Confidential

Shinko Technomist

文書番号 SKS-SC-yyyymmdd-Px

株式会社 Example 御中

Web アプリケーション診断報告書

yyyy 年 mm 月 dd 日

本報告書は、yyyy 年 mm 月 dd 日から dd 日にかけて株式会社 Example「Example システム」の脆弱性を診断した結果を報告するものです。報告書の内容には、脆弱性に関する情報が含まれていますので、お取り扱いには十分にご注意ください。

株式会社 信興テクノミスト

目次

1 はじめに

1.1. 診断の目的

インターネットを介した不正アクセスの多くが、Web サイトを狙った攻撃となった今日では、不正アクセス等の攻撃に耐えられる堅牢な Web システムが求められています。

堅牢な Web システムを構築するためには、システム構成要素の脆弱性を正しく認識し対処する必要があります。OS や、Web サーバおよびデータベースなどのミドルウェアの脆弱性情報は信頼のおける公的な機関から情報提供されますが、各々が構築する Web アプリケーションの脆弱性は Web システムを運用している企業が自ら調べるほか脆弱性を正しく認識することはできません。

弊社の Web アプリケーション脆弱性診断は、堅牢な Web システムを構築するために Web アプリケーションに潜在する脆弱性を弊社技術者が診断・発見し、脆弱性の種類、発見箇所および対策指針を報告いたします。弊社の Web アプリケーション脆弱性診断により、堅牢な Web システムの構築・運用に役立てていただくことを目的としています。

1.2. Web アプリケーション診断の位置づけ

堅牢な Web システムを構築・運用するためには、OS や Web サーバおよびデータベースなどのミドルウェアの脆弱性に対しても適切に対処する必要があります。また、ファイアウォールに代表されるネットワークのセキュリティ対策も不可欠です。したがって、Web アプリケーションの脆弱性診断と診断結果による対策も非常に重要であるとともに、Web サイトの各構成要素に対しても脆弱性診断とその結果による対策を施す必要があります。

本報告書が対象としているのは、Web アプリケーションの脆弱性診断のみとなっており、OS やミドルウェアなどの脆弱性に関する情報は含まれていません。別途、これらについても専門機関を利用するなどして十分な対策を行われることを推奨します。

1.3. 本報告書の取り扱いについて

本報告書には、お客様の Web アプリケーションに関するセキュリティ上の問題点が記載されています。この情報がひとたび攻撃者に渡ってしまうと、セキュリティ上の問題点を狙った不正アクセス攻撃を受け情報漏えい等の事故が発生する可能性があります。したがって、本報告書のお取り扱いには十分に注意して頂けますようお願いいたします。

1.4. セキュリティ対策の運用について

本報告書は診断時点でのお客様のWebアプリケーションのセキュリティ上の問題点を診断した結果が記載されています。Web アプリケーションへの攻撃は日々研究され進化し続けているため、時間経過とともにセキュリティ上の問題が増加することが考えられます。

堅牢な Web システムを継続的に運用するためには、定期的に Web アプリケーションに潜んでいるセキュリティ上の問題点を正しく認識し対策を施すことをお勧めいたします。

2 診断内容

2.1. 診断環境

2.1.1. 診断日時

- yyyy 年 mm 月 dd 日 ～ dd 日　10:00 ～ 18:00
- yyyy 年 mm 月 dd 日　10:00 ～ 18:00

2.1.2. 診断元 IP アドレス

- xxx.xxx.xxx.xxx
- yyy.yyy.yyy.yyy
- zzz.zzz.zzz.zzz

2.2. 診断項目

主な診断項目を以下に列挙します。

区分	名称
認証	パスワードポリシー
	不適切な認証
	脆弱なパスワードリマインダ
承認	セッションの推測
	不適切な承認
	セッションの固定
クライアント側での攻撃	クロスサイトスクリプティング
	コンテンツの詐称
コマンドの実行	バッファオーバーフロー
	書式文字列攻撃
	LDAP インジェクション
	OS コマンドインジェクション
	SQL インジェクション
	SSI インジェクション
	XPath インジェクション
情報公開	ディレクトリインデクシング
	ソース記載による情報漏えい
	推測可能なリソース位置
ロジックを狙った攻撃	機能の悪用
	パス・トラバーサル
	リダイレクタ
	不適切なプロセスの検証

2.3. 診断対象

2.3.1. 対象サイト

- Example システム

2.3.2. 対象 URL

- http://Example.co.jp/example1.html
- http://Example.co.jp/example2.cfm
- http://Example. co.jp /submit.cfm
- https://Example. co.jp /login.cfm
- ・・・

