☞ 早めのチェック 3工程によるセキュリティ品質確保 ☜

Webアプリケーション脆弱性対策チェックリスト

改訂 平成22年8月13日
改訂 平成22年7月 9日
平成22年4月16日
独立行政法人情報処理推進機構 セキュリティセンター 企画グループ

この文書は、「IPA セキュアプログラミング講座 Webアプリケーション編、2007年版」（以降「講座」と呼ぶ）にもとづいて、品質保証部門向けに再編集されたWebアプリケーション脆弱性対策チェックリストである。

本書の記述について

本チェックリストには、Webアプリケーション脆弱性対策の複数の項目を、大きく次の3つのカテゴリに分けて配置した。

1. 要件定義工程における保護対象の識別が重要である対策
2. 設計工程における考慮が重要である対策
3. 主に実装工程で実施する対策

各カテゴリの中は、脆弱性の危険度が高いものから低いものへの順に配列した。カテゴリ中の記述の単位は、「1.1」「1.2」等の節である。各節が、「講座」の記事テーマひとつにおおむね対応する。ただし「講座」では、ひとつのテーマが複数の記事に分かれているところがある等の事情から、チェックリストの記述の単位の数と「講座」の記事の数は必ずしも一致しない。

各節は次のように構成した。

- 冒頭で「リスク対策に関わる属性」として、その脆弱性テーマの危険度の高さ、Webアプリケーションの中に要警戒箇所が多いか否か、脆弱性および防御の仕組みの複雑さ等の属性を記述している。
- その次に、必要がある場合は直接のチェックリスト項目に入る前に把握が必要な事項について「予備質問」を記述している。
- その後ろに、チェックリスト本体である「点検事項」の記述が並ぶ。

リスク対策に関わる属性

各節の「リスク対策に関わる属性」には、次の6項目を記述している。

危険度

その脆弱性の危険度。次のいずれか：

- 高　Webサイト全体がいちどに大きな打撃を被るおそれがある
- 中　Webサイトのユーザひとりひとりが個別に被害を被るおそれがある
- 低　攻撃者に、攻撃を成功させるヒントを与えるおそれがある

要警戒箇所

その脆弱性を警戒すべき、Webアプリケーション中の箇所の多さ。次のいずれか：

- 多　脆弱性は、Webサイトの各画面、各入力項目、各出力項目に生じ得る
- 中　脆弱性は、複数ある画面のうちのある程度の割合について生じ得る
- 少　脆弱性が生じるのは数少ない特定の箇所のみである

複雑さ

その脆弱性の攻撃と防御のメカニズムの複雑さ。次のいずれか：

- 大　多くの場合、複数の画面にかかわる仕組みに関する事項である
- 中　主に、単一の画面に対する攻撃および防御にかかわる事項である
- 小　主に、リソースの配置ミスや単純なプログラムミスにかかわる事項である

