CONFIDENTIAL

2023 年 6 月 14 日／文書番号：SKS-WR-20230614-S01-01

株式会社信興テクノミスト 御中

Web アプリケーション診断報告書

株式会社 信興テクノミスト

ハイブリッドWebセキュリティ診断

ABURIDA®

目次

1 はじめに

1.1. 本報告書について

本報告書は、お客様よりご依頼いただいた Web アプリケーションの脆弱性を診断した結果を報告するものです。

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。

1.2. 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。

1.3. 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。

1.4. 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

診断環境は、以下の通りです。

診断開始日	2023/06/05(月)
診断終了日	2023/06/09(金)
診断元 IP アドレス	124.35.140.166 122.249.147.224 118.243.81.247 118.238.212.37 210.166.157.210 13.112.165.80 153.120.31.228

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 ✓ クロスサイトリクエストフォージェリ
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**

ABURIDA FRUITS STORE

2.3.2. **対象** URL

(1) http://aburidafruits.com/demo/01_xss_detail.html
(2) http://aburidafruits.com/demo/get_creditcard.php
(3) http://aburidafruits.com/demo/get_mypage.php
(4) http://aburidafruits.com/demo/get_nickname.php
(5) http://aburidafruits.com/demo/get_userid.php
(6) http://aburidafruits.com/demo/index.html
(7) http://aburidafruits.com/demo/login.php
(8) http://aburidafruits.com/demo/xss.php
(9) http://aburidafruits.com/demo/02_sqli_mypage.html
(10) http://aburidafruits.com/demo/03_authority.html

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	aburidafruits.com
データベースサーバ	MySQL
Web サーバ	Apache
開発言語	PHP

3 診断結果概要

3.1. 総合評価

E.非常に危険な状態です

＜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	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	High	1
コマンドの実行	SQL インジェクション	High	1

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、一部のパラメータにおいてチェック漏れがあるため、非常に危険度の高い脆弱性が複数存在することが確認されました。

- ✓ いくつかのパラメータでチェック漏れが発生しており、その漏れがクロスサイトスクリプティングの脆弱性を発生させる原因となっていることがわかりました。この脆弱性を悪用された場合、なりすましやフィッシング詐欺などの被害が発生する可能性があります。
- ✓ 一部の機能においてデータに対するアクセス権限のチェックに不備が発生していることが確認されました。このような不備がある場合、その発生している箇所や状況によっては非常に危険なものとなる可能性があります。
- ✓ SQL インジェクションの脆弱性が存在することが確認されました。こ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。

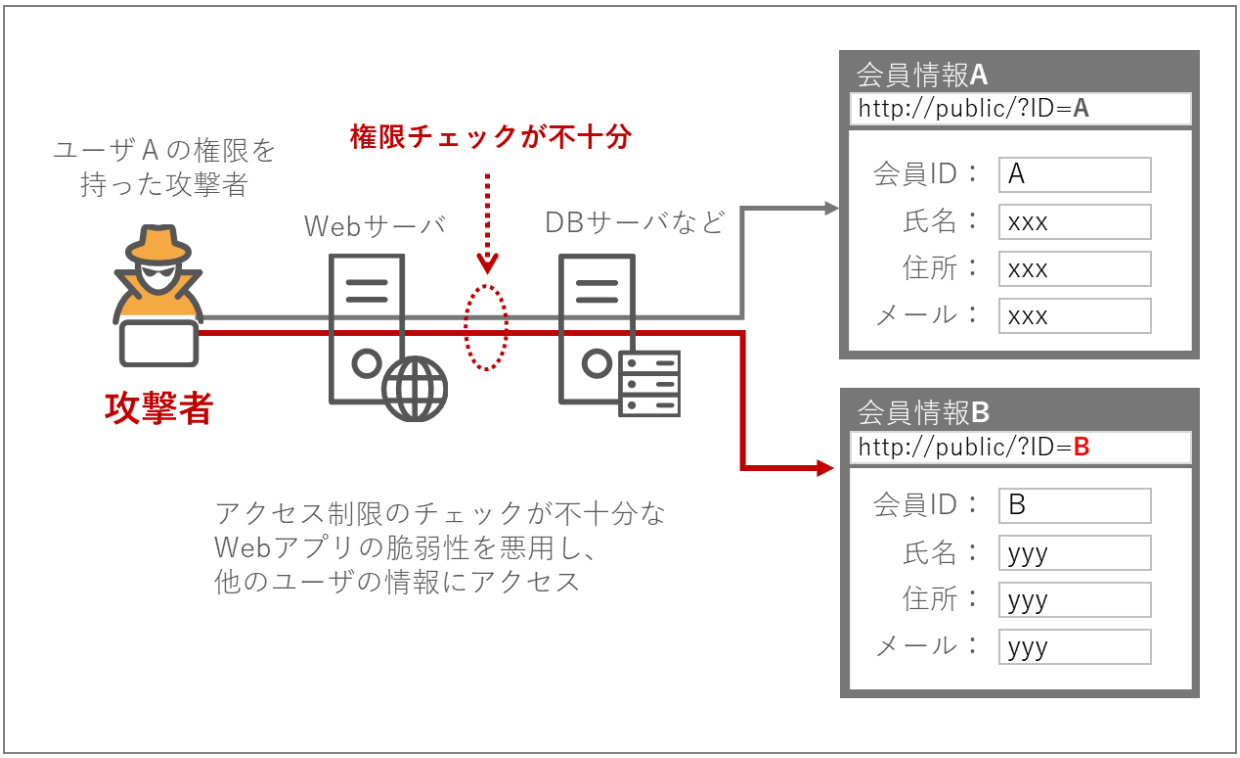
今一度 Web サイトのフローを見直されてセキュリティ対策の徹底を図っていただければと思います。