2.3.3. Web サイトのシステム情報

お客様情報シートに記載いただいた情報と、診断にて判明した情報をもとに記載しています。

ホスト名	Example.co.jp
IP アドレス	xxx.xxx.xxx.xxx
OS	Windows Server 2012 R2
Web サーバ	Microsoft-IIS/8.5
アプリケーションサーバ	Apache Tomcat 8.0.15
データベース	PostgreSQL 9.4
開発言語	Java JDK 8u25 JSF

3 診断結果概要

3.1. 総合評価

E. 非常に危険な状態です

3.1.1. 総合評価について

総合評価では、診断結果詳細にて記載されている各脆弱性の危険度（High、Medium、Low）のレベルと危険度の個数を元に下記の表のような評価をしております。なお、危険度の詳細については、付録 A を参照してください。

評価	基準
A	脆弱性が検出されなかった
B	危険度 Low の脆弱性のみ検出
C	危険度 Medium の脆弱性を検出
D	危険度 High の脆弱性を検出
E	危険度 High の脆弱性を複数検出

3.2. 検出された脆弱性一覧

脆弱性区分	脆弱性名	危険度	個数
承認	不適切な承認 （権限のない機能の利用）	High	5
	不適切な承認 （クロスサイトリクエストフォージェリ）	Low	6
	セッションの固定化	Medium	1
クライアント側での攻撃	クロスサイトスクリプティング	Medium	3
コマンドの実行	SQL インジェクション	High	3

3.3. 総評

今回の診断の結果、お客様の Web サイトではユーザから送信されてくる各種の値に対する基本的なチェックが行われている様子が確認されました。しかし、いくつかのパラメータにおいてチェック漏れがあるため、SQL インジェクションなどの非常に危険度の高い脆弱性が存在することが確認されました。これらの脆弱性を悪用された場合、データベースに格納されている個人情報の漏えいやデータの改ざんなどの被害が発生する可能性があります。今一度 Web サイトのフローを見直されてセキュリティ対策の徹底を図っていただければと思います。

また、セッション管理に関係した部分を数点指摘しています。セッション管理のような重要な情報を扱う部分につきましては、より慎重に設計・実装を行う必要があります。やや細かい部分の指摘となっておりますが、その点を心がけて Web サイトを設計されますとお客様の Web サイトのセキュリティはなおいっそう高いものとなります。

その他にも危険度は低いのですが、何点か問題となる箇所を指摘しています。これらについても対策されることでお客様の Web サイトのセキュリティはなおいっそう高いものとなります。

なお、「5 注意事項」にて何点か確認いただきたい項目を記載しています。また、同様の問題が診断対象以外の箇所でも存在していないか、問題の検出傾向なども参考にしながら確認されるようにお願いいたします。

4 診断結果詳細

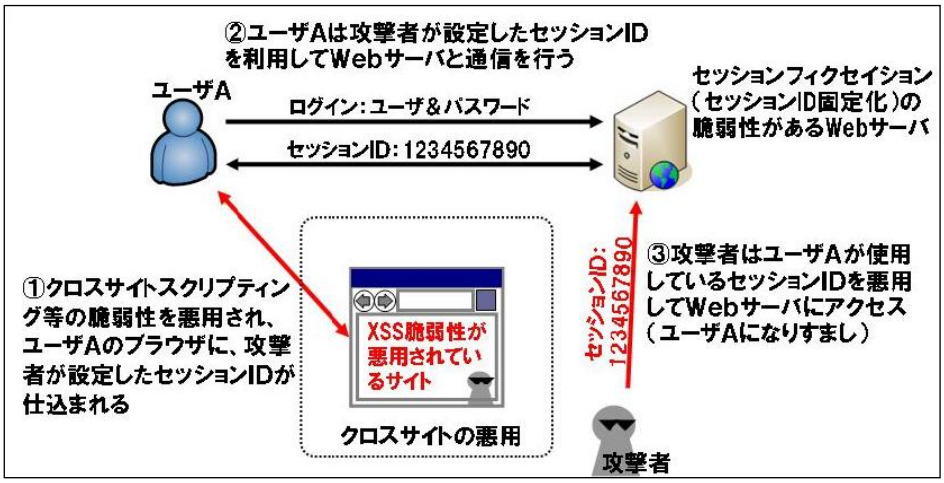
4.1. 承認

4.1.1. セッションの固定化

(1) 危険度

Medium

(2) 脆弱性概要



外部から送信されるセッション ID をそのまま使用することが可能なセッション管理の場合、悪意のある第三者がセッション ID を任意の値に変更できてしまう可能性があります。