対策を行う工程

要件定義、設計、プログラミング等の工程のうち、主にどの工程で対策に注力するかを述べている

対策共通部品の効果

プログラミングの際、脆弱性対策共通部品を用いることが効果を発揮するか否かについて述べている

セミナー区分

セキュアプログラミング講座の「セミナー3回コース」と組み合わせた際の対応を述べている

取りうる値: 第1回、第2回、第3回

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
高	1.1 プログラムを通じたファイル流出対策	1.1.1 リスク対策に関わる属性	危険度=高; 要警戒箇所=少; 複雑さ=小; 対策を行う工程=主にプログラミング工程; 対策共通部品の効果=該当箇所が多ければ効果がある; セミナー区分=第1回。	1-2-2 プログラムからのファイル流出対策		
		1.1.2 予備質問	(1) ファイル名またはファイル名を構成する綴りの一部を入力パラメータから受け取り、該当するファイルの内容をレスポンスとしてブラウザへ返す場面があるか？			
		1.1.3 攻撃パターン対策				
		1.1.3.1 ファイル名またはファイル名の綴りの一部を入力パラメータから受け取り、該当するファイルの内容をレスポンスとしてブラウザへ返す場面がある場合	(1) 絶対パスや、上位ディレクトリ表現を用いた攻撃パターンを、少なくとも次の仕様で排除しているか？ a) [仕様] サーバOSがUnixやGNU/Linuxの場合、ファイル名の入力パラメータ値の中にスラッシュ「/」が含まれているものを受理しない b) [仕様] サーバOSがWindowsの場合、ファイル名の入力パラメータ値の中にスラッシュ「/」、逆スラッシュ「¥」、コロン「:」のいずれかが含まれているものを受理しない。			
		1.1.4 管理の強化	(1) プログラマの自己裁量によってファイル名入力パラメータの新たな設置を禁止し、ソースコードを点検して違反がないかチェックしているか？			
			(2) ファイル名入力パラメータを受け取ってファイル内容を提供する場面のプログラムの設計・実装に経験豊かな技術者を割り当てているか？			
高	1.2 Webサーバソフトウェアを通じたデータファイル流出対策	1.2.1 リスク対策に関わる属性	危険度=高（流出するファイルの重要度によっては=中）; 要警戒箇所=中; 複雑さ=小; 対策を行う工程=要件定義、設計、プログラミング、運用; 共通部品の効果=該当せず; セミナー区分=第1回。	1-2-1 Webサーバからの ファイル流出対策		
		1.2.2 ファイル配置対策	(1) サーバ上の全てのファイルについて、次の区別が明確になっているか？ a) あらゆるユーザに開示してよいファイル b) 特定のユーザのみに開示してよいファイル c) どのユーザにも開示してはならないファイル。			
			(2) あらゆるユーザに開示してよいファイル以外はWebサーバの公開領域に置かないよう、ファイルの配置が設計されているか？			
			(3) Webサーバの公開領域に配置してよいファイルには何があるか、設計において方針が示され、実装工程において詳細なリストが作られているか？			
			(4) 配置してよいとされているファイルのみがWebサーバの公開領域に置かれ、それ以外のファイルが置かれていないか？			
			(5) あるユーザには開示するが、別のユーザには開示しない等のアクセス制御の対象となるデータファイルは、アクセス認可ロジックを備えたプログラムを通じて、内容がユーザへ開示されるか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
高	(1.2)	1.2.3 最小限の権限とアクセスパーミッション	(1) Webサーバソフトウェア、Webアプリケーションサーバ、およびそれらの配下で動作するWebアプリケーションプログラムのすべてを権限の小さなアカウントで稼働させているか？	(1-2-1 Webサーバからの ファイル流出対策)		
			(2) サーバ上の全てのファイルについて、Webサーバソフトウェア、Webアプリケーションサーバ、Webアプリケーションプログラムが稼働するアカウントからのアクセス（読み出しを含む）を一切許さないアクセスパーミッション設定を行っているか？　ただし、次のものは除く。 a) Web公開領域上で一般に公開するファイル b) Webプログラムが内容を読み取る必要のあるファイル c) Webサーバ、Webアプリケーションサーバの諸設定ファイル。			
		1.2.4 運用におけるWeb公開領域の警戒	(1) 本番サーバに関し、保守作業者がファイルのバックアップコピー等新たなファイルを保存してよい場所として、Web公開領域の外の特定の場所が明示されており、保守の際はその場所のみを使う運用が行われているか？　または、そのような運用が予定されているか？			
			(2) Web公開領域に置いてよいファイルのリストが維持されており、リストに載っているもの以外のファイルがないか定期的にチェックする運用が行われているか？　または、そのような運用が予定されているか？			
中	1.3 コンテンツ間パラメータ対策	1.3.1 リスク対策に関わる属性	危険度=中; 要警戒箇所=多; 複雑さ=小; 対策を行う行程=画面間データフロー設計、プログラミング; 対策共通部品の効果=高い; セミナー区分=第1回。	1-2-3 コンテンツ間パラメータ対策		
		1.3.2 コンテンツ間パラメータの共通対策	(1) サイトの構造や内容を露呈するような情報が渡されていないか？例えば、次のような情報: a) サーバ内のファイル名（プログラムからのファイル流出問題が起こりうる） b) SQL文の全体または一部（SQL注入脆弱性が避けられない）。			
			(2) コンテンツ間パラメータの受け渡しには、セッション変数を積極的に用いているか？			
		1.3.3 URLパラメータ、POSTパラメータ、Cookieによる搬送での対策				
		1.3.3.1 URLパラメータ対策	(1) URLパラメータは、Referer:ヘッダやサーバなどのログを通じて情報が外部に出てしまうため、URLパラメータに秘密情報を含めない設計、プログラミングになっているか？　例えば、次のような情報: a) ユーザIDとパスワード b) クレジットカード番号 c) 個人情報 d) プライベートデータ。			
			(2) フォームデータがURLパラメータではなくPOSTパラメータで送信されるよう、<form> タグには必ず method="post" を明記しているか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(1.3)	(1.3.3)	1.3.3.2 POSTパラメータ対策	(1) ユーザ本人による改ざんを警戒し、たとえhiddenであっても権限や識別に関わる情報を含めない設計、プログラミングになっているか？ 例えば、次のようなパラメータ a) 個人識別に関わるパラメータ b) ユーザの権限に関わるパラメータ c) リソースの識別に関わるパラメータ。	(1-2-3 コンテンツ間パラメータ対策)		
			1.3.3.3 Cookie対策	(1) Cookieの属性をより厳しい条件に設定しているか？ 例えば次のような設定： a) domain：なるべく狭い範囲を指定 b) path：なるべく狭い範囲を指定（なるべく長いパス・プレフィックスを指定） c) max-ageまたはexpires：指定しないか、なるべく短い有効期間を設定 d) secure：暗号通信（https:）を使う場合は必ず指定。			
低	1.4 デバッグオプション対策	1.4.1	リスク対策に関わる属性	危険度=低; 要警戒箇所=少; 複雑さ=小; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第1回。	1-2-4 デバッグオプション対策		
		1.4.2	デバッグオプションの活性化対策	(1) 次のいずれかの対策が行われているか？ a) デバッグ出力はすべてサーバ内のログファイルに書き出すようにする b) 開発時はHTMLページ上へのデバッグ出力機能を持たせるが、プログラムが本番サーバに搭載される前にすべてソースコード上から削除する c) 開発時はHTML出力、本番運用時はログファイル出力となるようなデバッグ出力APIと、大本の切り替え機構を設ける。			
		1.4.3	本番サイトへのWebプログラムの設置許可	(1) Webプログラムのソースコードを検査し、デバッグオプションあるいはHTML上にデバッグ出力する設定がないことを確認した上で、本番サイトへのWebプログラムの設置を許可しているか？			
		1.4.4	HTMLコメントへの注意	(1) HTMLコメントに不用意な情報が書かれていないことを確認しているか？例えば次のような内容 a) プログラムのロジック b) プログラムを記述したプログラマの氏名、メールアドレス、会社名 c) その他、ユーザに閲覧させるべきでない情報。			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
低	1.5 プロキシキャッシュ対策	1.5.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=多; 複雑さ=小; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第1回。	1-2-5 プロキシキャッシュ対策		
		1.5.2 予備質問	(1) 異なるユーザにはそれぞれ異なる情報を開示する画面において、Cookieやそのほかの HTTPリクエストヘッダを用いてセッションの追跡を行っているため、複数ユーザ間でHTTPリクエストのURLに違いが生じないというケースがあるか？			