4 診断結果詳細

4.1. 承認

4.1.1. 不適切な承認(権限のないデータの利用)

危険度：High



アクセス管理が不十分なため、本来ならば権限が無くアクセスできないはずのページや機能へアクセスできます。

(1) 発見場所

No.	URL	パラメータ名
1	http://aburidafruits.com/demo/get_mypage.php	JSON パラメータ userId の値

※ 本脆弱性は、診断対象全体に対し数箇所をピックアップして手動診断により確認したものとなります。従って、上記の発生箇所以外にも同様の脆弱性が存在する可能性があります。

(2) 脆弱性発生状況

本来ならばアクセスすることのできないページや機能に対してアクセス可能な状態となっていることが確認されました。

まず、userid「seki」のユーザでログインして、マイページを表示すると図 4-1 のような画面が表示されます。

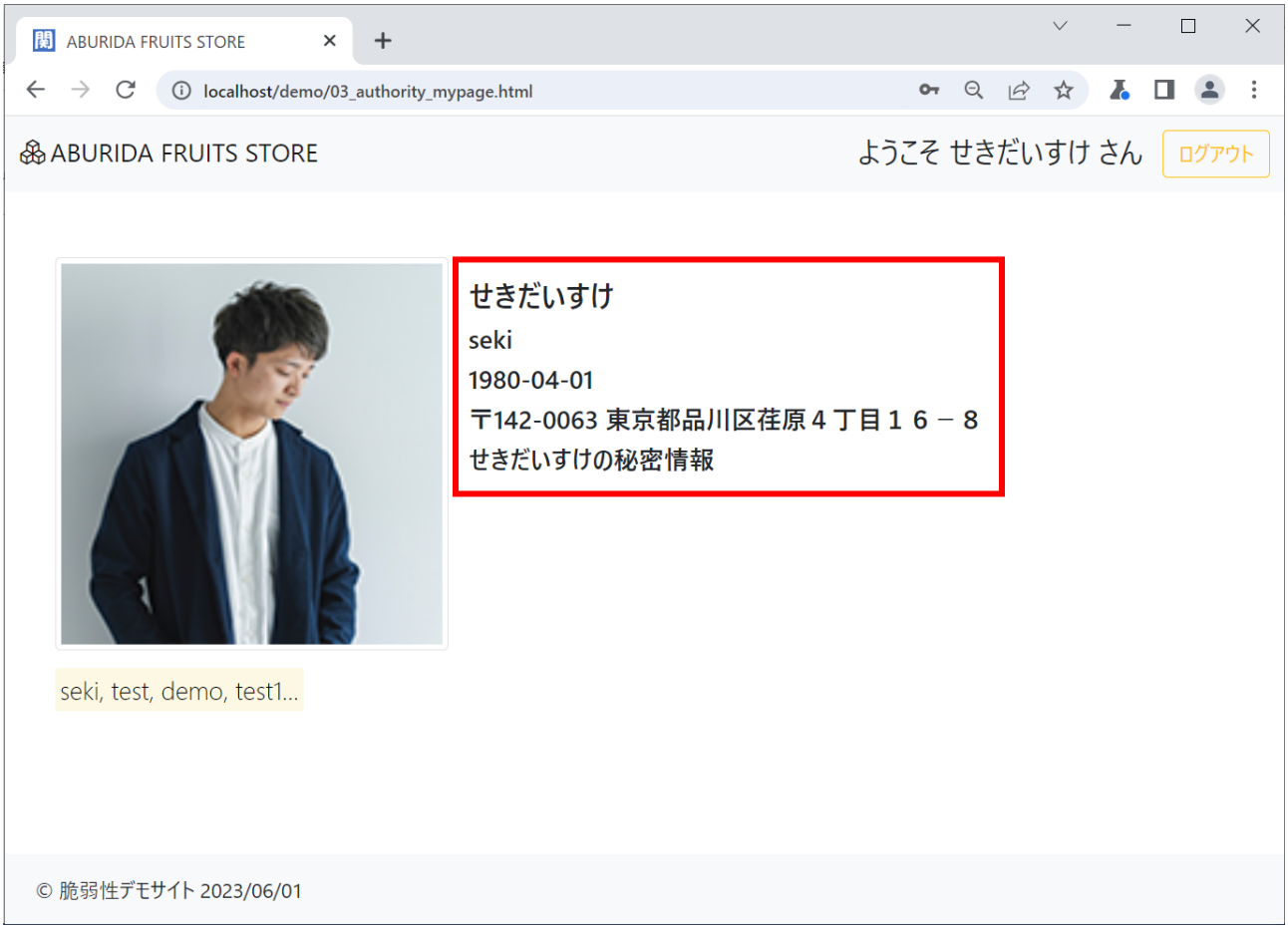


図 4-1

上図より、userid「seki」のユーザは「せきだいすけ」であることが確認されます。また、マイページにはユーザの住所等の個人情報などが表示されることが確認されます。

