CONFIDENTIAL

SAMPLE

〇〇〇〇〇株式会社　御中

Web アプリケーション標準診断
診断結果報告書

脆弱性診断後 10 営業日以内に提出いたします。本診断結報告書では診断結果の総括に加え、脆弱性と認められた根拠、脆弱点の技術的な対処方法・対策方法の説明を盛り込んでおります。

〇〇〇〇年〇月〇日

株式会社サイバーセキュリティクラウド

SAMPLE

【資料改編履歴】

日 付	内 容	承 認	作成/更新
○○○○年○月○日	第一版作成	○○	○○

目　　次

1　はじめに

本診断結果報告書は、〇〇〇〇年〇月〇日～〇日の期間で実施した Web アプリケーション標準診断の診断結果についてご報告するものです。

1－1　目的

本診断の目的は、診断対象 Web アプリケーションに対して、リモートからの脆弱性診断を行い、内在する脆弱性を検出することにあります。

また、脆弱性が検出された場合、そのリスク評価および脆弱性への対策を支援する情報の提供も行います。

1－2　診断期間および診断内容

本診断は、表 1-2-1 の日程で実施いたしました。

表 1-2-1　診断期間および診断内容

診断種別	診断期間	診断内容
リモート診断	〇〇〇〇年〇月〇日（10 時から 18 時まで）	リモートにて、脆弱性診断ツールによるスキャン診断および手動での情報収集・診断を実施しました。
	〇〇〇〇年〇月〇日（10 時から 18 時まで）	

1－3　診断対象

本診断における診断対象は以下の通りです。

〇〇〇〇〇サイト　(20 画面)

	画面名称	URL※1	備考 ※2
画面イメージ(1)PC			
1	契約者ログイン②	https://XXXXXXXX/PayEntry.jsp https://XXXXXXXX/PayEntry.jsp	5.画面イメージ(1)PC-①
2	ご契約内容確認③	https://XXXXXXXX/PayEntryDetails.jsp	5.画面イメージ(1)PC-①

	画面名称	URL※1	備考 ※2
3	注意事項④	https://XXXXXXXX/PayEntryDetails2.jsp https://XXXXXXXX/PayEntryKiyaku_inc.jsp	5.画面イメージ (1)PC-①
4	カード情報/ メールアドレス入力⑤	https://XXXXXXXX/PayEntryInput.jsp	5.画面イメージ (1)PC-②
5	ヘルプ： ご利用可能カード⑥	https://XXXXXXXX/PayEntryInputHelp.jsp	5.画面イメージ (1)PC-②
6	ヘルプ： セキュリティコード⑦	https://XXXXXXXX/PayEntryInputHelp2.jsp	5.画面イメージ (1)PC-②
7	入力内容確認⑧	https://XXXXXXXX/PayEntryConfirm.jsp	5.画面イメージ (1)PC-③
8	登録結果確認⑨	https://XXXXXXXX/PayEntryEnd.jsp	5.画面イメージ (1)PC-③
9	契約者ログアウト⑩	https://XXXXXXXX/Logout.jsp	5.画面イメージ (1)PC-③
10	カード情報/ メールアドレス入力⑤ （「カード情報修正」 ボタン）	https://XXXXXXXX/PayEntryInput.jsp	5.画面イメージ (1)PC-②
画面イメージ(2)モバイル			
11	契約者ログイン②	https://XXXXXXXX/PayEntryDetails.jsp	5.画面イメージ (2)モバイル
12	ご契約内容確認③	https://XXXXXXXX/PayEntryDetails.jsp	5.画面イメージ (2)モバイル
13	カード情報/ メールアドレス入力⑤	https://XXXXXXXX/PayEntryInput.jsp	5.画面イメージ (2)モバイル
14	入力内容確認⑧	https://XXXXXXXX/PayEntryConfirm.jsp	5.画面イメージ (2)モバイル
15	登録結果確認⑨	https://XXXXXXXX/PayEntryEnd.jsp	5.画面イメージ (2)モバイル

	画面名称	URL※1	備考 ※2
16	契約者ログアウト⑩	https://XXXXXXXX/Logout.jsp	5.画面イメージ (2)モバイル
17	カード情報/メールアドレス入力⑤ （「画面訂正・入力画面へ戻る」ボタン）	https://XXXXXXXX/PayEntryInput.jsp?	5.画面イメージ (2)モバイル
18	ヘルプ A：利用可能 クレジットカード	https://XXXXXXXX/PayEntryConfirm.jsp	5.画面イメージ (2)モバイル
19	ヘルプ B： セキュリティコード	https://XXXXXXXX/PayEntryConfirm.jsp	5.画面イメージ (2)モバイル
20	ヘルプ C： カード支払い方法	https://XXXXXXXX/PayEntryConfirm.jsp	5.画面イメージ (2)モバイル

※1　URL 及び URL のパラメータについては検査時点において有効だったものを記載しております。
　　また、POST パラメータについては割愛させて頂いております。

※2　「仕様書.pdf」において、診断対象画面が記載されている章番号になります。

△△△△△サイト　(10 画面)

	画面名称	URL※3	備考 ※4
画面イメージ(1)PC			
21	契約者ログイン②	https://YYYYYYYY/PayEntry.jsp https://YYYYYYYY/PayEntry.jsp	5.画面イメージ (1)PC-①
22	ご契約内容確認③	https://YYYYYYYY/PayEntryDetails.jsp	5.画面イメージ (1)PC-①
23	注意事項④	https://YYYYYYYY/PayEntryDetails2.jsp https://YYYYYYYY/PayEntryKiyaku_inc.jsp	5.画面イメージ (1)PC-①
24	カード情報/ メールアドレス入力⑤	https://YYYYYYYY/PayEntryInput.jsp	5.画面イメージ (1)PC-②
25	ヘルプ： ご利用可能カード⑥	https://YYYYYYYY/PayEntryInputHelp.jsp	5.画面イメージ (1)PC-②
26	ヘルプ： セキュリティコード⑦	https://YYYYYYYY/PayEntryInputHelp2.jsp	5.画面イメージ (1)PC-②