(3) 発生箇所

No.	URL	Cookie 名
1	https://Example. co.jp /login.cfm	sid

(4) 脆弱性発生状況

今回診断した Web サイトではセッション管理に使われるセッション ID として sid が使われていました。

通常、sid などのセッション ID の値としては Web サーバから割り当てられたランダム性の高いもの（例：17d97c7ce01fdad49abbeff15b870934）を利用することになります。しかし、Web ブラウザから強制的に sid として 1 を送信してログインすることで、それ以降の処理においてこの値をセッション ID として利用可能でした。

この脆弱性を悪用することで以下のような攻撃シナリオが発生することが考えられます。

① 不正なセッション ID の登録

攻撃者がクロスサイトスクリプティングなどの脆弱性を利用して被害者の Web ブラウザにセッション ID である sid に対して 1 をセットします。

② 不正なセッション ID を利用して、Web サイトへアクセス

被害者は、セッション ID である sid に 1 をセットされていることに気付かずログインなどの操作を行います。この時点で被害者のアカウントとセッション ID の 1 が結びつきます。すなわち、sid が 1 の場合は、被害者の権限で Web サイトを利用可能となります。

③ 攻撃者による不正アクセス

攻撃者が自らの Web ブラウザの sid の値を 1 にセットして Web サイトにアクセスします。この ID は被害者のアカウントと結びついているため、結果として被害者へなりすますことが可能となります。

(5) 想定される脅威

他のユーザへのなりすましが行われます。

(6) 想定される被害

① 脆弱性悪用の実現度

被害者を悪意のある Web サイトに誘導し、クロスサイトスクリプティングなどの脆弱性を利用してセッション ID をユーザの Web ブラウザにセットする必要があります。ユーザのリテラシが高ければ誘導されることはありませんので実現度は比較的低いと考えられます。

② 脆弱性悪用により想定される被害

- なりすましによる他のユーザ権限の不正利用
- なりすましによる個人情報の漏えい
- 情報漏えい等の事故を原因とする Web サイトの停止による機会損失
- 情報漏えい等に対する損害賠償や見舞金
- 情報漏えい等の事故による企業の信用失墜

(7) 対策

認証成功後や、重要情報を入力開始する段階で推測の困難なランダムな値のセッション ID を再発行するようにしてください。今回のケースではログイン成功後にセッション ID を再発行する必要があります。

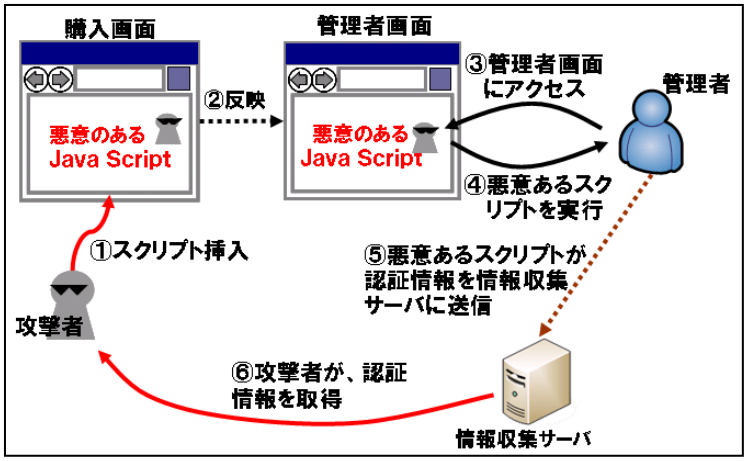
4.2. クライアント側での攻撃

4.2.1. クロスサイトスクリプティング

(1) 危険度

Medium

(2) 脆弱性概要



悪意あるスクリプトがユーザの Web ブラウザ上で実行されてしまうことで、認証情報が攻撃者に取得される可能性があります。また、Web ページの内容を改ざんし、フィッシング詐欺に悪用される可能性があります。

(3) 発生箇所

No.	URL	パラメータ名
1	http://Example.co.jp/example2.cfm	parm1
2		parm2

(4) 脆弱性発生状況

発生箇所 No.1 を例に解説します。パラメータ parm1 に対して、以下のようにスクリプトを含む不正な値を設定して送信します。

```
parm1="><script>alert(document.cookie)</script>
```