		1.5.3 Cache-Control:レスポンスヘッダの発行	(1) ユーザ本人以外に開示すべきでないコンテンツを、キャッシュサーバやプロキシサーバ等のWebキャッシュ設備を通じて別人が閲覧する機会が生じないよう、当該コンテンツを開示するHTTPレスポンスに次のようなCache-Control:レスポンスヘッダを含めているか？ a) Cache-Control: private b) Cache-Control: no-store c) Cache-Control: no-cache d) Cache-Control: must-revalidate e) 上記の組み合わせ。			
			(2) HTTP 1.0のみを解釈するWebキャッシュ設備の存在を想定し、ユーザ本人以外が閲覧すべきでないコンテンツを開示するHTTPレスポンスには次のPragma:レスポンスヘッダを含めているか？ a) Pragma: no-cache　　。			
低	1.6 Webサーバソフトウェアからのソースコード流出対策	1.6.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=中; 複雑さ=小; 対策を行う工程=設計、実装、運用; 共通部品の効果=該当せず; セミナー区分=第1回。	1-2-1 Webサーバからの ファイル流出対策		
		1.6.2 予備質問	(1) Webアプリケーションの中で、ソースコードをWebサーバへ直接配置するタイプのプログラミング言語を用いているか？　例えば、JSP、Perl、PHP等。			
		1.6.3 ファイル名拡張子対策	(1) ソースコードの形でWebサーバへ配置するプログラムファイルのファイル名拡張子は、インクルードで参照されるものも含め、どれもWebサーバやアプリケーションエンジンによって当該プログラミング言語のファイルであるとして扱われる所定の拡張子であるか？			
			(2) Webサーバやアプリケーションエンジンが当該プログラミング言語のファイルであると認識するもの以外のファイル名拡張子を使用せざるを得ない場合、それらの拡張子についても当該プログラミング言語のファイルである旨、Webサーバやアプリケーションエンジンに設定を追加しているか？			
		1.6.4 本番機におけるソースコード修正の禁止	(1) 本番サーバ上ではアプリケーションプログラムのソースコード修正を禁止し、いったん開発機上で検証したものを本番サーバへリリースする旨をルール化し、それに従っているか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	2.1 ユーザ認証対策	2.1.1 リスク対策に関わる属性	危険度=中; 要警戒箇所=少; 複雑さ=大; 対策を行う工程=プログラミング; 対策共通部品の効果=有用; セミナー区分=第2回。	2-2-1 ユーザ認証対策		
		2.1.2 予備質問	(1) ユーザ認証にはどのような方式を用いているか？ a) アプリケーションが用意したカスタムログインフォームでユーザIDとパスワードを入力 b) HTTPベーシック認証 c) HTTPダイジェスト認証 d) SSLクライアントディジタル証明書 e) 上記以外。			
			(2) ユーザIDの体系（桁数や取り得る値）はどのようなものか？			
			(3) カスタムログインフォームでユーザIDとパスワードを入力する方式の場合、パスワードは次のどちらが発行するか？ a) ユーザ自身がパスワードを考案しシステムへ登録する b) システムが自動でパスワードを発行する。			
			(4) パスワードの有効期間はどのくらいの長さか？			
			(5) パスワードを忘れたユーザを救済する「パスワードリマインダ機能」が存在するか？			
			(6) いちど設定をしておくと、しばらくの間はパスワードを入力せずにログインできる「自動ログイン機能」が存在するか？			
		2.1.3 シークレット	(1) どのユーザ認証方式を使用する場合においても、本人しか提示できない何らかの秘密の情報を入力してもらうことによって、ログインしようとしている人物が別人でないことを確かめているか。			
		2.1.4 ユーザID	(1) ユーザIDは、システムに実在するものの推定が容易でないものになっているか？ ・避けた方がよいユーザID体系の例 a) 1ずつ値が増加していく等の連番 b) 社員番号等、一定の人々の間ではよく知られている識別番号 c) 電子メールアドレス。ただし、セキュリティレベルの高さよりも利便性を重視するサイトでは、電子メールアドレスを用いることが多い。			
			(2) ユーザIDにチェック桁が含まれているか？　すなわち、正規のユーザIDの綴りの中に、ユーザを識別する目的ではなく、ユーザIDの綴りが特定の判定式を満たすよう調整するための桁が埋め込まれていて、攻撃者が試行するユーザIDの多くをデータベースを検索せずに不正なものであるとして検出できる工夫がされているか？			
			(3) ユーザIDにランダム桁が含まれているか？　すなわち、正規のユーザIDの綴りの中に、攻撃者が連番で試行したとき実在するアカウントのユーザIDに該当する確率を下げるための、ランダムに生成した桁が含まれているか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.1)	2.1.5	パスワード		(2-2-1 ユーザ認証対策)		
			2.1.5.1 パスワードの強度	(1) ユーザ自身がパスワードを考案してシステムへ登録する場合、システムは強度の低いパスワードを拒絶するようになっているか？ ・強度の低いパスワードの例 a) ユーザIDと同じ綴りであるか、ユーザIDの綴りに文字列を追加したり綴りを変形させたもの b) 辞書に載っている単語そのものか、それに数字等を付け加えたもの c) 桁数が少ないもの。一概には言えないが、7桁未満のパスワードは推奨できない。			
				(2) システムが自動でパスワードを発行する場合、システムは強度の低いパスワードを発行しないようになっているか？			
			2.1.5.2 パスワードの有効期限とパスワード変更	(1) パスワードには一定の有効期限が定められ、期限が過ぎるとパスワードを変更しなければならないようになっているか？			
				(2) ユーザがパスワードを変更する操作を行う場合、現行のパスワードを用いた認証が行われるか？			
				(3) ユーザがパスワードを変更しようとした際、現行と同じものを新パスワードとして登録することを禁じているか？			
				(4) ユーザがパスワードを変更しようとした際、以前使用していたパスワードを再度新パスワードとして登録することを、何世代か前まで遡って禁じているか？			
			2.1.5.3 パスワードの平文ログ記録回避	(1) ユーザが投入したパスワードは、それが正しい場合でも誤っている場合でも、ログに平文で記録されることのないようになっているか？			
			2.1.5.4 管理者による一般ユーザパスワードアクセスの禁止	(1) システム管理者用のWebページから一般ユーザのパスワードを閲覧することはできず、また、変更もできないようになっているか？			
		2.1.6	カスタムログインフォーム				
			2.1.6.1 アカウントのロックアウト	(1) カスタムログインフォームにおいて、ユーザが何回も連続してログインに失敗した場合、失敗の回数が一定数を超えると、そのアカウントがロックアウトされる（アカウントからユーザが締め出される）ようになっているか？			
				(2) アカウントをロックアウトしている間は、たとえユーザが正しいユーザIDと正しいパスワードを提示してもログインを許さないようになっているか？			
				(3) アカウントのロックアウトは、次のいずれかの方式で解除されるようになっているか？ a) ロックアウトされてから一定時間が経過する b) ユーザから連絡を受けたシステム管理者が解除の操作を行う。			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.1)	(2.1.8)	2.1.6.2 親切すぎないログイン失敗エラーメッセージ	(1) カスタムログインフォームにおいて、ユーザがログインに失敗した際、ログインできなかった理由が詳しく表示されないようになっているか？　例えば、次の要因を「示さない」ようになっているか？ ・ログインを試みた人物への開示を避けるべきログイン失敗要因の例 a) ユーザIDが間違っている b) パスワードが間違っている c) アカウントがロックアウトされている。	(2-2-1 ユーザ認証対策)		
		2.1.7	パスワードリマインダ	(1) パスワードリマインダへ事前設定する際、他者が知り得る情報ではないものを「秘密の質問」と「答え」として登録するよう、強くユーザに促しているか？			
				(2) パスワード忘れを救済する際、次のような方式を避けて実現しているか？ ・避けるべきパスワードリマインダの振る舞いの例 a) 秘密の質問に対するユーザの答えが正しいとき、その場で現行パスワードをWebページに表示する b) パスワード再登録のためのURL等を電子メールでユーザへ送信する際、あらかじめ登録されていたものではなく、ユーザが新たに入力した電子メールアドレス宛にメール送信する c) ユーザへ送信する電子メール中に現行パスワードを平文で掲載する d) 一連のWebページでSSLを不使用。			
				(3) パスワードリマインダを用いてパスワード忘れを救済する際、「秘密の質問」への答えが何回も連続して間違っていて、間違いの回数が一定数を超えた場合、パスワードリマインダをロックアウトするようにしているか？			
