OWASP Top 10 演習 — 受講者マニュアル

目的：Webアプリケーションの主要脆弱性を安全に体験し、検出と対策を学ぶ。

安全上の注意（必ず守ること）

・ 本演習は教育目的のみである。実環境や第三者のシステムへの攻撃は行わないこと。
・ 演習は指定された演習用サーバ（配布されたEC2）でのみ実施すること。ローカル以外で実行する前に必ず講師の許可を得ること。
・ SSRFや外部参照を行う課題では、講師指定の安全なURL（例：https://example.com）のみ使用すること。
・ 個人情報や実データは扱わない。ログに含めないよう注意すること。

事前準備（確認事項）

・ ブラウザで https://<指定ドメイン> にアクセスできること。
・ 与えられたログイン情報（必要な場合）を講師から受け取っていること。
・ 実行に必要な時間は各モジュール10〜20分。全体で約2時間を想定。
・ 環境のリセットは講師の指示に従うこと（`docker compose down -v` 等）。

問題 1: アクセス制御の不備 (A01)（難易度：★）

目的：管理用ページへ不正にアクセスできる状況を確認し、認可チェックの重要性を説明できること。

手順：
・ ブラウザで /admin にアクセスする。表示内容を確認する。
・ PDFの問題文を読み、どのようにアクセス制御が不足しているかメモする。
・ 対策版（/login -> /admin/secure）で同じ操作を行い、挙動の違いを確認する。

ヒント：フロント側で表示を隠しているだけの場合、サーバ側の認可チェックが不要になっていないか確認する。

期待される挙動：脆弱版では誰でも /admin
にアクセスできる。対策版ではログインが必要で403が返る。

模範解答（要点）：原因：サーバ側での権限検査がない。対策：エンドポイント毎に認可チェックを実装し、RBAC/最小権限を適用する。

問題 2: 暗号化の不備 (A02)（難易度：★）

目的：パスワード保管の安全な方式（ハッシュ＋ソルト）を理解すること。

手順：

・ アプリの設定やDBに保存されているパスワードの扱いを確認する（講師が用意した情報を参照）。
・ なぜ可逆的なエンコードが危険かを簡潔に書く。
・ 対策案（bcrypt/Argon2の利用）を1行で記述する。

ヒント：base64はエンコードであって暗号ではない。

期待される挙動：可逆な保存は漏洩時に危険。推奨は計算負荷のあるKDFを使用すること。

模範解答（要点）：パスワードはbcryptやArgon2でハッシュ化し、ユニークソルトを付与する。送信はTLSで保護する。

問題 3: インジェクション (A03) （難易度：★★）

目的：SQLインジェクションの原理を理解し、プリペアドステートメントの効果を確認すること。

手順：
・ 脆弱版の /module_sql で、検索フォームに特定の文字列を入力して結果を観察する。
・ 例として ' OR '1'='1 を入力して結果が変わるか確認する。
・ 対策版 /module_sql/secure で同じ入力を実行して、結果が変わらないことを確認する。

ヒント：シングルクォートやセミコロンを含む入力に注意。

期待される挙動：脆弱版では全件取得等が可能。対策版では入力がパラメータ化されるため効果がない。

模範解答（要点）：対策：プリペアドステートメント（パラメータ化）、入力検証、DBユーザの最小権限化。

問題 4: 安全でない設計 (A04)（難易度：★★）

目的：設計段階での脅威モデリングやエラーハンドリングの重要性を理解すること。

手順：

・ ログイン失敗時のレスポンスを確認し、返される情報量をメモする。
・ 異なる失敗原因で異なるメッセージが出るか確認する。
・ 簡潔に改善案（2つ）を記載する。

ヒント：細かすぎるエラーメッセージは情報漏洩につながる。

期待される挙動：ユーザ向けには一般的なメッセージ、詳細は内部ログに限定する。

模範解答（要点）：脅威モデリング・共通エラーメッセージ・レート制限（CAPTCHA等）を設計に組み込む。

問題 5: 設定ミス (A05)（難易度：★）

目的：設定ミスがもたらす情報漏洩リスクを理解し、チェックリストを作成できること。

手順：

・ サーバの公開ディレクトリやデバッグモードの有無を確認する（講師が示す例を参照）。
・ 公開すべきでないファイル（.git, .env 等）を列挙する。
・ チェックリストを3項目作成する。