その結果、図 4-1 のようにスクリプトが実行されることが確認されました。

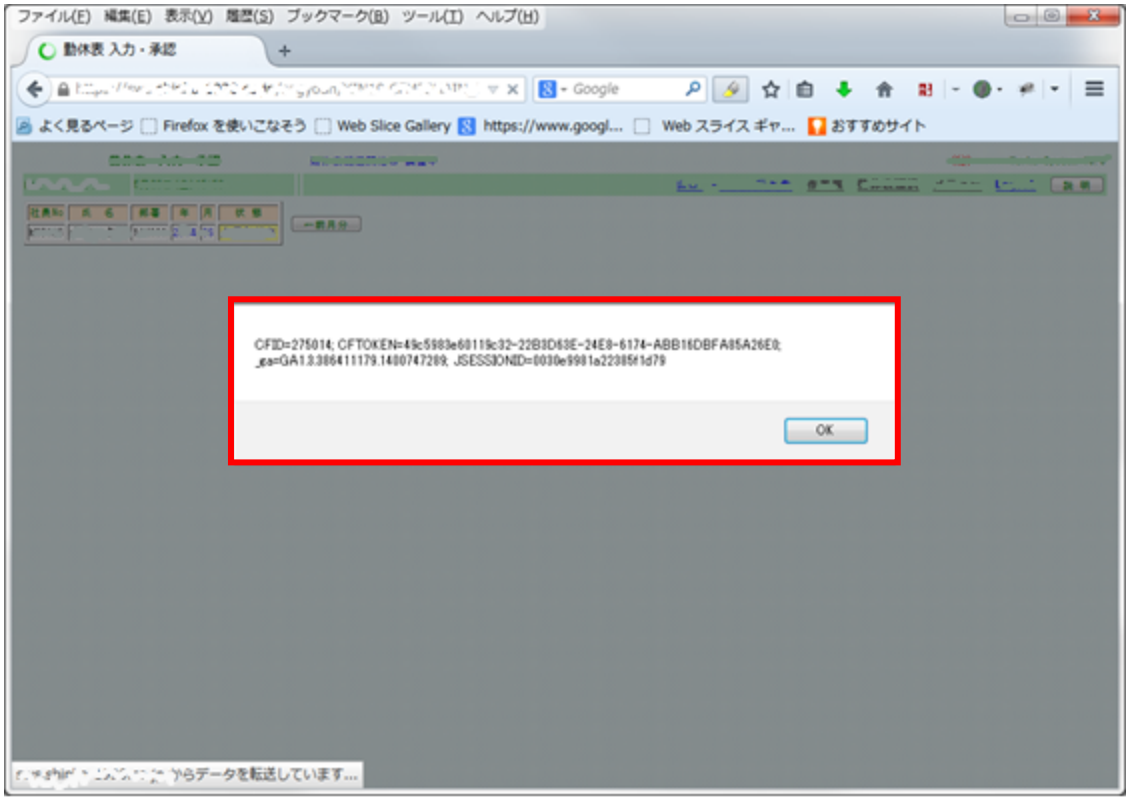


図 4-1

このようなクロスサイトスクリプティングの脆弱性を悪用することにより、Cookie の値を攻撃者のサーバに送信することでその値を盗みとることなどが可能となります。また、スクリプトにより画面の表示内容を別のものに見せかけることで、本来とは異なる別のサーバにデータを送信させられてしまう場合やフィッシングサイトとして悪用される場合があります。

(5) 想定される脅威

ユーザの Cookie の情報が攻撃者に取得され、他のユーザになりすましが行われるほか、サイトの内容を書き換えられ、さらにその結果としてフィッシングサイトとして悪用される可能性があります。

(6) 想定される被害

① 脆弱性悪用の実現度

クロスサイトスクリプティングの脆弱性を悪用するためには、被害者に不正なスクリプトが仕込まれたリンクをクリックさせたり、不正なスクリプトが仕込まれた Web ページを表示させたり、といったステップが必要となるため、比較的実現度は低いものと考えられます。しかしながら、クロスサイトスクリプティングという名称が世間的に広く知られている脆弱性であるため、脆弱性の存在が報道された場合の風評被害が大きいという特徴があります。

② 脆弱性悪用により想定される被害

- なりすましによる他のユーザ権限の不正利用
- なりすましによる個人情報の漏えい
- 情報漏えい等の事故を原因とするサイト停止による機会損失
- 情報漏えい等に対する損害賠償や見舞金
- 情報漏えい等の事故による企業の信用失墜

(7) 対策

ユーザから入力された値を画面上に出力するような処理の場合、出力する値の中にスクリプトなどの不正な値が含まれていないかを確認します。不正な値が含まれていた場合には、無害な文字に置き換える処理（サニタイジング）を行ってください。

また、より有効な対策としては、ユーザから値を渡された時点で値をチェックすることで、入力可能となる文字を制限する（郵便番号であれば数字とハイフンのみを受け付け、それ以外の値が混入したらエラー処理を行う）などの処理を行うことがあげられます。このような手法によって不正な文字列の混入を防止することで、より安全なサイトが構築可能となります。

(8) 参考情報

① 無害化処理について

一般的には HTML タグとして認識される文字を置換することになります。HTML タグとして認識される文字とは以下のようなものがあります。

- & ⇒ &
- < ⇒ <
- > ⇒ >
- " ⇒ "
- ' ⇒ '