	画面名称	URL※3	備考 ※4
27	入力内容確認⑧	https://YYYYYYYY/PayEntryConfirm.jsp	5.画面イメージ (1)PC-③
28	登録結果確認⑨	https://YYYYYYYY/PayEntryEnd.jsp	5.画面イメージ (1)PC-③
29	契約者ログアウト⑩	https://YYYYYYYY/Logout.jsp	5.画面イメージ (1)PC-③
30	カード情報/ メールアドレス入力⑤ （「カード情報修正」 ボタン）	https://YYYYYYYY/PayEntryInput.jsp	5.画面イメージ (1)PC-②

※3　URL 及び URL のパラメータについては検査時点において有効だったものを記載しております。
　　また、POST パラメータについては割愛させて頂いております。

※4　「仕様書.pdf」において、診断対象画面が記載されている章番号になります。

2　診断結果

2－1　総合評価

今回実施した診断の診断結果に基づき、弊社の評価基準に照合した総合評価を脆弱性診断に対して行いました。以下に評価クラスと評価の根拠となった診断結果を示します。

〇〇〇〇〇サイト

B	サーバ設定の不備に起因する軽微な脆弱性のみが確認されていますが、診断対象 Web アプリケーションにおいては脆弱性が確認されていないことから、セキュリティ上問題の少ない状態です。

- 診断対象 Web アプリケーションにおいては脆弱性が確認されていません
- システム情報漏洩やなりすまし等に繋がる可能性がある、サーバの設定不備に起因する軽微な脆弱性が確認されています

△△△△△サイト

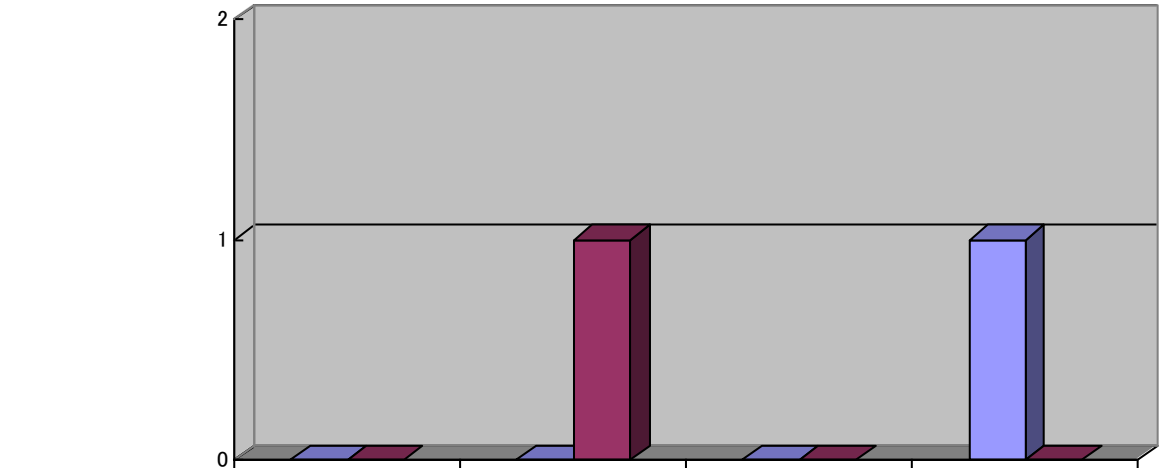
C	危険性の高い脆弱性が確認されており、セキュリティ上問題のある状態です。

- フィッシング詐欺やユーザのなりすまし等に悪用される可能性のある「クロスサイトスクリプティング」が確認されております

評価基準につきましては、付録の「6－3　評価基準」として添付しておりますので、必要に応じてご参照下さい。

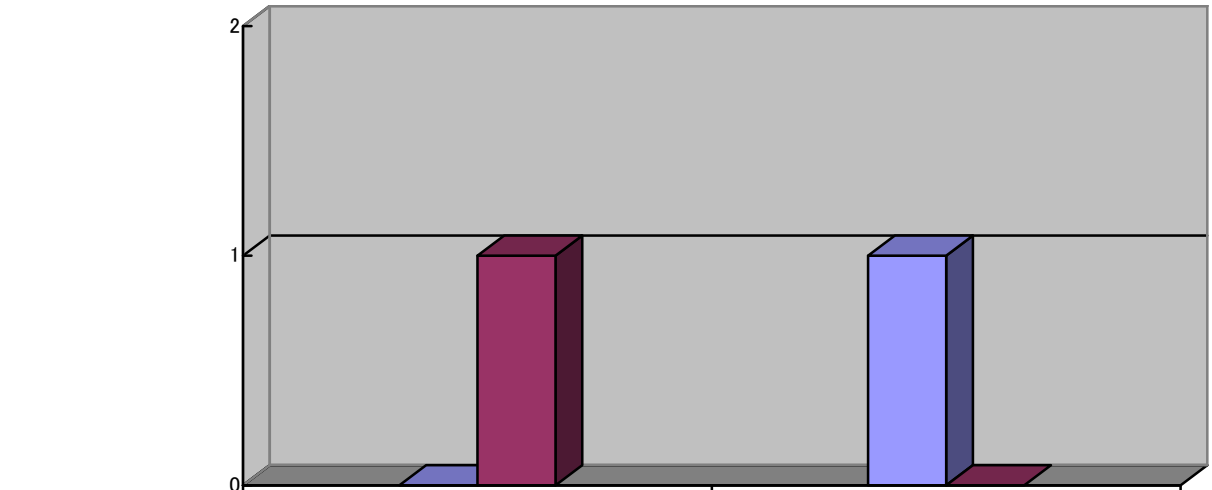
2－2　概要

本診断の診断対象範囲において検出された脆弱性を危険度別に集計したものを図 2-2-1　危険度別脆弱性検出数に、診断項目別に集計したものを図 2-2-2　診断項目別脆弱性検出数、表 2-2-1　診断項目別脆弱性検出数一覧に示します。