				(4) パスワードリマインダがロックアウトされている間は、たとえユーザが正しい「秘密の質問」への答えを提示しても、パスワード忘れの救済処理を行わないようになっているか？			
				(5) パスワードリマインダのロックアウトは、次のいずれかの方式で解除されるようになっているか？ a) ロックアウトされてから一定時間が経過する b) ユーザから連絡を受けたシステム管理者が解除の操作を行う。			
		2.1.8	自動ログイン	(1) 自動ログイン機能が存在する場合、この機能を利用すると他者による「なりすまし」のリスクがあることをユーザに十分説明しているか？			
				(2) 自動ログインを、次のような方式を避けて実現しているか？ ・自動ログインにおいて避けるべき方式の例 a) 平文のユーザIDもしくはユーザIDを（暗号化でなく）エンコードしたものを搭載したCookieを提示すれば、Webサイト側が自動ログインを許す方式 b) 自動ログインの継続期間が長い。例えば、1箇月以上。			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	2.2 アクセス認可対策	2.2.1 リスク対策に関わる属性	危険度=中; 要警戒箇所=多; 複雑さ=大; 対策を行う工程=設計、プログラミング; 対策共通部品の効果=高い; セミナー区分=第2回。	2-2-2 アクセス認可対策		
		2.2.2 予備質問	(1) ログインしているユーザのみにアクセスを許可し、他は禁止すべきコンテンツがあるか？			
			(2) 特定のユーザの種別や役割（ロール）のみにアクセスを許可し、他は禁止すべきコンテンツがあるか？			
			(3) ひとつのページの中で、アクセスの許可・禁止が次の3つ、ユーザ、ユーザが呼び出そうとしているデータ、行おうとしているアクセスの種類（読み出し、書き込み、削除等）の組み合わせに応じて変化し得る場面があるか？　ここには、ユーザ本人のみにアクセスを許可し、他は禁止すべきコンテンツを含む。			
			(4) ユーザ認証にはどの方式が用いられているか？ a) ベーシック認証 b) ダイジェスト認証 c) ログインフォーム d) クライアントディジタル証明書 e) それ以外。			
		2.2.3 アクセス認可ロジックの枠組み	(1) ユーザがWebサイトにログインする手続き（ユーザ認証）および、ログインセッション（ユーザが認証済みである状態）を追跡する仕組みが存在するか？			
			(2) コンテンツを開示するプログラムの冒頭でアクセス認可ロジックが働いて、権限のない人物が当該コンテンツへアクセスすることが防止される形でプログラムが作られているか？			
			(3) そのアクセス認可ロジックは、次の3つのロジック要素がここに並べられている順で実行されるよう配置されているか？　ただし、明らかに必要のないロジック要素は省略できる。 a) ログイン有無によるアクセス許可・禁止――ログインしているユーザにのみコンテンツへのアクセスを許可し、他は禁止する仕組み b) ページ単位のアクセス許可・禁止――特定の種類や役割のユーザにのみコンテンツへのアクセスを許可し、他は禁止する仕組み c) パラメータ単位のアクセス許可・禁止――次の3つ、ユーザ、ユーザが呼び出そうとしているデータ、行おうとしているアクセスの種類（読み出し、書き込み、削除等）について、特定の組み合わせのみを許可し、他は禁止する仕組み。			
			(4) アクセス認可ロジックが参照するユーザIDは、ログイン（ユーザ認証）時に確保されサーバ内で保持されていたもののみが使われ、Webクライアントからユーザ認証を伴わずに供給されたものが使われないようになっているか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.2)	2.2.4 ログイン有無によるアクセス許可・禁止		(2-2-2 アクセス認可対策)		
		2.2.4.1 ログインフォームによってユーザ認証を行い、ページの連続性の維持のためにCookie等で搬送されるセッションIDを利用してユーザのログイン状態を追跡する場合				
		2.2.4.1.1 ユーザがログインに成功した時点において	(1) ユーザがログインに成功した時点でそれまでと異なる新しい値のセッションIDを発行するようにしているか？			
			(2) ログインに成功したユーザのユーザIDをセッション変数またはデータベースに記録しているか？			
			(3) 現在のユーザがログイン済みであることを示すセッション変数または、セッションIDと関連づけられたデータベースレコードを記録しているか？			
		2.2.4.1.2 未ログインユーザからのアクセスを拒絶すべき各コンテンツにおいて	(1) ユーザへコンテンツを開示するに先立ち、ユーザがログイン済みであることを示すセッション変数もしくはセッションIDに関連づけられたデータベースレコードの存在と妥当性を確認しているか？			
		2.2.4.2 ログインフォームによってユーザ認証を行い、ページの連続の維持のためのセッションIDとは別に設けたCookie等の項目でユーザのログイン状態を追跡する場合				
		2.2.4.2.1 ユーザがログインに成功した時点において	(1) ログイン状態を維持するための照合値をCookie等を用いて次の仕様で発行しているか？ a) ユーザがログインに成功するたびに異なる値が発行される b) 第三者の推測が困難であるランダムかつ桁数の多い値が用いられる c) 後でその値が妥当であるか否かを判定できるよう、発行した値をデータベース等に記録するかまたは、妥当性を検証できる判定式が存在する。			
			(2) ログインに成功したユーザのユーザIDをセッション変数に記録しているか？			
		2.2.4.2.2 未ログインユーザからのアクセスを拒絶すべき各コンテンツにおいて	(1) ユーザへコンテンツを開示するに先立ち、ユーザがログイン済みであることを示す照合値の妥当性をチェックしているか？			
		2.2.5 ページ単位のアクセス許可・禁止	(1) どの種類もしくは役割のユーザにどのページの閲覧を許すかを示すドキュメントが存在するか？			
			(2) どの種類もしくは役割のユーザにどのページの閲覧を許すかが、次のいずれかの形でプログラミングされているか？ a) 許可する組み合わせがデータとして記述され、プログラムから参照する方法が用意されている b) プログラムのロジックとして直接記述されている。			
			(3) ページ単位のアクセス許可・禁止の判定に先立ち、ユーザがログイン済みであることを確認しているか？			
			(4) ユーザへコンテンツを開示するに先立ち、現在ログインしているユーザと閲覧しようとしているページの組み合わせが「許可されている」ものであることをチェックしているか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.2)	2.2.6	パラメータ単位のアクセス許可・禁止	(1) 次の3つ、ユーザ、アクセス対象のデータを識別するキー値、ユーザが行おうとしているアクセスの種類（読み出し、書き込み、削除等）のどのような組み合わせが許可されるべきものであるか記述するドキュメントが存在しているか？	(2-2-2 アクセス認可対策)		
				(2) どのユーザ、どのデータ、どのアクセス種類について許可を与えるかが、次のいずれかの形でプログラミングされているか？ a) 許可する組み合わせがデータとして記述され、プログラムから参照する方法が用意されている b) プログラムのロジックとして直接記述されている。			
				(3) パラメータ単位のアクセス許可・禁止の判定に先立ち、ユーザがログイン済みであることを確認しているか？			
				(4) ユーザへコンテンツを開示するに先立ち、次の3つ、現在ログインしているユーザ、アクセス対象のデータを識別するキー値、ユーザが行おうとしているアクセスの種類（読み出し、書き込み、削除等）の組み合わせが妥当であるか検査しているか？			
				(5) ユーザ本人にかかわるデータにのみアクセスを許可すべき場面において、データベースへアクセスするSQL文のWHERE句にユーザIDを用いた制約条件を記述しているか？			
中	2.3 セッション乗っ取りおよびリクエスト強要（CSRF）対策	2.3.1	リスク対策に関わる属性	危険度=中; 要警戒箇所=多; 複雑さ=大; 対策を行う工程=設計、サーバ設定、プログラミング; 対策共通部品の効果=該当せず; セミナー区分=第2回。	2-1-1 リクエスト強要[CSRF]対策 2-1-2 セッション乗っ取り―― セッションIDと侵害手口 2-1-3 セッションID強度を高める 2-1-4 https:の適切な運用 2-1-5 セッションIDのお膳立てへの対策 2-1-6 乗っ取り警戒と被害の不拡大		
		2.3.2	予備質問				
			2.3.2.1 セッション追跡方式	(1) ページの前後関係の追跡にはどのような方式が用いられているか？ a) ベーシック認証 b) ダイジェスト認証 c) Cookie、Webアプリケーション処理系が自動で発行するもの d) Cookie、Webアプリケーションのロジックで発行するもの e) hidden項目 f) URLリライティング g) 上記以外。			
				(2) ユーザのログイン状態の追跡にはどのような方式が用いられているか？ a) ページの前後関係の追跡の仕組みをそのまま使用 b) ユーザのログイン成功時、ページの前後関係の追跡の仕組みで用いているCookieへ、アプリケーションから新たな値を与える c) ページの前後関係の追跡の仕組みとは別に、ユーザのログイン状態を追跡するためのCookieを発行する d) 上記以外。			