また、ColdFusion ではクロスサイトスクリプティング対策用の関数が用意されていますので、これを利用することもできます。

```
<cfoutput>
<div>#HTMLEditFormat(Form.first)#</div>
<div>#HTMLEditFormat(Form.last)#</div>
</cfoutput>
```

② HttpOnly フラグについて

サーバにて Cookie を発行する際に HttpOnly フラグを設定することで、Internet Explorer などの Web ブラウザに対してスクリプトによる Cookie の値の読み取りの禁止を指定することができます。これにより、仮にクロスサイトスクリプティング脆弱性が存在したとしても、それによって受ける影響を小さくすることが可能です。

ただし、このフラグを設定することでクロスサイトスクリプティング脆弱性の影響を容認できるようになるわけではないことに注意が必要です。また、このフラグはほとんどのブラウザで有効ですが、Internet Explorer 6 SP1 から実装された設定項目のため、バージョンの古いブラウザではサポートされていないことにも注意が必要です。

なお、JavaScript から Cookie の値を参照するような処理を行っている場合、正常な動作に影響を与えてしまう可能性があります。実際に HttpOnly フラグを設定する場合には、そのような影響の有無を確認してから設定するようにしてください。

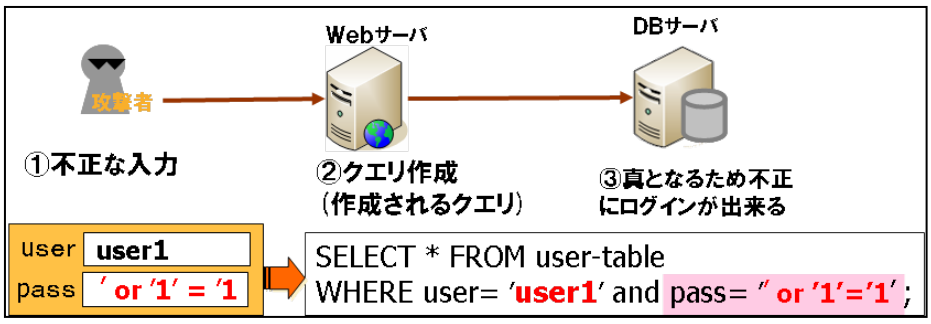
4.3. コマンドの実行

4.3.1. SQL インジェクション

(1) 危険度

High

(2) 脆弱性概要



データベースの SQL クエリのパラメータとなる入力に、不正な文字列を挿入（インジェクション）して、不正な SQL クエリを実行させる攻撃です。上図は不正な文字列により認証を回避する際の SQL インジェクション攻撃の例です。

(3) 発生箇所

No.	URL	パラメータ名
1	http://Example. co.jp /submit.cfm	key
2	・・・	・・・
3	・・・	・・・

(4) 脆弱性発生状況

発生箇所 No.1 を例に解説します。発生箇所に記載したパラメータに対して、SQL 文を入力することで、その SQL 文がそのまま実行される様子が確認されました。以下に詳細を説明します。

・・・（実際は、ここに詳細な再現手順を記載します）・・・

(5) 想定される脅威

データベースに保存されている情報の漏えい、改ざん、破壊等の可能性があります。

(6) 想定される被害

① 脆弱性悪用の実現度

攻撃方法にある程度の知識を必要とする点、攻撃成功のためには多数のリクエストが必要となる点を考慮すると、実現度はやや低くなります。また、攻撃者にとってより有効な情報を得るためには、非常に多くの SQL 文を実行することが必要となってきます。このため、侵入検知システムなどを導入している場合は容易に検知可能であると考えられます。

② 脆弱性悪用により想定される被害

- データベースに保存されているユーザ情報等の漏えい
- データベースに保存されているユーザ情報等の改ざん、破壊
- 情報漏えい等の事故を原因とするサイト停止による機会損失
- 情報漏えい等に対する損害賠償や見舞金
- 情報漏えい等の事故による企業の信用失墜

(7) 対策

SQL 文を文字列組み立てにより生成するのではなく、プレースホルダ、バインドメカニズムを用いてあらかじめ実行する SQL 文を決定しておくことを推奨します。

そのような対策が取れず、ユーザから入力された値を元にしてデータベースへの問い合わせを行う場合、実際にデータベースへの問い合わせを行う段階で、入力された値の中に不正な文字列が含まれていないかをチェックし、その結果としてエラー処理や無害な文字列に変換するなどの処理を行ってください。