	Critical	High	Medium	Low
○○○○○サイト	0	0	0	1
△△△△△サイト	0	1	0	0

図 2-2-1　危険度別脆弱性検出数



	クロスサイトスクリプティング	サードパーティ製品の設定ミス
○○○○○サイト	0	1
△△△△△サイト	1	0

図 2-2-2　診断項目別脆弱性検出数

表 2-2-1　診断項目別脆弱性検出数一覧

診断項目	危険度	○○○○○サイト	△△△△△サイト	合計
クロスサイトスクリプティング	High	-	1	1
ステルスコマンド	-	-	-	-
SQL インジェクション	-	-	-	-
バッファオーバーフロー	-	-	-	-
既知の脆弱性	-	-	-	-
強制ブラウジング	-	-	-	-
Hidden フィールドの操作	-	-	-	-
サードパーティ製品の設定ミス	Low	1	-	1
バックアップファイルの検出	-	-	-	-
バックドア、デバッグオプション	-	-	-	-
HTML 中のコメント	-	-	-	-
ディレクトリトラバーサル	-	-	-	-
不適切なエラーハンドリング	-	-	-	-
パラメータの改竄	-	-	-	-
Web サービスの脆弱性	-	-	-	-
クロスサイトリクエストフォージェリ	-	-	-	-
セッション管理の脆弱性	-	-	-	-
合　計	Critical	0	0	0
	High	0	1	1
	Medium	0	0	0
	Low	1	0	1

「○○○○○サイト」については、診断対象 Web アプリケーションにおいては脆弱性が確認されておらず、サーバ設定に起因する軽微な脆弱性のみが確認されていることから、セキュリティ上問題の少ない状態です。
サーバの設定変更による対策が可能であることから、可能であれば対策の実施を推奨します。

「△△△△△サイト」については、診断対象 Web アプリケーションに危険度 High の脆弱性として「クロスサイトスクリプティング」が確認されております。
「クロスサイトスクリプティング」については、フィッシング詐欺等に悪用される可能性のある脆弱性になりますので、早急に対策を実施することを推奨します。

危険度につきましては、付録の「6－2　危険度の判定基準」として添付しておりますので、必要に応じてご参照下さい。

3　診断結果詳細

本章では、検出された脆弱性について解説を行います。

脆弱性の確認では Web ブラウザとして Google Chrome 99.0.4844.82 を使用しているため、それ以外のアプリケーションでは再現しない場合があります。
また、本章に記載されている URL や HTML、検出根拠の内容は診断時において有効だったものを記載しており、アクセスの度に変化するパラメータ等により、そのままの形では再現できない場合がございますので、あらかじめご了承下さい。

3−1　クロスサイトスクリプティング

対象サイト	△△△△△サイト
診断項目	クロスサイトスクリプティング
危険度	High

対象サイトにおいて、クロスサイトスクリプティングを確認しております。

クロスサイトスクリプティングとは、入力フォームや URL に含まれるパラメータ等からユーザに設定された文字列について、十分なチェックを行わずに HTML 中に埋め込んで表示するために引き起こされる、Web アプリケーション脆弱性の代表的なものです。細工された文字列を入力フォーム等のパラメータに与えることで、JavaScript を含む任意の HTML 要素を埋め込み、ユーザの意図しない不正なプログラムのダウンロード、ページの偽装やセッション ID の不正取得等を行われる可能性があります。

図 3-1-1～図 3-1-4 は、対象サイトにおいて確認されたクロスサイトスクリプティングを検証したものです。

△△△△△サイトにおけるのお問い合わせページ（図 3-1-1）を表示するリクエスト内容（図 3-1-2）の URL に対して、JavaScript を含む細工された文字列を追加（図 3-1-3）してリクエストを送信すると、追加した JavaScript が実行（図 3-1-4）されることを確認しています。