危険度	対策テーマ	場面			点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.3)	(2.3.2)	(2.3.2.1)		(3) スクリプト強要（CSRF）対策にはどのような方式が用いられているか？ a) Webアプリケーションエンジン、フレームワーク、ライブラリ等が発行してくれるトークンを照合 b) Webアプリケーションのロジックでページにトークンを埋め込んで照合 c) 上記以外。	（2-1-1 リクエスト強要[CSRF]対策 2-1-2 セッション乗っ取り―― セッションIDと侵害手口 2-1-3 セッションID強度を高める 2-1-4 https:の適切な運用 2-1-5 セッションIDのお膳立てへの対策 2-1-6 乗っ取り警戒と被害の不拡大）		
			2.3.2.2 単純セッションIDの仕様					
				2.3.2.2.1 ページの前後関係の追跡に、Cookie、hidden項目、URLリライティング等が用いられる場合	(1) ページの前後関係を追跡するためのセッションID（以下「単純セッションID」）にはどのような値が用いられるか？			
					(2) 単純セッションIDの桁数は何桁か？			
					(3) 単純セッションIDに用いられている文字種はどのようなものか？			
			2.3.2.3 ログインセッションIDの仕様					
				2.3.2.3.1 ユーザのログイン状態の追跡に、Cookie、hidden項目、URLリライティング等が用いられる場合	(1) ユーザのログイン状態を追跡するためのセッションID（以下「ログインセッションID」）にはどのような値が用いられるか？			
					(2) ログインセッションIDの桁数は何桁か？			
					(3) ログインセッションIDに用いられている文字種はどのようなものか？			
			2.3.2.4 リクエスト強要（CSRF）対策トークンの仕様					
				2.3.2.4.1 ユーザのログイン状態の追跡に、ベーシック認証、ダイジェスト認証、Cookie等が用いられる場合	(1) Webサーバから供給されたフォームを使ってリクエストを投入していることを確認するためのトークン（以下「リクエスト強要対策トークン」）にはどのような値が用いられるか？			
					(2) リクエスト強要対策トークンの桁数は何桁か？			
					(3) リクエスト強要対策トークンに用いられている文字種はどのようなものか？			
			2.3.2.5 クライアントが発行するセッションIDの受け入れ					
				2.3.2.5.1 Webアプリケーション処理系が発行するCookieをページの前後関係の追跡に用いている場合	(1) Webサーバ側が発行したものではない値のセッションIDがWebクライアントから送られてきたとき、そのWebアプリケーション処理系は、そのセッションIDを妥当なものとして受け入れるか？　また、設定によっては受け入れるようになるか？			
			2.3.2.6 https:の使用要領		(1) SSL, TLS等の暗号通信を用いて保護すべきコンテンツが存在するか？			
					(2) 当該Webサイトのコンテンツには、https: を用いてアクセスするページと、http: でアクセスするページが混在しているか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.3)	2.3.3 セッションIDおよびトークンの推測困難さ	(1) ログインセッションID、リクエスト強要対策トークンの値は、それぞれユーザがログインに成功するたびに異なる値が用いられるようになっているか？	(2-1-1 リクエスト強要[CSRF]対策 2-1-2 セッション乗っ取り―― セッションIDと侵害手口 2-1-3 セッションID強度を高める 2-1-4 https:の適切な運用 2-1-5 セッションIDのお膳立てへの対策 2-1-6 乗っ取り警戒と被害の不拡大)		
			(2) 単純セッションID、ログインセッションID、リクエスト強要対策トークンの値には、どれも次のものの使用が避けられているか？ a) ユーザIDそのもの b) 会員番号、社員番号、職員番号等の、他者が知り得る識別番号 c) 連番またはそれに準ずるもの d) コンピュータの時刻の数値またはそれに準ずるもの e) 上記 a～d を何らかの形式でエンコードしたもので、暗号化されていないもの。			
			(3) 単純セッションID、ログインセッションID、リクエスト強要対策トークンは、取り得る値の個数が十分多いか？ 参考　通信回線の速度およびサーバの処理能力の高さにもよるが、単純セッションID、ログインセッションID、リクエスト強要対策トークンが取り得る値の個数は、それぞれ1015通り（毎秒1億回試行して4箇月以上を要する数）より多いことが望ましい。			
		2.3.4 セッションIDのお膳立てへの対策				
		2.3.4.1 ログインセッションの追跡に、ページ前後関係の追跡のためのCookieが兼用で用いられている場合	(1) ユーザがログインに成功した時点で、第三者による推測が困難な、新しいセッションIDが発行されているか？			
			(2) ユーザがログインに成功した時点で、それまで使用されていたセッションIDが確実に無効にされているか？　すなわちログイン成功後、ログイン前に使用していたセッションIDを再び用いてサイトへのアクセスを試みても、当該ユーザのログインセッションへ接続できないようになっているか？			
		2.3.4.2 クライアント発行セッションID対策	(1) Webサーバ側が発行していない値をWebクライアントから受け入れて常に妥当なセッションIDとして扱うWebアプリケーション処理系を、採用しないようにしているか？			
			(2) 設定によっては、Webサーバ側が発行していない値をクライアントから受け入れて妥当なセッションID として扱うことのあるWebアプリケーション処理系において、その機能を不活性化しているか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.3)	2.3.5	リクエスト強要（CSRF）対策		（2-1-1 リクエスト強要[CSRF]対策 2-1-2 セッション乗っ取り―― セッションIDと侵害手口 2-1-3 セッションID強度を高める 2-1-4 https:の適切な運用 2-1-5 セッションIDのお膳立てへの対策 2-1-6 乗っ取り警戒と被害の不拡大）		
			2.3.5.1 ログインセッションの追跡の方式として、ベーシック認証、ダイジェスト認証、Cookieのいずれかが用いられている場合	(1) 更新処理を引き起こす入力フォームには、リクエスト強要対策トークンが埋め込まれていて、第三者が用意した偽のフォームから送られたリクエストではないことの確認後、Webサーバ側における更新処理が行われるようになっているか？　なお、ここでいう入力フォームには、入力項目を持たず、「処理を実行せよ」という主旨のボタンのみが配置されたものも含まれる。			
				(2) リクエスト強要対策トークンの照合ロジックは、当該トークンがリクエストに含まれていないときも動作し、トークンの値が妥当でない旨の判断を下すよう作られているか？　すなわち、「入力データにトークンが含まれているときのみトークンを照合する」のではない造りになっているか？			
			2.3.5.2 ひとつのフォームからのリクエストを1回しか受け付けないことが求められる場合	(1) 更新処理を引き起こす入力フォームにおいて、リクエスト強要対策トークンの値を、ページを送出する都度異なる値にしているか？			
		2.3.6	ログアウトとタイムアウト				
			2.3.6.1 ログアウト機能	(1) サイトにはログアウト機能が用意されているか？			
				(2) 画面は、ユーザがログアウトしやすいようデザインされているか？			
				(3) ユーザがログアウトした時点で、それまで使用していたセッションIDが確実に無効にされているか？　すなわちログアウト後、ログアウト前に使用していたセッションIDを再び用いてサイトへのアクセスを試みても、元のログインセッションへ接続できないようになっているか？			
			2.3.6.2 ログアウトしやすさ（ログインセッション乗っ取り機会の低減）	(1) 各画面にログアウトボタンやログアウトリンクが配置されているか？			
			2.3.6.3 ログインセッションのタイムアウト	(1) ログインセッションにはタイムアウトが存在するか？　すなわち、ユーザが操作を行わずに一定時間が経過したら、ログインセッションが自動で無効になるか？			
				(2) ログインセッションのタイムアウト時間は、十分短いか？ 参考　ログインセッションのタイムアウト時間は、長くても、1時間以内であることが望ましい。			
			2.3.6.4 復帰先画面パラメータを用いた詐欺への対策	(1) ユーザがログインセッション内のページを要求した際、ログインセッションがタイムアウトしているかまたはユーザが元々ログインしていないときにはまずログインフォームへ誘導してから要求された画面へ自動で遷移する仕組みが存在するとき、ログイン成功後の遷移先画面を示すパラメータに不正な値が指定され第三者のWebサイトの画面に遷移するということがないようになっているか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(2.3)	2.3.7	https:の適切な適用		(2-1-1 リクエスト強要[CSRF]対策 2-1-2 セッション乗っ取り―― セッションIDと侵害手口 2-1-3 セッションID強度を高める 2-1-4 https:の適切な運用 2-1-5 セッションIDのお膳立てへの対策 2-1-6 乗っ取り警戒と被害の不拡大)		
			2.3.7.1 保護対象コンテンツの識別と分離	(1) Webサイトの画面体系がデザインされた時点で、暗号通信を用いて保護すべきページが特定され、明記されたドキュメントが存在するか？			
				(2) https: で保護すべきコンテンツと、http: で公開するコンテンツの両方が存在する場合、これらを異なるホストまたは異なるディレクトリを用いて分離する旨明記したドキュメントが存在するか？ また、そのようにコンテンツが分離され配置されているか？			
				(3) ひとつのホストの上に https: で保護すべきコンテンツと http: で公開するコンテンツを混在せざるを得ない場合、https: で保護すべきコンテンツのための Cookie にはsecure 属性が付与されているか？			
			2.3.7.2 サーバディジタル証明書	(1) 本番運用のWebサーバにおいて、サーバディジタル証明書は正規に発行されたものが使用されているか？			
				(2) Webサーバの運用のためのドキュメントの中に、サーバディジタル証明書の有効期限と、有効期限満了の前にサーバディジタル証明書を更新するための諸手続きおよびコンピュータ操作手順が記述されているか？			
			2.3.7.3 サーバソフトウェア設定	(1) 設計において特定された保護すべきページすべてについて暗号通信を行う旨が明記されたドキュメントが存在するか？ また、そのようにサーバソフトウェアが設定されているか？			
				(2) https: を用いて保護されるページは http: では閲覧できないようにする旨が明記されたドキュメントが存在するか？ また、そのようにサーバソフトウェアが設定されているか？			
				(3) サーバが使用するSSL、TLSのバージョンは、SSL 3.0、TLS 1.0、またはそれ以上のものを使用するよう方針を定める旨が明記されたドキュメントが存在するか？ また、そのようにサーバソフトウェアが設定されているか？			
		2.3.8	セッション乗っ取り被害不拡大のための再認証	(1) Webサイトに登録されている個人情報や重要な機密情報の表示と修正、金銭取引の執行、資金移動等、重要な処理を行う前に、ログインセッション中であっても再度ユーザ認証を行い、パスワードを知らないセッション乗っ取り犯がこれらの機能を濫用できない仕組みを設けているか？			
		2.3.9	セッション乗っ取り兆候の警戒	(1) ひとつのログインセッションの途中で、User-Agent:リクエストヘッダの値やWebクライアントのIPアドレスが変化したことをセッション乗っ取りの兆候として捉え、Webサイト管理者への警報の発報、ログへの記録、ユーザへの再ログインの要求等を行っているか？ ただし、ユーザが接続しているネットワーク環境によっては、Webサーバから見たWebクライアントのIPアドレスが一定の値の範囲で変化することは異常でないケースがある。			
		2.3.10	スクリプト注入対策	(1) スクリプト注入（XSS）対策（3.3節に記載）が漏れなく実施されているか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
低	2.4 想定外ナビゲーション対策	2.4.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=多; 複雑さ=大; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第2回。	2-1-7 想定外ナビゲーション対策		
		2.4.2 URL直接呼び出しに対する対策	(1) 一連のページの流れの途中では、すべてPOSTメソッドで次のページへ遷移するようにし、それらのページはGETメソッドでは呼び出せないようにしているか？			
			(2) HTTPリクエスト内のReferer:ヘッダを見て所定のページから遷移してきたことを確認しているか？			
		2.4.3 送信ボタン複数回クリックに対する対策	(1) JavaScriptを用い、送信ボタンのクリック回数カウンタをもつとともに、2回目以降のクリックではデータを送信しないようにしているか？			
			(2) ユーザが「送信」ボタンをクリックした際、JavaScriptでポップアップウィンドウを出してマウスクリックイベントをそのウィンドウで受け止めるようになっているか？			
		2.4.4 いくつかの再送信操作に対する対策	(1) データベース更新処理と更新結果照会処理をふたつのWebプログラムに分け、HTTPのリダイレクトレスポンスでふたつを結びつけて連続実行させる仕組みを導入しているか？			
			(2) 各ページごとに異なる識別情報を埋め込むことによって、同一のフォームからのデータ送信の重複を検出するようにしているか？例えば次のようなロジック a) 入力フォームには、表示するごとに異なる値の識別情報（トークン）を埋め込む b) 識別情報はサーバ側でも控えを保持し、サーバ側で発行した識別情報であり有効なものであることを確認できるようにしておく c) フォームデータを受け取ったWebプログラムはこの識別情報を調べ、それが有効であり、かつ初めて受信するものであるときに限り、データを受理し所定の処理を行う d) 有効でない識別情報を含んでいたり、すでに1度処理した識別情報を含むフォームデータは受理しないようにする e) サーバ側にデータが送られてこなかったフォームの識別情報は、ユーザがログアウトした時点で無効にする f) 識別情報には、ランダム性の高い桁数の多い値を用いるのがよい。単純な連番等では、ユーザがHTTPリクエストをねつ造できるおそれがある。			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
高	3.1 SQL注入対策	3.1.1 リスク対策に関わる属性	危険度=高; 要警戒箇所=多; 複雑さ=中; 対策を行う工程=主にプログラミング工程; 対策共通部品の効果=高い; セミナー区分=第3回。	3-1-1 SQL注入:#1 実装における対策		
		3.1.2 予備質問	(1) 使用するDBソフトウェアの種類とバージョンは何か？			
			(2) DBのプログラミングにおいて、プログラマが直接SQL文を取り扱うか？ またはO/Rマッピング技術等を用いてDBにアクセスするか？			
			(3) DBソフトウェアへアクセスするためのAPIライブラリには何が用いられるか？			
			(4) そのAPIはプリペアドステートメントをサポートしているか？			
			(5) そのプリペアドステートメントAPIは、APIライブラリ内部で特殊記号のエスケープ処理を行う擬似的なものではないことが確認できているか？			
			(6) DBソフトウェアに日本語データを渡す際、どの文字コードが用いられるか？			
			(7) 日本語を表すマルチバイトデータの途中に 0x27「'」や 0x5C「¥」のビットパターンが含まれているとき、DBソフトウェアがこれらをASCII文字「'」や「¥」と誤って認識してしまうことがないか？			
		3.1.3 入力パラメータ値のSQL文への埋め込みの際の対策				
		3.1.3.1 DBのプログラミングにおいて、プログラマは直接SQL文を取り扱わず、O/Rマッピング技術等を用いてDBにアクセスする場合				
			(1) 当該O/Rマッピング等の技術の実装において、SQL注入対策が施されているか？ すなわち、下記1.3.2以降の対策が施されているか？			
		3.1.3.2 DBのプログラミングにおいて、プログラマが直接SQL文を取り扱う場合				
		3.1.3.2.1 DBソフトウェアのAPIが擬似ではないプリペアドステートメントをサポートしている場合	(1) 開発プロジェクトのコーディング規約に、SQL文へパラメータを埋め込む際、プリペアドステートメントを使う旨の規定があるか？			
			(2) 文字列連結によってSQL文の途中にパラメータが埋め込まれる箇所が一箇所も無いようソースコードが記述されているか？			
			(3) プリペアドステートメント機能を用いて埋め込むことのできない文法要素を可変にする目的で文字列連結演算が使用されている場合、埋め込まれる値が、信頼できないソース（Webクライアントからのパラメータ、外部から受信したファイルの内容の一部等）からのものでないことが確実になっているか？			
		3.1.3.2.2 DBソフトウェアのAPIがプリペアドステートメントをサポートしていないか、内部でエスケープ処理を行う擬似的な実装しか持たない場合	(1) 開発プロジェクトのコーディング規約に、SQL文へパラメータを埋め込む際、文脈に応じた特殊記号対策を行う旨の規定があるか？			
			(2) 'xxx'の内側にパラメータ値を埋め込む文脈において、常に次のエスケープ処理が行われているか？ ・[仕様] SQL文に埋め込まれるパラメータの中に含まれる特殊記号に対し、次の置き換えが施されている a) 「'」→「''」(2個) b) 「¥」→「¥¥」。			