ここで、「マイページ」をクリックした際に送信されるリクエストのパラメータ userid に対して以下のような改ざんして送信します。

userid=~~seki~~test

すると図 4-2 のような画面が表示されました。

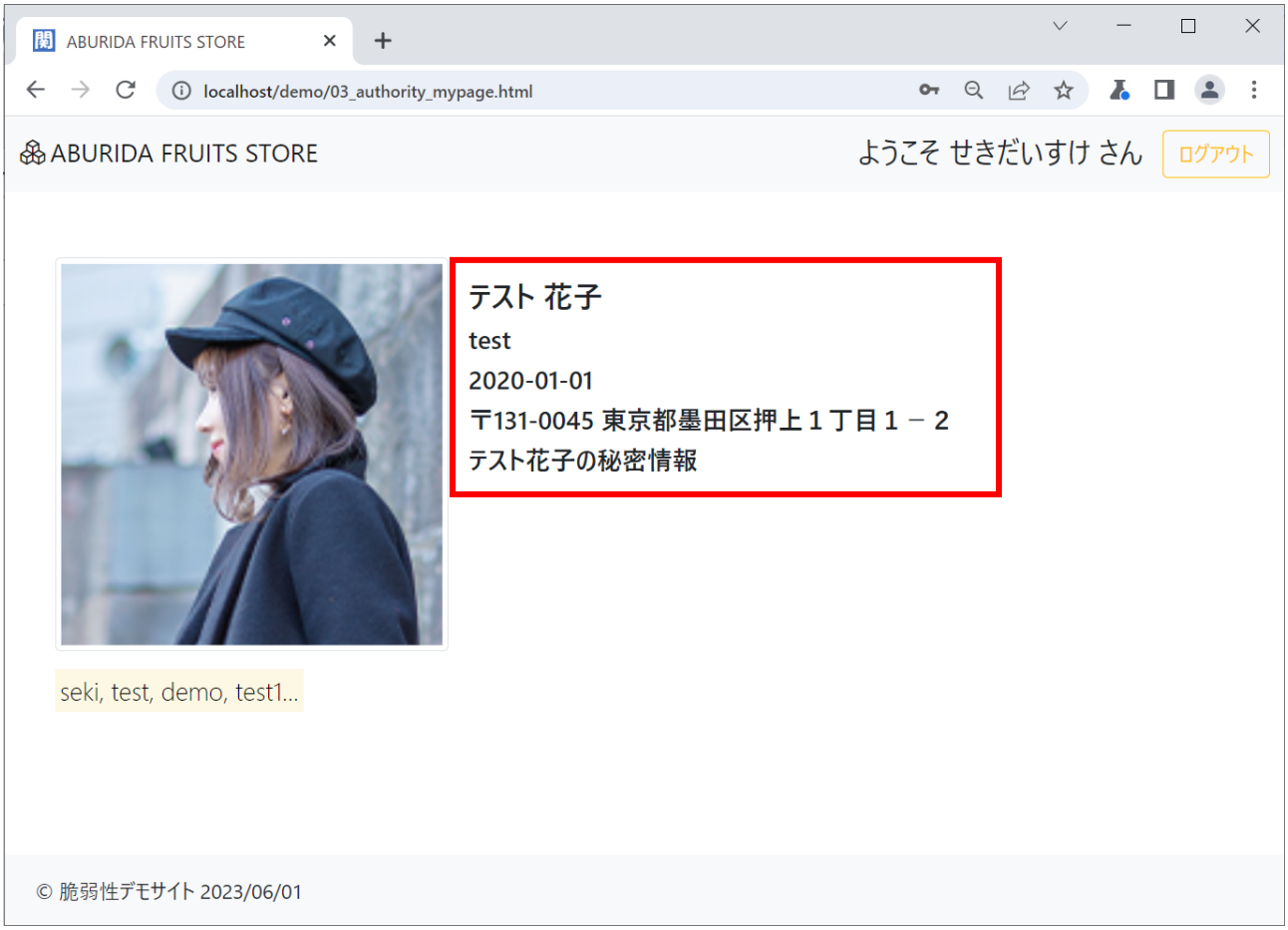


図 4-2

上図より、userid「seki」のユーザではない「テスト花子」の情報が表示されていることが確認されます。

以上の結果から、マイページにおいて他のユーザのマイページ情報を閲覧できることが確認されました。

(3) 想定される脅威

権限を有していないユーザによって各種機能を利用されることにより、データの漏えいや改ざん、なりすまし等が行われる可能性があります。

(4) 想定される被害

① 脆弱性悪用の実現度

攻撃対象のパラメータに対する有効な値が容易に推測できる場合、攻撃が成功する可能性が高くなります。

userid 等の攻撃対象のパラメータに対する有効な値が容易に推測できる為、攻撃が成功する可能性は高くなります。

② 脆弱性悪用により想定される被害

- ✓ 重要なデータの漏えい・改ざん
- ✓ 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
- ✓ 個人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