このことから、任意の JavaScript が埋め込み可能な状態となっており、クロスサイトスクリプティングが存在すると確認できます。

図 3-1-1　△△△△△サイトお問い合わせページ

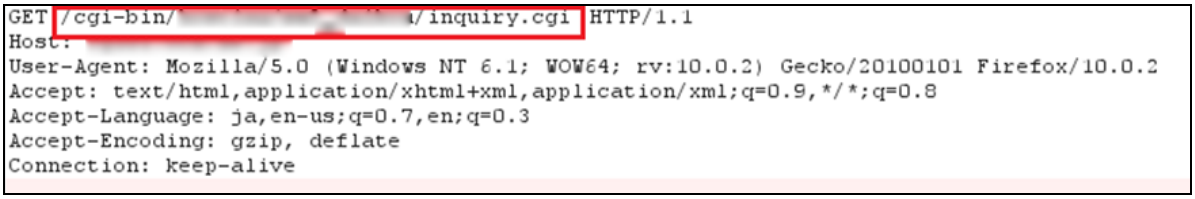


図 3-1-2　図 3-1-1 にアクセスする際のリクエスト内容

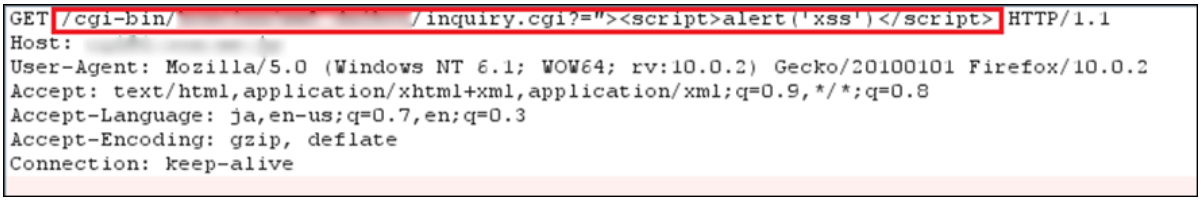


図 3-1-3　図 3-1-2 のリクエスト内容の URL に JavaScript を含む文字列を追加

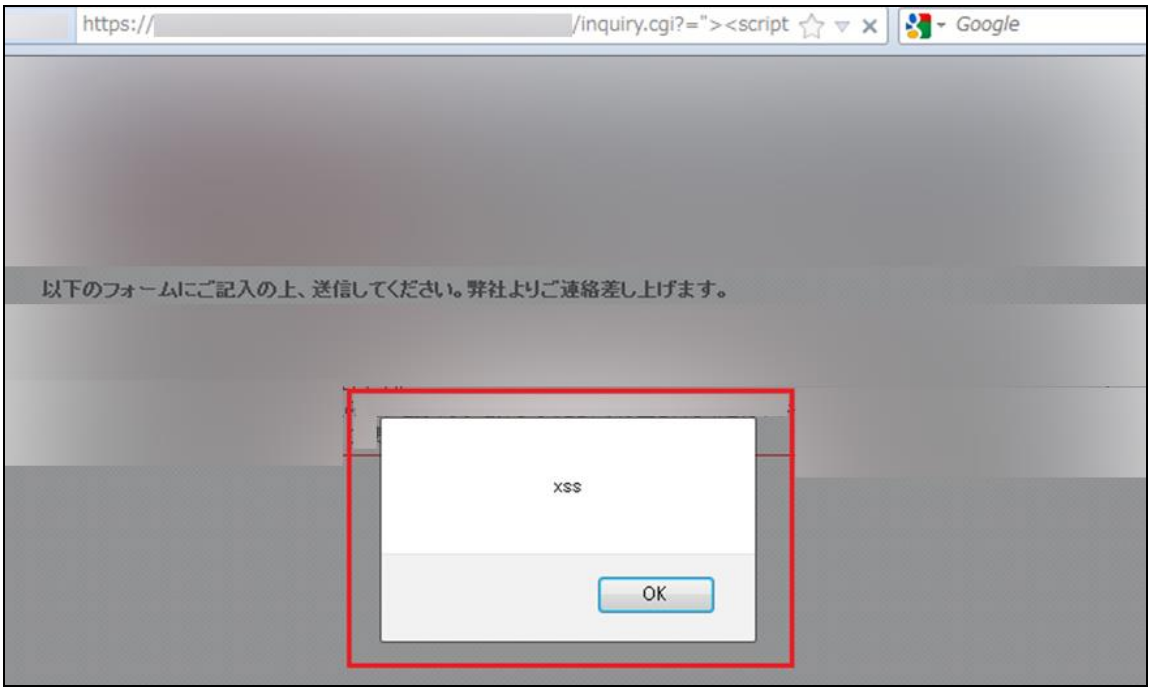


図 3-1-4　図 3-1-3 の内容でリクエストを実行した結果（JavaScript 実行）

クロスサイトスクリプティングは Web サーバや Web アプリケーションに対する直接的な不正侵入やサービス妨害に繋がることはありませんが、ユーザが通常にページアクセスを行うだけで第三者サイトに設置されたウイルスやマルウェア等のダウンロードが行われることに悪用されたり、ページ偽装を行うことによりユーザの認証情報や個人情報を不正取得するために悪用されたり、ログイン中のセッションを奪うために cookie 等を不正取得するために悪用されたりする深刻な脆弱性となることがあります。

対象サイトは個人情報を保持する Web サイトであり、本脆弱性はフィッシング詐欺に悪用される可能性がありますので、対策を実施することを推奨します。

クロスサイトスクリプティングへの対策としては、一般的には以下の２つの方法が考えられます。

1. 問題を起こす可能性のある文字を影響が出ないように変換してから出力（無毒化、サニタイジング）

パラメータ等、ユーザから入力されたデータや Web ブラウザから Web サーバに送信されたデータの内容を HTML 中に埋め込む際に、以下のように文字単位で変換処理を行うことで、HTML において特別な意味を持つ文字を単なる文字として Web ブラウザに解釈させることが可能です。
この変換処理を一般的には無毒化・サニタイジングと呼びます。また、変換後の文字列は実体参照と呼ばれます。

変換前	変換後
<	<
>	>
&	&
″	"
’	'

クロスサイトスクリプティングは<>のように HTML として特別な意味を持つ文字を埋め込むことにより引き起こされますので、無毒化を行った上で出力することでこの脆弱性の影響を受けないようにすることが可能です。

なお、タグ属性の値は必ずダブルクォート記号（″）、もしくはシングルクォート記号（’）で囲うようにしてください（例：<input type="text" value="1">）。囲っていない場合は、無毒化を行った場合でもクロスサイトスクリプティングの影響を受ける可能性があります。ダブルクォート記号とシングルクォート記号が混在している場合も問題が生じる場合がありますので、サイト全体で統一することを推奨します。

上記は一般的な対策ですが、値が埋め込まれている箇所によっては前述の文字単位の変換では不十分な場合もありますので、実際に対策を行う際には独立行政法人 情報処理推進機構（IPA）が公開している資料についても参照することを推奨します。

セキュアプログラミング講座 Web アプリケーション編
第 7 章 エコーバック対策 スクリプト注入
http://www.ipa.go.jp/security/awareness/vendor/programmingv2/contents/601.html

安全なウェブサイトの作り方
http://www.ipa.go.jp/security/vuln/websecurity.html

2. Web アプリケーションファイアウォール（WAF）でのアクセス拒否

Web アプリケーションファイアウォールを導入することで、Web アプリケーションの修正を行わずに不正なアクセスを拒否することが可能です。
Web アプリケーションファイアウォールは全ての Web アプリケーションの脆弱性に対応出来るものではありませんが、一般的なクロスサイトスクリプティングや SQL インジェクション、コマンドインジェクション等の脆弱性については大部分の不正アクセスを拒否することが可能です。
すぐに Web アプリケーションの修正が出来ないようなケースでは有効な対策方法となります。

安全なウェブサイトの作り方
http://www.ipa.go.jp/security/vuln/websecurity.html

Web アプリケーションファイアウォールの必要性（@IT）
http://www.atmarkit.co.jp/fsecurity/rensai/waf01/waf01.html

Web Application Firewall 読本
http://www.ipa.go.jp/security/vuln/documents/waf.pdf

前者は Web アプリケーションの修正、後者はシステムの導入となりますが、可能であれば根本的な対策である前者の対策を行うことが推奨されます。

3−2　アプリケーション標準エラーページの使用

対象サイト	○○○○○サイト
診断項目	サードパーティ製品の設定ミス
危険度	Low

対象サイトにおいて、アプリケーションサーバの標準エラーページが使用されていることを確認しております。

アプリケーションの標準エラーページにはシステム情報が含まれる場合があり、その内容からバージョン等を推測することで、公開されているセキュリティホール情報よりインストールされているアプリケーションに存在する脆弱性を探すために利用される可能性があります。
また、使用しているアプリケーションのバージョンによっては、標準エラーページに脆弱性が存在している場合があります。

図 3-2-1 は、対象サイトにおいて確認された「アプリケーション標準エラーページの使用」についての結果です。

「○○○○○サイト」において、BASIC 認証によるログインが必要となる URL（https://XXXXXXXX/aaaa/%5C../manager/html）にアクセスを行って認証に失敗した結果、Web サーバからの応答としてエラー（401 Unauthorized）が返り、アプリケーションサーバ Apache Tomcat の標準エラーページが出力（図 3-2-1）されることを確認できます。