危険度	対策テーマ	場面			点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
高	(3.1)	(3.1.3)	(3.1.3.2)	(3.1.3.2.2)	(3) 'xxx'の外側にパラメータ値を埋め込む文脈において、埋め込むパラメ5ータ値に対して次の文字種検査が行われ、妥当でない文字が含まれていた場合はSQL文を組み立てずにエラー処理を行うようになっているか？ ・[仕様] 埋め込むパラメータ値を構成する文字種が次のいずれかに限定されていること a) 整数――負符号「-」および数字(0-9) b) 小数部をもつ数値――負符号「-」、数字(0-9)、小数点「.」 c) 指数部をもつ数値――正負の符号「+」「-」、数字(0-9)、小数点「.」、指数部を示す英字「E」「e」。	(3-1-1 SQL注入:#1 実装における対策)		
					(4) 本来ASCII文字として認識すべきでない 0x27 や 0x5C のビットパターンにDBソフトウェアが反応してしまう場合、これらのビットパターンに関してエスケープ処理を行っているか？			
		3.1.4	エラーメッセージの抑制		(1) DBアクセスに関するエラーが発生したとき、その旨をエラーメッセージとしてブラウザに表示するアプリケーションのロジックを設けないようにしているか？	3-1-2 SQL注入:#2 設定における対策		
					(2) DBアクセスに関するエラーが発生したとき、その旨を知らせるエラーメッセージが自動でブラウザに表示されることがないよう、Webアプリケーション実行処理系を設定しているか？			
		3.1.5	DBソフトウェアの設定		(1) アプリケーションがDBアクセスの際に使用するDBアカウントは、DB管理アカウント以外のものが用いられているか？			
					(2) アプリケーションが用いるDBアカウントへ付与している権限は、SELECT, INSERT, UPDATE, DELETE等のうち必要最小限のものに限定されているか？			
					(3) 検索系画面と更新系画面で、DBアカウントを使い分けているか？　すなわち、更新系画面ではSELECTのみの権限のDBアカウントを用い、更新系画面においてのみINSERT, UPDATE, DELETE等の権限をもつDBアカウントを用いているか？			
					(4) 悪用されたときのリスクが高いストアドプロシージャ――例えば、サーバOSへ任意のコマンドを投入できる等――を削除もしくは停止させているか？			
高	3.2 コマンド注入対策	3.2.1	リスク対策に関わる属性		危険度=高; 要警戒箇所=少; 複雑さ=中; 対策を行う工程=主にプログラミング工程; 対策共通部品の効果=ある程度効果がある; セミナー区分=第3回。	3-1-3 コマンド注入攻撃対策		
		3.2.2	予備質問		(1) 使用するプログラミング言語の種類とバージョンは何か？			
					(2) Webアプリケーションの中で、サーバOS上の別の実行可能ファイルを起動する場面があるか？			
					(3) 別のプログラムを起動する際に使用するAPIは何か？			
					(4) そのAPIの中で「シェル」が動作するか？			
					(5) 動作するシェルの種類とバージョンは何か？			
					(6) Webアプリケーションの中で、文字列データを、あるプログラミング言語のソースコードとして解釈実行させる場面があるか？			
					(7) 解釈実行させるのはどのプログラミング言語のソース文字列か？			
					(8) 解釈実行させるAPIには何が用いられるか？			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
高	(3.2)	3.2.3	シェルに関する対策		(3-1-3 コマンド注入攻撃対策)		
			3.2.3.1 シェル使用の回避	(1) サーバOS上の別の実行可能ファイルを起動する形態を用いずに、アプリケーションを構成するようにしているか？			
				(2) サーバOS上の別の実行可能ファイルの起動が不可欠である場合、その内部で「シェル」が動作しないAPIを用いているか？			
			3.2.3.2 入力パラメータ値対策	(1) 内部で「シェル」が動作するAPIを用いて、サーバOS上の別の実行可能ファイルを起動することが不可欠である場合、そのプログラムを呼び出すコマンド文字列へ埋め込む入力パラメータ値について、文字種上の厳しい制限（英字と数字のみ許可する等）を課しているか？			
		3.2.4	シェル以外のスクリプトエンジンに関する対策				
			3.2.4.1 評価関数使用の回避	(1) スクリプト言語のソースコード文字列を実行時に解釈実行することのできる関数（評価関数）が、プログラム中で使用されずにいるか？　もし、使用されている場合、それはやむをえない必要最小限の使用であるか？ ・[警戒すべき例] PerlおよびPHP のeval() 関数 。			
中	3.3 スクリプト注入（XSS）対策	3.3.1	リスク対策に関わる属性	危険度=中; 要警戒箇所=多; 複雑さ=中; 対策を行う行程=プログラミング; 対策共通部品の効果=高い; セミナー区分=第3回。	3-2-1 スクリプト注入:#2 攻撃の解説 3-2-2 スクリプト注入:#1 対策		
		3.3.2	予備質問	(1) 使用するプログラミング言語やアプリケーションフレームワークは、プログラマが特別な指定をしなくても、スクリプト注入（XSS）対策を自動で行う機能をもつか？　また、自動で行われるスクリプト注入（XSS）対策の仕様はどのようなものであり、次の「文脈」のうちどこまで対応しているか？ a) HTMLの一般のテキスト部分（タグではない部分）であって、<script>...</script>や<style>...</style>の内側でないもの b) HTMLのコメント（<!-- ... --> の内側） c) HTMLのタグの属性値のうち、URLを指定可能な属性、style属性、イベントハンドラ属性を除くもの d) HTMLのタグのstyle属性値および<style>...</style>の内側 e) HTMLのタグのイベントハンドラ属性値 f) HTMLの<script>...</script>の内側 g) HTMLのタグ名およびタグの属性名。			
				(2) Webアプリケーションの入出力データとしてHTMLタグそのものや簡易タグを使う場面があるか？			
				(3) 英語版ページを扱う場面があるか？			
		3.3.3	文脈（プログラムが値を書き出す箇所）別の対策				
			3.3.3.1 HTMLの一般のテキスト部分（タグではない部分）であって、<script>...</script> や <style>...</style> の内側でない箇所へ値を書き出す場合	(1) 次の5種類の特殊記号をエンティティ表現へ置換する対処を行っているか？ ・「<」→ < 「>」→ > 「"」→ " 「'」→ ' 「&」→ & 。			