また、あらかじめ入力される文字種が限定されているような場合、ユーザから値を渡された時点で値をチェックすることで、想定外の文字列が入力されることを制限する（郵便番号であれば数字とハイフンのみを受け付け、それ以外の値が混入したらエラー処理を行う）ようにしてください。このような手法によって不正な文字列の混入を防止することで、より安全なサイトが構築可能となります。

(8) 参考情報

① プレースホルダ、バインドメカニズム

発行するSQL文のうちで値を入力できる部分をあらかじめ決めておくことで、意図しないSQL文が発行されることを防ぐことが出来ます。

プレースホルダ、バインドメカニズムの使用方法は利用されているデータベースソフトウェアのマニュアルを参照してください。

② ホワイトリスト方式

不正な値を排除する方法として「不正な値が含まれていないか」を確認するよりも「正当な値か」を確認したほうがチェック漏れを減らすことが出来ます。例えば電話番号入力欄には入力された値が数字かを確認し、性別選択欄では男女のどちらかが入力されているかを確認します。「不正な値が含まれていないか」を確認する方法はブラックリスト方式と呼ばれており、「正当な値か」を確認する方法はホワイトリスト方式と呼ばれています。

入力値のチェックを行う場合には、「ホワイトリスト方式」を使用するように心がけてください。

③ 必要最小限の権限付与

万一攻撃に成功されてしまうことも想定してデータベースの権限分離を行っておくことで被害を最小限に食い止めることができます。例えば、データベースのシステムテーブル(ユーザIDやパスワード等を管理しているテーブル)には、一部のユーザだけがアクセス権を有していれば十分です。各データベース、テーブルに対して必要最小限のアクセス権が与えられていることを確認してください。

5 注意事項

ここに記載されている事項は、Web サイトの安全性を一層高めるため対応されることを推奨するものです。

5.1. システム情報の漏えい（バージョン情報）

今回診断した Web サイトでは、サーバからのレスポンスヘッダ内に、利用されているサーバ・ソフトウェアのバージョン情報が表示されていました。攻撃者はこのようなところから入手した情報を手掛かりにして、さらなる攻撃を行う可能性があります。可能な場合、レスポンスヘッダにサーバのバージョン情報等、重要な情報は含めないようにしてください。

```
HTTP/1.1 200 ACT
Content-Type: text/html;charset=UTF-8
Server: Microsoft-IIS/8.5
Date: Thu, 05 Feb 2015 05:37:14 GMT
Connection: keep-alive
Content-Length: 24115
```

以下のような手順を実施することで Server ヘッダの送信を抑制することが可能です。

仮想ディレクトリに App_Code というフォルダを作成し、同フォルダ内に以下の内容のファイルを保存します。ファイル名には拡張子として cs をつけてください（ここでは「DeleteHeaders.cs」とします）。

```
using System;
using System.Web;
using System.Configuration;

namespace HeaderDeleteModule
{
    public class DeleteHeaders : IHttpModule
    {
        public void Dispose() {}

        public void Init(HttpApplication context)
        {
            context.PreSendRequestHeaders += OnPreSendRequestHeaders;
        }

        void OnPreSendRequestHeaders(object sender, EventArgs e)
        {
            HttpContext.Current.Response.Headers.Remove("Server");
        }
    }
}
```

「インターネット インフォメーション サービス(IIS) マネージャー」を起動し、App_Code フォルダを作成した Web サイトにて[機能 ビュー] から [モジュール] アイコンをダブルクリックします。