ヒント：本番と開発の設定が混ざっていないか確認する。

期待される挙動：デバッグは無効、本番用設定のみを公開する。不要ファイルは除去する。

模範解答（要点）：対策：設定分離、CIでの設定チェック、自動スキャンと最小公開。

問題 6: 脆弱・古いコンポーネント (A06)（難易度：★★）

目的：依存関係の管理と脆弱性評価の基本を理解すること。

手順：

・ 配布されたSBOMやrequirementsを見て古いライブラリがないか確認する。
・ 脆弱性情報（CVEやNVD）を確認する方法をメモする。
・ アップデート優先度の基準を1つ示す。

ヒント：CVSSや利用箇所の重要度で優先度を決める。

期待される挙動：重要度の高い脆弱性は早急にパッチ適用する。

模範解答（要点）：依存スキャン、SBOM、定期的な更新・テスト・ロールバック計画を運用する。

問題 7: 認証・識別の不備 (A07)（難易度：★★）

目的：安全なパスワードリセットとセッション管理の要件を理解すること。

手順：

・ 脆弱なリセットリンクの形式を確認する（講師資料参照）。
・ 安全なトークンの要件を箇条書きで3つ書く。
・ セッションCookieの属性（Secure, HttpOnly）を確認する。

ヒント：トークンは推測不可能で短命にする。

期待される挙動：推測困難・一度限り・期限付きのトークンを使用する。

模範解答（要点）：対策：一時トークン、MFAの検討、セッション属性の適切設定。

問題 8: ソフトウェア／データ整合性の失敗 (A08)（難易度：★★★）

目的：外部コードやイメージの整合性確認の必要性を理解すること。

手順：

・ 配布物に署名やハッシュが付与されているか確認する。
・ 署名検証の流れを簡潔に説明する。
・ CIでの整合性チェック案を1つ示す。

ヒント：署名（GPG）やコンテナ署名を確認する。

期待される挙動：配布元の確認、ハッシュ/署名検証、信頼できるレジストリ利用。

模範解答（要点）：対策：署名・ハッシュ検証、SBOM、CIでのチェック導入。

問題 9: ログ・監視の不備 (A09)（難易度：★★）

目的：何をログに残し、どう監視すべきかを理解すること。

手順：

・ ログに何が記録されているか（認証失敗、権限変更等）を確認する。
・ 重要イベントの優先度を3つに分けて記述する。
・ アラートの閾値設計について短くまとめる。

ヒント：個人情報はログに残さない。改ざん防止を考慮する。

期待される挙動：認証失敗・重要設定変更・権限エラーなどを優先してログに残す。

模範解答（要点）：ログは集中管理・改ざん防止・適切な保管期間と併せて運用する。

問題 10: SSRF（サーバサイドリクエストフォージェリ）(A10)（難易度：★★★）

目的：SSRFのリスクを理解し、ホワイトリスト方式の防御を実践できること。

手順：
・ 脆弱版の /module_ssrf
にて講師指定の安全なURL（https://example.com）を指定して結果を確認する。
・ 対策版 /module_ssrf/secure で意図的に不正なホストを入力し拒否されることを確認する。
・ ホワイトリストの利点と欠点を短くまとめる。

ヒント：内部IPやメタデータサービスへのアクセスは禁止されるべきである。

期待される挙動：脆弱版では任意URLを取得できる可能性がある。対策版ではホストが限定される。

模範解答（要点）：ホワイトリスト・DNS検証・内部IPブロック・プロキシ経由のみ許可などを組み合わせる。

総括と環境リセット

各モジュールの要点を復習し、脆弱版と対策版の差を説明できるようにすること。

環境のリセット（講師用）

・ SSHでEC2へ接続
・ 対象ディレクトリへ移動: /home/ec2-user/vuln-lab-extended
・ コンテナ停止とボリューム削除: docker compose down -v
・ 再起動: docker compose up -d

参考リンク

・ OWASP Top Ten: https://owasp.org/www-project-top-ten/
・ OWASP Cheat Sheet Series: https://cheatsheetseries.owasp.org/
・ Certbot - Let's Encrypt: https://certbot.eff.org/