危険度	対策テーマ	場面		点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	(3.3)	(3.3.3)	3.3.3.2 HTMLのコメント（<!-- ... --> の内側）へ値を書き出す場合	(1) 次の5種類の特殊記号をエンティティ表現へ置換する対処を行っているか？ ・「<」→ < 「>」→ > 「"」→ " 「'」→ ' 「&」→ & 。	(3-2-1 スクリプト注入:#2 攻撃の解説 3-2-2 スクリプト注入:#1 対策)		
			3.3.3.3 HTMLのタグの属性値のうち、URLを指定可能な属性、style属性、イベントハンドラ属性を除く箇所へ値を書き出す場合	(1) 次の5種類の特殊記号をエンティティ表現へ置換する対処を行っているか？ ・「<」→ < 「>」→ > 「"」→ " 「'」→ ' 「&」→ & 。			
				(2) タグの属性値は引用符（「"」あるいは「'」）で囲まれているか？			
			3.3.3.4 HTMLのタグの、URLを指定可能な属性値へ値を書き出す場合	(1) 書き出す値が次のいずれかの条件に合うもののみに制限されているか？ ・「http://」「https://」「/」のどれかで始まっている ・*file* や *dir/file* のような形式の相対パスであって、途中にコロン「:」と解釈される字句（「:」そのもののほか「:」「:」の表現やこれらのバリエーション）を含まない。			(H22/7/9改訂)
			3.3.3.5 HTMLのタグのstyle属性の中へ値を書き出す場合	(1) 書き出される値が次の特殊記号をひとつも含まないよう制限されているか？ ・タグ記号「<」「>」 引用符「'」「"」「`」 セミコロン「;」 コロン「:」 括弧「(」「)」 アンド記号「&」。			(H22/7/9改訂)
			3.3.3.6 HTMLのタグのイベントハンドラ属性へ値を書き出す場合	(1) 書き出される値が次の特殊記号をひとつも含まないよう制限されているか？ ・タグ記号「<」「>」 引用符「'」「"」「`」 セミコロン「;」 コロン「:」 括弧「(」「)」 アンド記号「&」。			(H22/7/9改訂)
			3.3.3.7 <script>...</script>の内側へ値を書き出す場合	(1) 書き出される値が次の特殊記号をひとつも含まないよう制限されているか？ ・タグ記号「<」「>」 引用符「'」「"」「`」 セミコロン「;」 コロン「:」 括弧「(」「)」 アンド記号「&」。			(H22/7/9改訂)
			3.3.3.8 HTMLのタグ名やタグの属性名（英数字名）の箇所へ値を書き出す場合	(1) 書き出される値が、次の種類の文字のみ含むよう制限されているか？ ・英字（a～z A～Z） 数字（0～9） スラッシュ「/」。			
		3.3.4 入出力データとしてのタグを含むHTMLデータ					
			3.3.4.1 入出力データとしてタグを含むHTMLデータを扱わない場合	(1) タグを含むHTML文字列が入出力データに含まれていないことを確認しているか？			
			3.3.4.2 入出力データとしてタグを含むHTMLデータを扱う場合	(1) 入力されたHTMLドキュメントを構文解析し、不要なスクリプトが含まれていないことを確認しているか？			
				(2) 入力されたHTMLドキュメントを出力する際に不要なスクリプトを削除してから出力しているか？			
		3.3.5 文字セットの明示		(1) HTMLドキュメントを内容とするHTTPレスポンスを送出する際、英語版コンテンツであっても、必ず使用文字セットを明示するようにしているか？			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
中	3.4 入力検査漏れ対策	3.4.1 リスク対策に関わる属性	危険度=中; 要警戒箇所=多; 複雑さ=中; 対策を行う行程=詳細設計、プログラミング; 対策共通部品の効果=高い; セミナー区分=第3回。	3-1-4 入力検査もれ対策		
		3.4.2 検査項目の網羅	(1) リクエストで送られてくる全項目に対し入力検査を行っているか？例えば次のような項目： ・フォーム項目（主なもの） a) チェックボックス b) ラジオボタン c) <select> - <option>タグによる選択項目 d) 不可視項目 e) ボタン類 ・HTTPリクエストヘッダ項目（主なもの） f) Cookie: g) Referer: h) User-Agent:　　。			
		3.4.3 入力検査のタイミング	(1) クライアント側スクリプトでの入力検査だけではなく、サーバ側でも入力検査を行っているか？			
		3.4.4 数値範囲検査	(1) PerlやPHP等のプログラミング言語を使用している場合、数値範囲の検査の際、入力された文字に対し文字種検査を行い、数値のみであることを確実に識別できるようにしているか？			
		3.4.5 正規表現の先頭・末尾のマッチング	(1) 正規表現とのマッチングを用いて入力データを検査する際、データの先頭および末尾を表す記号として「^」と「$」ではなく、「＼A」と「＼z」を用いているか？　ただし、これらの表現がサポートされていないJavaScript等の処理系は除く。			
低	3.5 HTTPレスポンスによるキャッシュ偽造	3.5.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=少; 複雑さ=中; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第3回。	3-2-3 HTTPレスポンスによるキャッシュ偽造攻撃対策		
		3.5.2 予備質問	(1) Webアプリケーションの中に、HTTPレスポンスヘッダへ値を書き出す箇所があるか？			
		3.5.3 HTTPレスポンスヘッダ出力対策	(1) HTTPレスポンスヘッダ内へ値を書き出す箇所において、HTTPレスポンスの途中に改行コード（CrおよびLf）が含まれることのないよう、次のいずれかを行っているか？ a) 値の書き出しを中止し、当該HTTPリクエストの処理そのものをエラーとして失敗させる b) 書き出そうとしている文字列の中の改行コードおよび特殊記号類にURLエンコード（%xxの形の16進数表示等）を施してから書き出す c) 書き出そうとしている文字列の中の改行コードを半角スペース等、別の文字に置き換えてから書き出す d) 書き出そうとしている文字列の中から改行コードを削除したものを書き出す。			