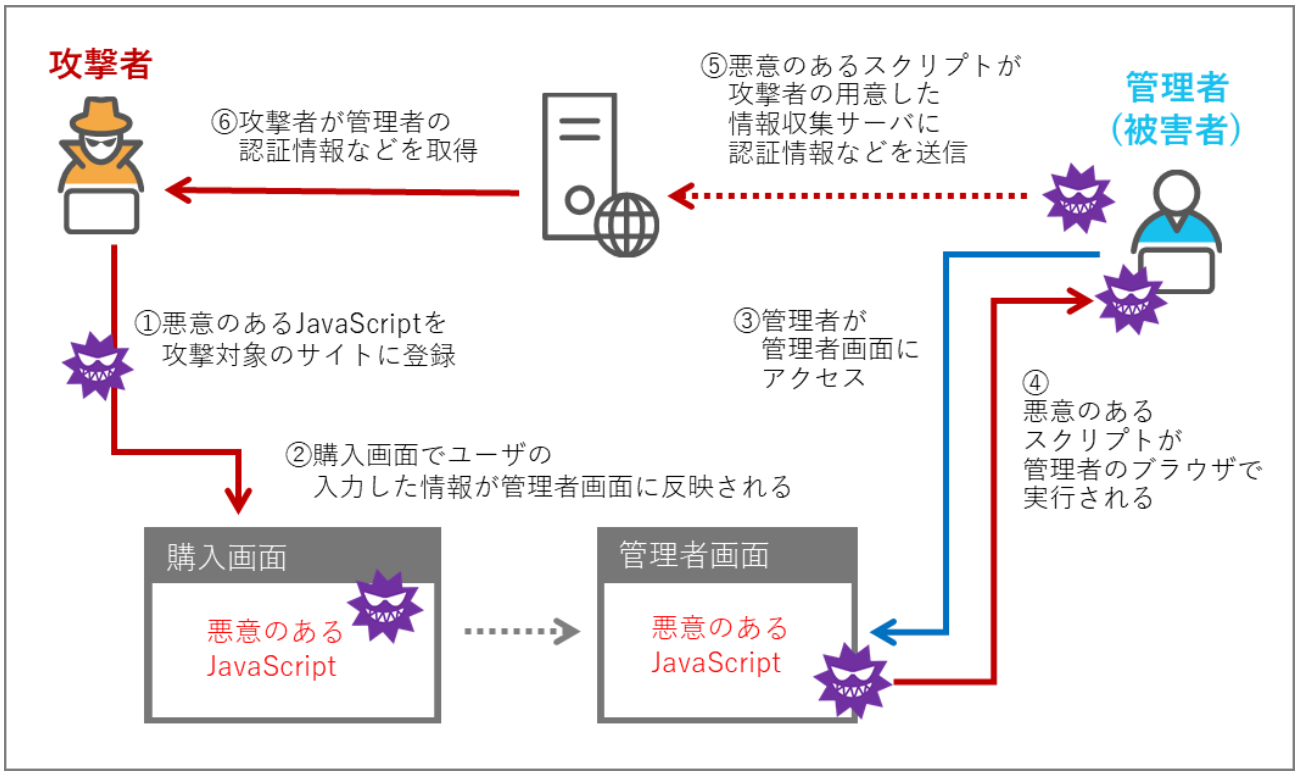
③ 対策

ユーザがデータへのアクセスを要求してきている場合、そのデータに対して、適切なアクセス権限を持っているかどうかのチェックを厳密に行うようにしてください。

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

危険度：High



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。

(1) 発見場所

No.	入力 URL	パラメータ名	発生 URL
1	http:// aburidafruits.com / demo/xss.php	comment	http://localhost/demo/01_xss_detail.html

※ 「入力 URL」、「パラメータ名」で指摘している箇所にてスクリプトが混入されると、「発生 URL」へのアクセスが発生した際に混入したスクリプトが実行されます。

※ 「パラメータ名」で送信した値がレスポンスに複数出力される可能性があります。すべての出力箇所に対して対策を行ってください。

(2) 脆弱性発生状況

「レビュー投稿」画面にて、「レビューを投稿する」をクリックした際に送信されるパラメータ comment 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。

```
美味しかったです<script type='text/javascript'>if(window.confirm('アンケートに回答していただけますか？')){location.href='./01_xss_trap.html'}</script>
```

次に、レビューを投稿した商品の詳細画面を表示します。
その結果、図 4-3 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。