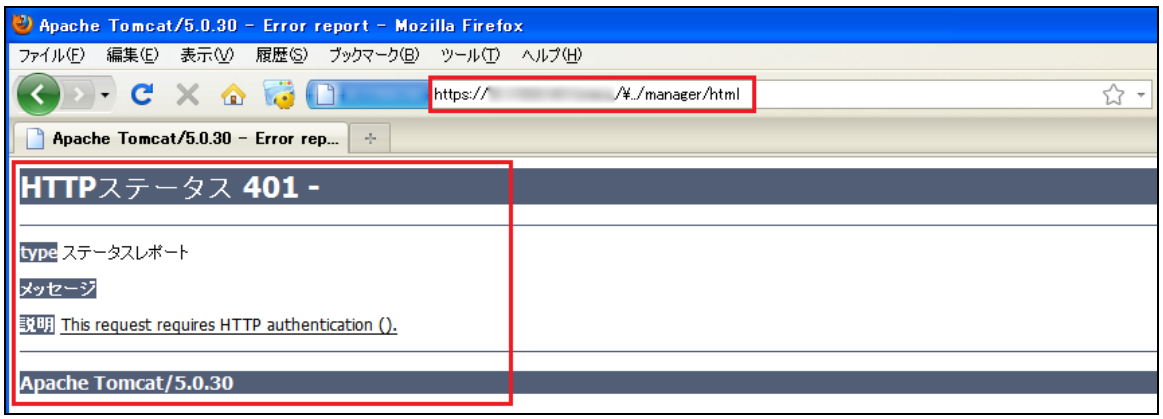


図 3-2-1　BASIC 認証に失敗した結果（Apache Tomcat の標準エラーページが出力）

Apache や Apache Tomcat の標準エラーページは、静的なページではなくエラーの箇所や原因等を埋め込んで出力される動的ページとなっているため、旧バージョンにおいては脆弱性が確認されています。

一般に公開される Web サイトであること、および将来的に標準エラーページに存在する新たな脆弱性が公開される可能性があることを考慮して、アプリケーションの標準エラーページについては使用せず、カスタムエラーページを作成して使用することを推奨します。

アプリケーションサーバ Apache Tomcat でカスタムエラーページを使用するには、設定ファイル web.xml において<error-page>要素の定義を行うことにより可能です。

図 3-2-2 に示す設定例では、エラーコード 500（Internal Server Error）のみを定義していますが、実際に設定を行う場合は、起こりうる全てのエラーコードに対してカスタムエラーページの定義を行うようご注意下さい。

```
...省略...
<error-page>
    <error-code>500</error-code>
    <location>/error/500.html</location>
</error-page>
...省略...
```

図 3-2-2　<error-page>要素によるカスタムエラーページ設定例

詳細については、以下のページをご参照下さい。

Tomcat 質問集（カスタマイズしたエラーページを返すには、どうしたらいいの？）
http://www.jajakarta.org/tomcat/faq/misc.html#error

4　総括

本診断の結果、「△△△△△サイト」において危険度の高い脆弱性である「クロスサイトスクリプティング」が検出されており、セキュリティ上問題のある状態と言えます。
危険性の高い脆弱性については対策の確実な実施、現時点で危険性の低い脆弱性についても、今後のサイト更改等による影響も考慮し、可能であれば対策を実施することを推奨します。

今後の Web アプリケーションセキュリティへの対策としては、脆弱性が発生することを予防することも念頭に置き、以下のような対応の検討をお奨めします。

✓　リリース前の脆弱性診断の徹底
リリース前の脆弱性診断を徹底することで、脆弱性が存在する Web アプリケーションを公開するリスクを減らすことが可能です。

✓　開発当初からセキュリティを考慮した Web アプリケーションの実装の実施
開発時からセキュリティを考慮した実装を行うことにより、よりセキュリティ的に安全な Web アプリケーションを構築することが可能となります。通常の開発では開発者・開発会社によってセキュリティレベルが大きく変化することが予想されますが、開発指針・チェックリストを作成しておくことで最低限のセキュリティレベルを維持することが可能になると考えます。

✓　Web アプリケーションファイアウォールの導入による防衛
リリース前の脆弱性診断が難しいケースや、頻繁に Web サイトの更新を行うようなサイトでは、Web アプリケーションファイアウォールの導入が効果的です。
クロスサイトスクリプティングや SQL インジェクション等多くの Web アプリケーション脆弱性に対して防衛することが可能です。

以上で本報告を総括させて頂きますが、本診断報告書について指摘された脆弱性を修正するのみではなく、今後のセキュリティ対策に活用して頂ければ幸いです。

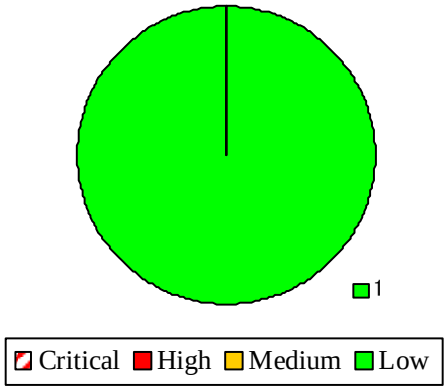
5　脆弱性詳細

本章では、本診断で検出された脆弱性の詳細を示します。

5−1　〇〇〇〇〇サイト

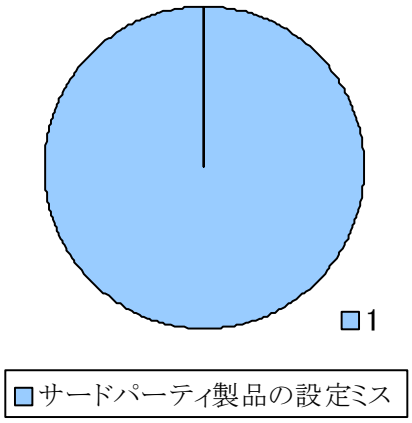
脆弱性危険度別検出数一覧／分布

危険度	検出数
Critical	0
High	0
Medium	0
Low	1
合計	1



診断項目別検出数一覧／分布

診断項目	検出数
クロスサイトスクリプティング	0
ステルスコマンド	0
SQL インジェクション	0
バッファオーバーフロー	0
既知の脆弱性	0
強制ブラウジング	0
Hidden フィールドの操作	0
サードパーティ製品の設定ミス	1
バックアップファイルの検出	0
バックドア、デバッグオプション	0
HTML 中のコメント	0
ディレクトリトラバーサル	0
不適切なエラーハンドリング	0
パラメータの改竄	0
Web サービスの脆弱性	0
クロスサイトリクエストフォージェリ	0
セッション管理の脆弱性	0
合計	1