危険度	対策テーマ	場面	点検事項	セミナスライドの該当箇所	点検結果（○・×・外）	備考（明細、除外理由 等）
低	3.6 メールの第三者中継対策	3.6.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=少; 複雑さ=中; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第3回。	3-3-1 メールの第三者中継対策		
		3.6.2 パラメータの削減と検査強化	(1) 外部からのパラメータをメールヘッダの内容に指定しない設計および実装になっているか？ 例えば次のような箇所: a) 宛先(To,Cc,Bcc) b) 発信人(From) c) 標題全体(Subject) d) 本文全体。			
			(2) 外部からのパラメータを使用する必要がある場合には、その入力値について厳しく検査を行っているか？ 例えば次のような箇所: a) 宛先(To,Cc,Bcc) b) 発信人(From) c) 標題の一部 d) 本文の一部。			
		3.6.3 メールヘッダへの改行混入防止	(1) 外部からのパラメータをメールヘッダに指定する場合は、改行コード（CrおよびLf）を排除しているか？			
		3.6.4 自動化クライアント対策	(1) 自動化クライアントを利用した迷惑メール大量送信を防ぐために、人間が手動で操作していることをCAPTCHA（メール送信フォームに自動化プログラムによる判読が困難なノイズの入った文字列の画像を表示し、そこから読み取れる文字列をユーザに入力してもらう）の仕組みを用いて判定しているか？			
低	3.7 真正性の主張	3.7.1 リスク対策に関わる属性	危険度=低; 要警戒箇所=多; 複雑さ=中; 対策を行う工程=設計、実装; 共通部品の効果=有用; セミナー区分=第3回。	3-3-2 真正性の主張		
		3.7.2 偽ページ対策	(1) ユーザが偽ページに騙されないように本物か偽物かが判断できるユーザインターフェースになってか？ 例えば次のような項目 本物のページであると判断するための確認項目の例: a) ページのURLやリンク先のURL b) SSL/TLSサーバ証明書の有無 c) SSL/TLSサーバ証明書の内容。			
		2.7.3 本物であることの確認手段の確保	(1) ユーザが本物か偽物かを判断できる情報をすこしでも多く提供するように作っているか？ 例えば次のような項目: a) アドレスバーの表示 b) ステータスバーの表示 c) 右ボタンクリックによるプロパティ確認手段 d) SSL/TLSの導入 e) ブラウザが信頼済みとしている認証局から発行されたSSL/TLSサーバ証明書の利用。			
		3.7.4 リダイレクトのURL作成時の注意	(1) LocationヘッダのURLを組み立てる際に、リダイレクト先URLの作成時には、外部からのパラメータをできるだけ利用しない設計、実装になっているか？			