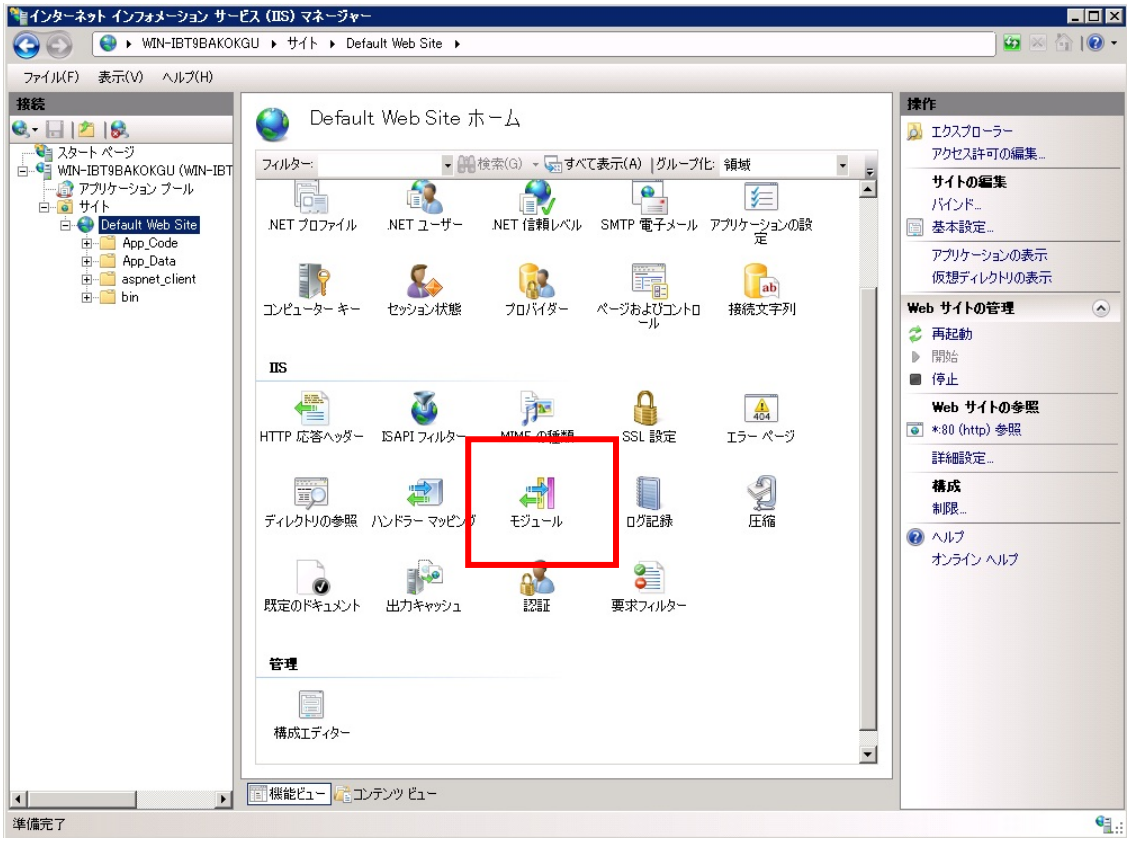


図 5-1

［モジュール］リストが表示されるので、画面右の［操作パネル］から［マネージモジュールの追加］リンクをクリックします。

［マネージモジュールの追加］ダイアログボックスが表示されるので、[名前］テキストボックスに 任意の名前を入力します（ここでは「DeleteHeaders」とします）。また、[種類］ドロップダウンリストボックスから、「HeaderDeleteModule.DeleteHeaders」を選択し、[OK］ボタンをクリックしてダイアログボックスを閉じます。