アプリケーション標準エラーページの使用

診断項目：サードパーティ製品の設定ミス

概要

対象サイトにおいて、アプリケーションサーバ Apache Tomcat の標準エラーページが使用されていることを確認しております。

アプリケーションの標準エラーページにはシステム情報が含まれる場合があり、その内容からバージョン等を推測することで、公開されているセキュリティホール情報よりインストールされているアプリケーションに存在する脆弱性を探すために利用される可能性があります。

また、使用しているアプリケーションのバージョンによっては、標準エラーページに脆弱性が存在している場合があります。

対策

アプリケーションの標準エラーページについては使用せず、カスタムエラーページを作成して使用することを推奨します。

アプリケーションサーバ Apache Tomcat でカスタムエラーページを使用するには、カスタムエラーページを使用するには、設定ファイル web.xml において<error-page>要素の定義を行うことにより可能です。

以下の設定例では、エラーコード 500(Internal Server Error)のみを定義していますが、実際に設定を行う場合は、起こりうる全てのエラーコードに対してカスタムエラーページの定義を行うようご注意下さい。

```
<error-page>
   <error-code>500</error-code>
   <location>/error/500.html</location>
</error-page>
```

詳細については、以下のページをご参照下さい。

Tomcat 質問集（カスタマイズしたエラーページを返すには、どうしたらいいの？）

http://www.jajakarta.org/tomcat/faq/misc.html#error

対象

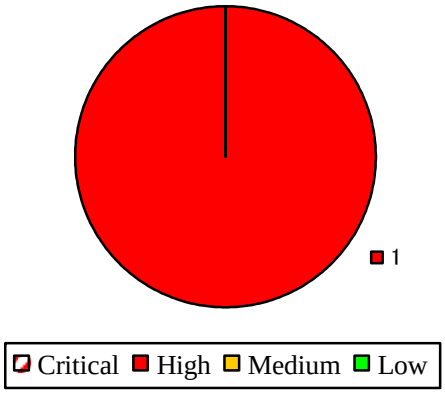
対象サイト／画面名称
○○○○○サイト／—
対象 URL
https://XXXXXXXX/aaaa/%5C../manager/html

対象パラメータ
なし
危険度
Low
検出根拠
対象 URL にアクセスを行い BASIC 認証に失敗した結果、Web サーバからの応答としてエラー（401 Unauthorized）が返り、エラーページとしてアプリケーションサーバ Apache Tomcat の標準エラーページが出力されることを確認。
備考
なし

5−2　△△△△△サイト

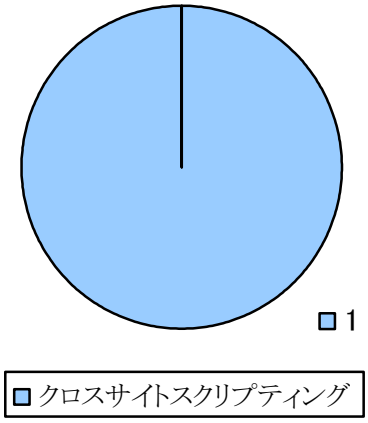
脆弱性危険度別検出数一覧／分布

危険度	検出数
Critical	0
High	1
Medium	0
Low	0
合計	1



診断項目別検出数一覧／分布

診断項目	検出数
クロスサイトスクリプティング	1
ステルスコマンド	0
SQL インジェクション	0
バッファオーバーフロー	0
既知の脆弱性	0
強制ブラウジング	0
Hidden フィールドの操作	0
サードパーティ製品の設定ミス	0
バックアップファイルの検出	0
バックドア、デバッグオプション	0
HTML 中のコメント	0
ディレクトリトラバーサル	0
不適切なエラーハンドリング	0
パラメータの改竄	0
Web サービスの脆弱性	0
クロスサイトリクエストフォージェリ	0
セッション管理の脆弱性	0
合計	1



クロスサイトスクリプティング

診断項目：クロスサイトスクリプティング

概要

対象サイトにおいて、クロスサイトスクリプティングを確認しております。

クロスサイトスクリプティングとは、入力フォームや URL に含まれるパラメータ等からユーザに設定された文字列について、十分なチェックを行わずに HTML 中に埋め込んで表示するために引き起こされる、Web アプリケーション脆弱性の代表的なものです。細工された文字列を入力フォーム等のパラメータに与えることで、JavaScript を含む任意の HTML 要素を埋め込み、ユーザの意図しない不正なプログラムのダウンロード、ページの偽装やセッション ID の不正取得等を行われる可能性があります。

対策

クロスサイトスクリプティングは Web サーバや Web アプリケーションに対する直接的な不正侵入やサービス妨害に繋がることはありませんが、

- ウイルスやマルウェアの意図しないダウンロード・インストール
- ページ偽装によるユーザの認証情報や個人情報の漏洩
- ログイン中のセッション ID が漏洩

のような被害が発生する可能性のある危険性の高い脆弱性となりますので、早急に対策を実施することを推奨します。

クロスサイトスクリプティングへの対策としては、一般的には以下の２つの方法が考えられます。

1. 問題を起こす可能性のある文字を影響が出ないように変換してから出力（無毒化、サニタイジング）

   パラメータ等、ユーザから入力されたデータや Web ブラウザから Web サーバに送信されたデータの内容を HTML 中に埋め込む際に、以下のように文字単位で変換処理を行うことで、HTML において特別な意味を持つ文字を単なる文字として Web ブラウザに解釈させることが可能です。
   この変換処理を一般的には無毒化・サニタイジングと呼びます。また、変換後の文字列は実体参照と呼ばれます。

変換前	変換後
<	<
>	>
&	&
″	"
’	'

クロスサイトスクリプティングは<>のようにHTMLとして特別な意味を持つ文字を埋め込むことにより引き起こされますので、無毒化を行った上で出力することでこの脆弱性の影響を受けないようにすることが可能です。

なお、タグ属性の値は必ずダブルクォート記号（″）、もしくはシングルクォート記号（’）で囲うようにしてください（例：<input type="text" value="1">）。囲っていない場合は、無毒化を行った場合でもクロスサイトスクリプティングの影響を受ける可能性があります。ダブルクォート記号とシングルクォート記号が混在している場合も問題が生じる場合がありますので、サイト全体で統一することを推奨します。

