを確認すると 0 ポイントになっていることが確認できる。

・想定される被害・影響

不適切な値のクーポンを発行・利用することでポイントを操作可能になる。

・対策

サーバ側で厳格なバリデーションを行い、非負整数のみ発行可能にする。

・備考

[38]管理者機能の認可不備 と合わせることでリスクが増大する。

[44] パスワード変更後のセッションの再生成の欠如

- 対象

  パスワード変更完了
  http://172.16.97.128/accountPasswordComplete.php

- 危険度

  Medium

- 解説

  パスワード変更後も同一セッションIDを利用。

- 想定される被害・影響

  パスワードを変えても乗っ取りが終わらない。

- 対策

  変更成功時に必ずセッションID再生成する。

- 備考

  なし

[45] ID 指定による任意画像の閲覧

- 対象

  画像取得（本番画像）
  http://172.16.97.128/img.php?id={商品id}

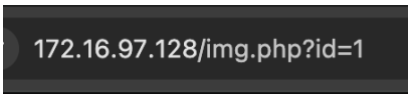
- 危険度

  Medium

- 解説

  商品idを指定したURLにアクセスすることで、購入者限定といったような特定のユーザ飲み閲覧可能な画像が、誰でも閲覧できてしまう。

  例えば以下のようなURLにアクセスする。

  以下のように指定した商品ID 1 の商品画像を閲覧することができる。

- 想定される被害・影響

  個人情報や機密素材を含む画像を第三者に閲覧されてしまう。

- 対策

  商品IDを直接指定して閲覧することをできなくする。また、画像閲覧時にアクセス権限を確認する。

- 備考

  その他上記に当てはまらないが報告すべきことがあれば記載

[46] セキュリティコード画面の露出

- 対象

  セキュリティコード計算（画面）
  http://172.16.97.128/secode.php

- 危険度

  High

- 解説

  本来公開されるべきではないセキュリティコードの計算画面が公開されており、パスを指定するだけで誰でもアクセスすることが可能になってしまっている。

  /secode.php のパスにアクセスすることで以下のような画面が表示される。

- 想定される被害・影響

  セキュリティコードの漏洩

- 対策

  ユーザがアクセスできないようにページの公開をやめる。

- 備考

  なし

[47] セキュリティコード生成ロジックの脆弱性

- 対象

  セキュリティコード計算（画面）
  http://172.16.97.128/secode.php

- 危険度

  low

- 解説

  セキュリティコードは、16進数の6桁で生成されており **16,777,216通り** と総当たり空間が非常に小さい。そのため、コードの衝突や復元リスクが高い。

- 想定される被害・影響

  生成したコードの衝突や復元リスク

- 対策

  セキュリティコード生成ロジックを複雑にする。

- 備考

  なし

[48] セキュリティコード送信時の GET メソッド使用

- 対象

  セキュリティコード計算（画面）
  http://172.16.97.128/secode.php

- 危険度

  High

- 解説

  ?card=xxxx... のようにカード番号をURLで受け取る仕様になっている。

  ```
  172.16.97.128/secode.php?card=23423929292342929
  ```

  そのため、カード番号がログに残ってしまい漏洩するといった問題が発生する。

- 想定される被害・影響

  カード番号がブラウザ履歴、ログ、Referer経由などによって漏洩してしまう。

- 対策

  機密性の高い情報を扱う場合、GETメソッドではなくPOSTメソッドを利用する。

- 備考

  なし

[49] キャッシュ制御の不在による機密情報の漏洩

- 対象

  セキュリティコード計算（画面）
  http://172.16.97.128/secode.php

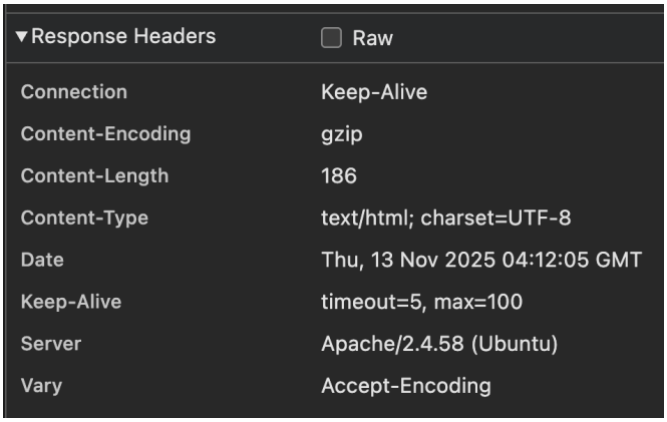
- 危険度

  Medium

- 解説

  Cache-Control / Pragma ヘッダが設定されていない。GETメソッドを利用しているため、セキュリティコードをキャッシュされてしまう可能性がある。

  レスポンスヘッダを見てみるとキャッシュ制御のヘッダが存在していない。



- 想定される被害・影響

  機密情報のキャッシュや、同一URLへ再アクセスした別ユーザへ応答が再送信される恐れがあり、それによって機密情報の意図しない漏洩が起こる可能性がある。

- 対策

  Cache-Control / Pragma ヘッダを設定する。

- 備考

  なし

[50] HTTP パラメータ汚染（HPP）

・対象

ユーザリスト
http://172.16.97.128/userlist.php
画像
http://172.16.97.128/img.php
CSVダウンロード
http://172.16.97.128/csvdownload.php
管理メニュー（画面）
http://172.16.97.128/____specific_administrative_functions.php
フォロー操作
http://172.16.97.128/user.php
メール送信・削除
http://172.16.97.128/readmail.php
商品一覧
http://172.16.97.128/product.php
クーポン適用
http://172.16.97.128/coupon.php
ログアウト
http://172.16.97.128/logout.php
セキュリティコード計算
http://172.16.97.128/secode.php