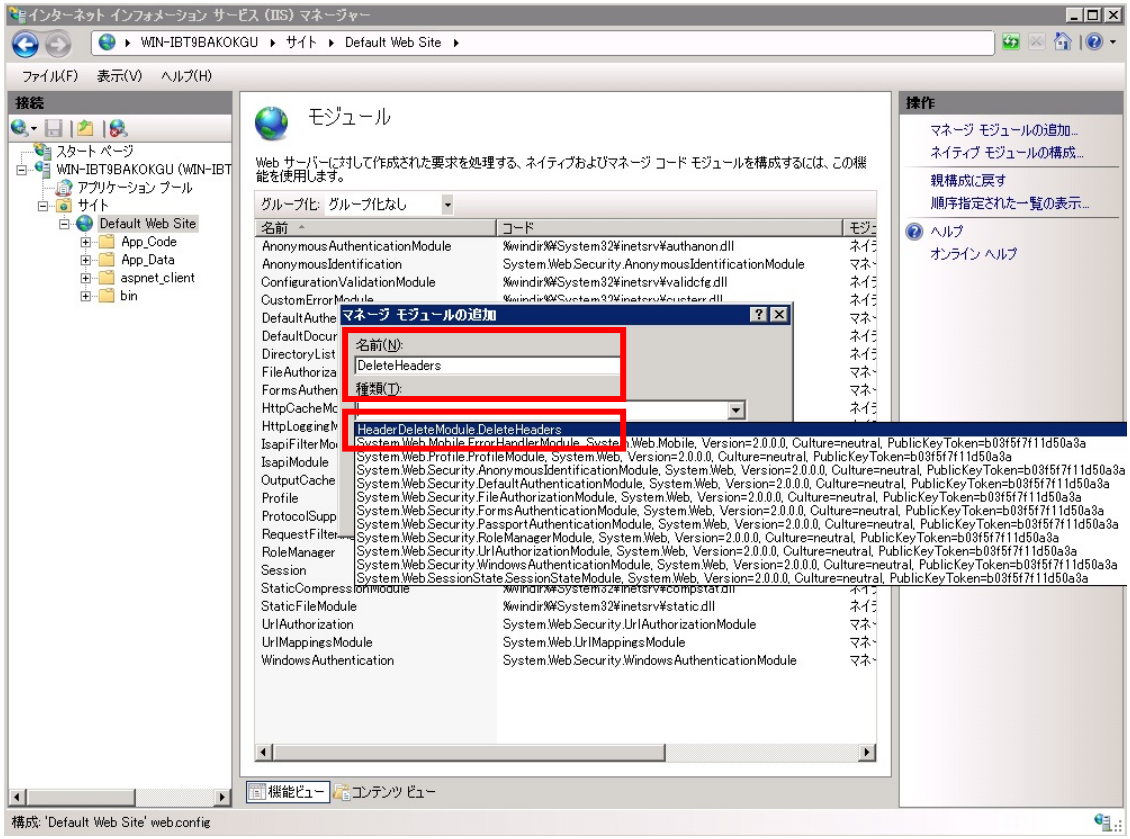


図 5-2

6 お問合せ

6.1. お問合せについて

本報告書の内容に関するお問合せは、下記メールアドレスまでご連絡ください。お問合せの対応は、報告書提出から 1 ヶ月以内に限らせていただきます。なお、電話や FAX でのお問合せは受付けておりませんので、ご了承ください。

webscan@shinko-1930.co.jp

6.2. 再診断について

弊社が指摘した脆弱性発生箇所をお客様にて修正された後に無料で再診断をご利用可能です。なお、再診断実施に当たっては以下の条件がございますので、ご注意ください。

- 再診断実施回数　1 回
- 期間　報告書提出から 1 ヶ月以内
- 診断箇所　弊社より指摘した脆弱性発生箇所（危険度 Medium 以上のもののみ）
- 成果物　報告書（報告会は実施しません）

6.3. その他のサービスについて

弊社では、Web アプリケーション診断をはじめとする、「診断・防御・教育」の 3 つのアプローチでサービスを提供しております。

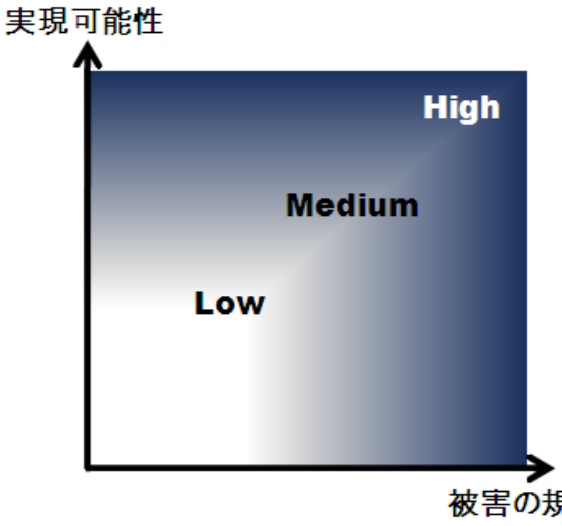
- プラットフォーム診断
  お客様のサーバやネットワーク機器に潜むセキュリティ上の問題点を診断します。

- Web アプリケーションファイアウォールサービス
  お客様サイトを攻撃する通信を弊社の Web アプリケーションファイアウォールが防御します。

- セキュリティ教育
  Web サイト管理者、開発者の視点に立ったセキュアな Web サイト設計のための教育サービスを提供します。

付録 A 危険度の判定基準

脆弱性の危険度は、以下の図のように攻撃の実現可能性と被害の規模を軸として High、Medium、Low の 3 段階の値に分けています。



・攻撃の実現可能性（縦軸）　：環境要因によるものや人的要因を考慮します。
・被害の規模（横軸）　：被害者数や被害の種類などを考慮します。

- High　攻撃の実現可能性が高く、被害の規模も大きい
- Medium　攻撃の実現可能性が低いかまたは被害の規模が小さい
- Low　攻撃の実現可能性が低く被害の規模も小さい

なお、実現度が低くても被害の規模が過度に大きいようであれば High と判定し、また実現度が高くても、被害の規模が小さいようであれば Low と判定します。

付録 B 参考文献

1. セキュアプログラミング講座
   http://www.ipa.go.jp/security/awareness/vendor/programming/
2. 新版 セキュアプログラミング講座
   http://www.ipa.go.jp/security/awareness/vendor/programmingv2/
3. 情報処理推進機構 安全なウェブサイトの作り方
   http://www.ipa.go.jp/security/vuln/documents/website_security.pdf
4. 情報処理推進機構 安全な SQL の呼び出し方
   http://www.ipa.go.jp/security/vuln/documents/website_security_sql.pdf
5. NPO 日本ネットワークセキュリティ協会「2013 年 情報セキュリティインシデントに関する調査報告書～個人情報漏えい編～」
   http://www.jnsa.org/result/incident/index.html