上記は一般的な対策ですが、値が埋め込まれている箇所によっては前述の文字単位の変換では不十分な場合もありますので、実際に対策を行う際には独立行政法人 情報処理推進機構（IPA）が公開している資料についても参照することを推奨します。

セキュアプログラミング講座 Webアプリケーション編
第7章 エコーバック対策 スクリプト注入
http://www.ipa.go.jp/security/awareness/vendor/programmingv2/contents/601.html

安全なウェブサイトの作り方
http://www.ipa.go.jp/security/vuln/websecurity.html

2. Webアプリケーションファイアウォール（WAF）でのアクセス拒否

Webアプリケーションファイアウォールを導入することで、Webアプリケーションの修正を行わずに不正なアクセスを拒否することが可能です。
Webアプリケーションファイアウォールは全てのWebアプリケーションの脆弱性に対応出来るものではありませんが、一般的なクロスサイトスクリプティングやSQLインジェクション、コマンドインジェクション等の脆弱性については大部分の不正アクセスを拒否することが可能です。

すぐに Web アプリケーションの修正が出来ないようなケースでは有効な対策方法となります。

安全なウェブサイトの作り方
http://www.ipa.go.jp/security/vuln/websecurity.html

Web アプリケーションファイアウォールの必要性（@IT）
http://www.atmarkit.co.jp/fsecurity/rensai/waf01/waf01.html

Web Application Firewall 読本
http://www.ipa.go.jp/security/vuln/documents/waf.pdf

前者は Web アプリケーションの修正、後者はシステムの導入となりますが、可能であれば根本的な対策である前者の対策を行うことを推奨します。

対象

対象サイト/画面名称
△△△△△サイト／□□□□□□：確認
対象 URL
https://YYYYYYYY/cgi-bin/xxxxxx/xxxxxx/inquiry.cgi

対象パラメータ
—（URL）
危険度
High
検出根拠
以下の URL にアクセスした結果、URL に設定された JavaScript が実行されることを確認しました。 https://YYYYYYYY/cgi-bin/xxxxxx/xxxxxx/inquiry.cgi?="><script>alert('xss')</script>
備考
なし

6　付録

6－1　診断項目一覧

本診断において行った診断項目を以下に示します。

クロスサイトスクリプティング

クロスサイトスクリプティングとは Web アプリケーションソフトウェアの脆弱性で、「サイトを跨ってスクリプトを実行する」という意味です。

Web アプリケーションで、入力されたデータの内容を充分チェックせずに HTML 内に出力していると、HTML 内に JavaScript などの任意のコードを埋め込むことができてしまいます。このような状態を「クロスサイトスクリプティング脆弱性がある」と言います。

例として、任意のタグがそのまま書き込めてしまう掲示板が挙げられます。悪意あるユーザが「<script>」などの HTML タグを含む内容を投稿すると、投稿内容を閲覧したときにスクリプトが実行されてしまう危険性があります。スクリプトの内容によっては cookie データの盗聴や改竄などが可能なため、商取引に使った cookie を横取りして、本人になりすまして物品の購入を行なったり、cookie を認証やセッション管理に使っているサイトに侵入したりするなど、より広範かつ深刻な損害を与える可能性があります。

ステルスコマンド

外部から任意の OS のコマンドや SSI(サーバサイドインクルード)などを実行することが可能な状態であることです。ユーザの入力がそのままシェルや SSI にコマンドとして渡せるようになっているとこのような事態が発生します。

SQL インジェクション

「インジェクション(injection)」とは「注入」という意味で、SQL データベースに対し、外部から任意の SQL を実行することができる状態を指します。任意のデータを抽出できてしまうことが問題となります。

例として、あるユーザが他のユーザのデータを見たり、パスワード情報を得たりできてしまう可能性があります。また、SQL の種類や設定によってはデータベースの改竄や削除ができてしまったり、さらにはサーバ内で任意のコマンドを実行することができてしまったりする危険性があります。

バッファオーバーフロー

想定よりも長いデータを処理しきれない場合に発生します。バッファが溢れる(オーバーフローする)ことを意味します。

本来書き込まれるべきメモリ領域からデータが溢れ、本来書き込まれてはならない別の領域に書き込まれてしまいます。その結果として何が起きるのかは様々ですが、任意のコードを実行されてしまうこともあり、致命的なセキュリティホールになる危険性があります。

既知の脆弱性

OS や Web サーバ、アプリケーションサーバ、サードパーティ製ツールなどの持つ一般的に広く知られている脆弱性のことです。

攻撃者はこれらの脆弱性を悪用することによって、アクセス権限の不正取得、機密情報の奪取、データ破壊等が行えるようになります。これらの問題の多くは主にベンダからのパッチプログラムの適用により解消できます。

強制ブラウジング

意図していないコンテンツが公開ディレクトリ上にあるために、第三者が URL を直接入力することでそれらのページやデータを取得できてしまうことです。これらは攻撃者に攻略の糸口となるヒントを与えたり、機密情報の漏洩をもたらされたりします。

例えば、アプリケーションのソースコードが公開ディレクトリにそのまま置かれている場合や、CSV などでまとめた顧客情報が漏洩してしまう場合などが考えられます。

hidden フィールドの操作

HTML の入力フォームの一つである hidden フィールドは、フォームの値を画面上には表示させずにアプリケーションに渡すことができます。これはページ間でのデータの受渡しによく使用されています。しかし、hidden フィールドに指定した値はクライアント側で容易に変更できてしまうため、値の信頼性はありません。

この hidden フィールドによって重要なデータをやり取りしている場合、アプリケーションによっては、アクセスコントロールを迂回されたり、予期せぬ動作を引き起こしたりします。

例えば、商取引サイトにおいて hidden フィールドに商品の価格を設定している場合には、hidden フィールドの改竄により商品の価格を不正に操作されてしまいます。

サードパーティ製品の設定ミス

サードパーティ製品の設定にミスがある状態です。主に人為的なミスが考えられますが、製品によっては初期状態からセキュリティ上、問題のある設定になっているものも見受けられます。
これらの情報は攻撃者に攻略のヒントを与えてしまうため、攻撃が成功する可能性を高くしてしまいます。