・危険度

info

・解説

パラメータを複数指定（card=a&card=b）した際、想定外の値が採用されてしまう場合がある。

・想定される被害・影響

検証ロジックのすり抜け、キャッシュ汚染。

・対策

単一値を強制、複数指定は400で拒否する。

・備考

なし

[51] phpMyAdmin の外部公開

- 対象

  phpMyAdmin
  http://172.16.97.128:5555

- 危険度

  High

- 解説

  本来、外部に公開してはいけない phpMyAdmin の画面が公開されてしまっている。

  172.16.07.128:5555 にアクセスする。

  以下のように phpMyAdmin の画面が表示される。

- 想定される被害・影響

  ログインを突破された場合、攻撃者はデータベースを自由に操作することが可能になる。

- 対策

  phpMyAdmin のような管理者向けの画面は外部に公開しない。

- 備考

  プライバシー配慮として本調査ではDBに対してログインなどの侵入調査は行わない。

[52]　Content Security Policy（CSP）の未設定

・対象

全てのページ
http://172.16.97.128

・危険度

High

・解説

CSPはブラウザに対し、

- 許可されるDOM操作やインラインスクリプトの使用可否
- 許可されたスクリプトのロード元
- Iframe 埋め込み可否
- 外部リソースの制御

を宣言する仕組みであり、Webアプリケーションの クライアント側攻撃に対する防御機構である。
当該アプリケーションでは、HTTPレスポンスヘッダに Content-Security-Policy（CSP） が設定されていない。CSPが未設定の場合、他の攻撃に対するリスクが増大する。

ヘッダー内にCSPが存在していない。

Cache-Control	no-store, no-cache, must-revalidate
Connection	Keep-Alive
Content-Encoding	gzip
Content-Length	1375
Content-Type	text/html; charset=UTF-8
Date	Thu, 13 Nov 2025 01:59:37 GMT
Expires	Thu, 19 Nov 1981 08:52:00 GMT
Keep-Alive	timeout=5, max=100
Pragma	no-cache
Server	Apache/2.4.58 (Ubuntu)
Vary	Accept-Encoding
X-Xss-Protection	0

・想定される被害・影響

CSP未設定により、XSSを起点としたあらゆる攻撃が無制限に成立してしまう。

・対策

HTTPレスポンスヘッダーにCSPを追加し、許可するリソースの読み込み元を制限する。

・備考

なし

[53] HTTP を利用した通信および HSTS の未設定

・対象

全てのページ
http://172.16.97.128

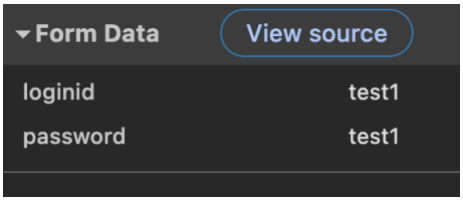
・危険度

High

・解説

ユーザーとサーバー間の通信が暗号化されておらず、平文状態でデータが送受信されている。

例えばログイン画面では、ログイン ID とパスワードが平文で送信されており、以下のように簡単に盗聴することが可能である。



・想定される被害・影響

秘密情報の盗聴

・対策

SSL/TLS証明書の導入およびHSTSの有効化

・備考

現在、Google等の検索エンジンはHTTPSサイトを優遇しており、SEO（検索順位）の観点からも必須の対応となる。

[54] サーバ情報の漏洩

- 対象

  存在しないページ
  http://172.16.97.128/{存在しないパス}

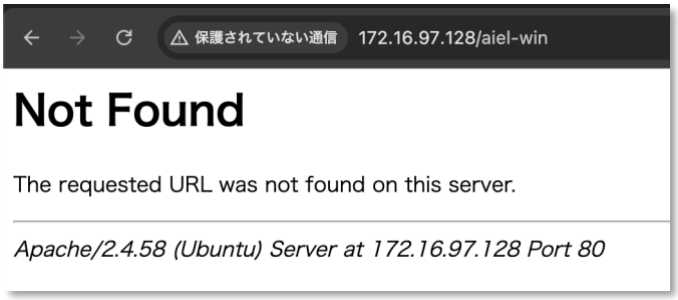
- 危険度

  info

- 解説

  存在しないパスにアクセスした際に表示される画面に、サーバ情報が表示されている。

  /aiel-win にアクセスした場合



- 想定される被害・影響

  サーバ情報が攻撃の準備に利用される可能性がある。

- 対策

  サーバ情報を表示させずに、存在しないパスにアクセスしたことを示す画面を追加する。

- 備考

  なし

[55] 依存ライブラリの脆弱性

・対象

Jquery使用ページ
http://172.16.97.128/static/market/jq.js

・危険度

Medium

・解説

検出されたライブラリ　jquery のバージョン 3.1.0 に脆弱性が存在した。
CVE-2020-11023
CVE-2020-11022
CVE-2019-11358

・想定される被害・影響

脆弱性を悪用した攻撃

・対策

最新のバージョンを利用する、もしくは脆弱な機能を使わない。

・備考

本サイトでは未使用のため影響なしの可能性が大きい。