バックアップファイルの検出

バックアップファイルと思われるものがサーバ上に存在する状態です。インタープリタ言語などによる動的なページを生成しているサイトにおいて、ファイルを変更した際にバックアップを設定した以外の拡張子のファイル名にした場合、処理が実行されずにソースコードが表示されてしまう可能性があります。

人為的にバックアップファイルを保存している場合や、エディタにより自動的にバックアップファイルが残っている場合に問題となります。

バックドア、デバッグオプション

アプリケーションの開発段階で使用されていたデバッグ用のオプションやバックドアがそのまま残されている状態です。これらを攻撃者に不正に利用されてしまう可能性があります。

HTML 中のコメント

HTML の中に重要な情報がコメントとして書かれている状態です。例えば、管理者のユーザ ID やパスワードの一部などが書かれている場合、それだけで不正にアクセスされてしまう可能性があります。

ディレクトリトラバーサル

Web サーバやアプリケーションサーバが通常表示させることの可能なルートディレクトリを越えて、ディレクトリをさかのぼることが出来てしまう状態のことを言います。システム構成が知られている場合には、パスワードなどの機密ファイルが漏洩したり、任意のコマンドを指定し実行される可能性があります。

典型的なパターンとしては、URL に「../」を多量に使用することで Unix 系のパスワードファイルが格納されている「/etc/passwd」ファイルを取得しようとすることが挙げられます。

不適切なエラーハンドリング

Web アプリケーションのエラー処理を行う際の画面表示内容が適切でない状態です。システムの情報を表示することは開発者にとって有用ですが、攻撃者にとっても有用であり攻略のヒントを与えてしまうため、攻撃が成功する可能性を高くしてしまいます。

例えば、SQL の実行エラーが表示されてしまっている場合には、攻撃者はその情報を見て任意の SQL を実行しようとします。

パラメータの改竄

Web アプリケーションが通常使用しているパラメータの値を不正な値に変更したり削ったりすることで、情報の漏洩やアクセスコントロールを迂回することが可能な状態です。
不正なメタ文字や、制御文字などをパラメータに入力することで予期せぬ動作を引き起こします。

Web サービスの脆弱性

Web サービスに特化した脆弱性です。Web サービスは一般の Web アプリケーションに存在する SQL インジェクションやクロスサイトスクリプティングなどの脆弱性の他にも XML 攻撃などの特殊なものがあります。

クロスサイトリクエストフォージェリ

クロスサイトリクエストフォージェリとは、記事の投稿や商品の購入等、永続的な影響を与えるリクエストが発行されるページに正規のユーザを誘い込むことで、正規ユーザに意図しない操作を実行させることが可能な脆弱性です。

確認ページの無い記事投稿ページや、本来受け付けるべきではない外部のリンクやフォームから発行されたリクエストをそのまま処理してしまうような Web サイト等でよく見られる脆弱性で、意図しない商品の購入や記事の投稿等の被害をユーザに与える危険性があります。

セッション管理の脆弱性

セッション管理とは、あるアクセスが特定のユーザからのものであることを識別管理することを意味します。その識別情報をセッション追跡パラメータといいます。一般的には cookie によるセッション管理がよく行われています。セッション追跡パラメータはユーザを識別するための重要な情報であり、漏洩した場合にはなりすましの被害に遭遇する可能性があります。

例えば、セキュアでない cookie を使用したり、URL をパラメータにしたりしている場合には、セッション追跡パラメータが暗号化されずにネットワーク上を流れるため、盗聴によって内容が漏洩する可能性があります。

また、ユーザ毎に識別が行われていなかったり、アクセスコントロールが正常でないページが存在したりすることもあるため、正しくセッション管理を行う必要があります。

6－2　危険度の判定基準

本診断結果報告書では検出された各脆弱性について、表 6-2-1 を基に危険度を判定し記載しています。

危険度は、検出された各脆弱性への対策の際に、どの脆弱性を優先的に修正すべきか判断するための目安として記載しているものです。

表 6-2-1　危険度の判定基準

危険度	判定基準
Critical	直接的に深刻な被害を及ぼすことが懸念される脆弱性。 SQL インジェクション等が該当します。
High	フィッシング詐欺等、受動的な攻撃により個人情報等の重要な情報を奪われるような被害が想定される重大な脆弱性。 Critical との違いは、Critical は攻撃者が能動的に攻撃を行うことが可能な脆弱性を対象としているのに対し、High はユーザが罠にかかるのを待つような受動的な手法が採られる脆弱性を対象としています。 クロスサイトスクリプティング等が該当します。
Medium	他の脆弱性と組み合わせることによって被害を受けることが想定される脆弱性。ディレクトリリスティングなどが該当します。
Low	Medium 以上に該当せず、被害を受ける可能性が低いと考えられる脆弱性。サードパーティ製品の設定ミスなどが該当します。

判定基準はあくまでも目安であり、脆弱性の検出された箇所・内容等により判定基準とは異なる危険度を脆弱性に与えることもありますので、ご了承下さい。

6－3　評価基準

本診断結果報告書における総合評価は、表 6-3-1 に規定される絶対評価と、診断対象の環境を考慮して評価される相対評価によるものです。

絶対評価は、A、B、C、D のいずれかのアルファベット 1 文字で表記され、診断結果を絶対評価の評価基準に照合し適合するクラスが評価として与えられます。

表 6-3-1　絶対評価の評価基準

クラス	評価基準
A	脆弱性が検出されていない。
B	システム情報の漏洩を始めとした、単体では被害を受ける可能性が低いと考えられる脆弱性のみ検出されている。
C	危険性の高い脆弱性が検出されており、被害を受ける可能性がある。
D	個人情報の漏洩に繋がる深刻な脆弱性が検出されている。または、検出されている複数の脆弱性を組み合わせることで個人情報の漏洩に繋がることが懸念される状態である。

相対評価は、絶対評価では表すことが出来ない診断対象の環境やリスト対象等、外的要因について考慮されて評価されるものであり、＋（プラス；より安全）、－（マイナス；より安全でない）を絶対評価に付与することで表されます。

なお、上記評価基準は、弊社の診断実績を基に、診断結果を簡潔に表現するために作成された、弊社独自基準になります。上記評価基準による評価は、あくまでも診断結果を簡潔に表現するためのものであり、弊社は評価に対しての保証や責任は負いかねますのでご了承下さい。

以上