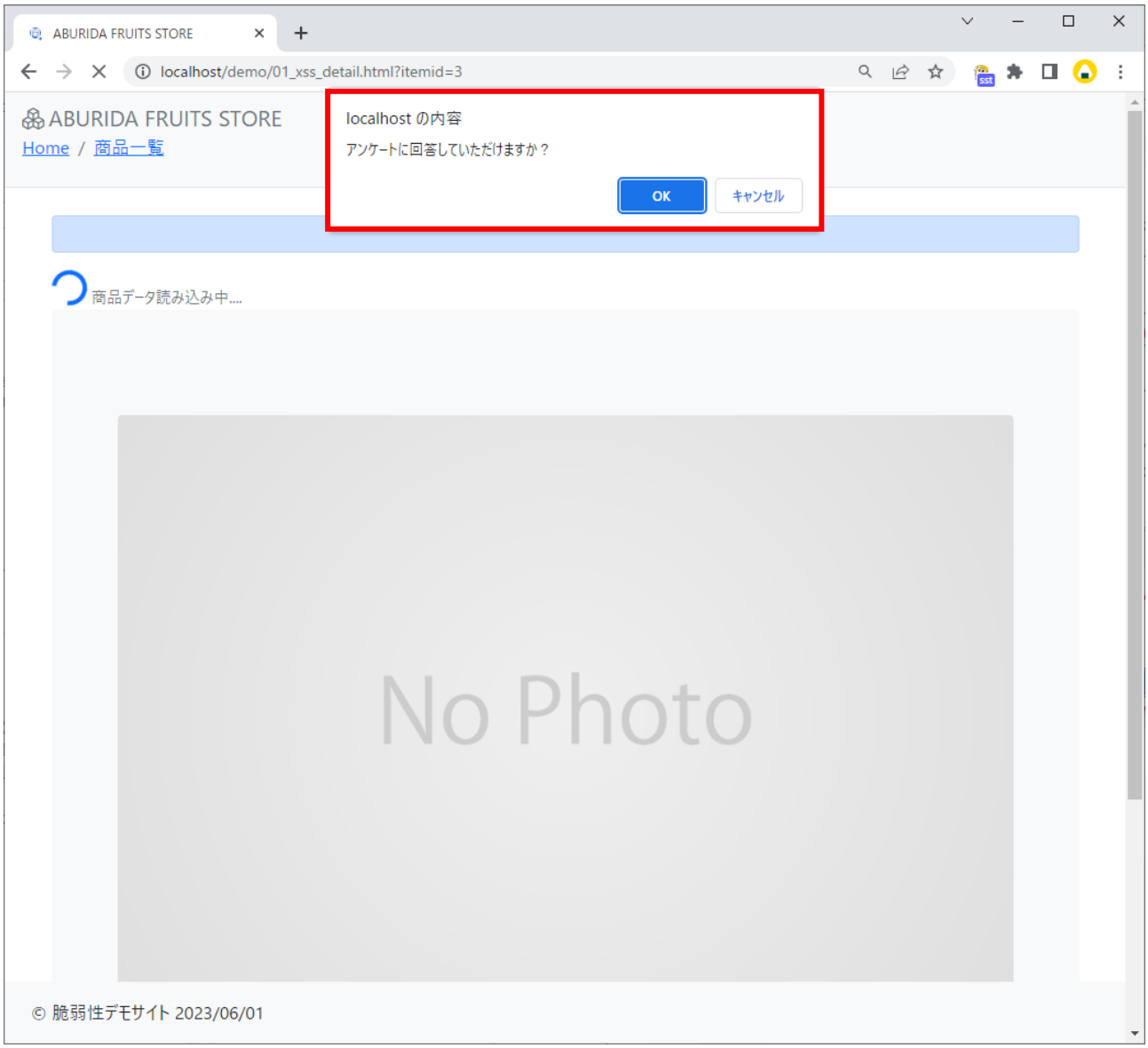


図 4-3

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。

(3) 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。

(4) 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

③ 対策

ユーザから入力された値を HTML 内に出力するような処理の場合、出力する値の中に「<」「>」「"」「'」「&」など HTML として特定の意味を持つ文字がそのまま出力されないよう、エスケープ（「<」ではなく「<」と出力する等）を行ってください。

また、補助的な対策として、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。

(5) 参考情報

① エスケープについて

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、いくつかの言語では HTML タグを変換する関数が用意されていますので、それらの関数を使うこともできます。

PHP の場合には htmlspecialchars()関数を使用することで HTML タグが入力された場合に無害な文字に置き換えることができます。

例として PHP では Cookie を発行する関数である setcookie を呼び出す際の引数にて赤字部分のように記述することで HttpOnly フラグを設定することが可能です。(PHP5.2.0 以上で利用可能)

```
setcookie("sstcookie", "value",0, "/", "www.example.com", false, true);
```

また、php.ini ファイルにて以下のように設定することでセッション管理に利用している Cookie に対して HttpOnly フラグを設定することが可能です。

```
session.cookie_httponly = 1
```

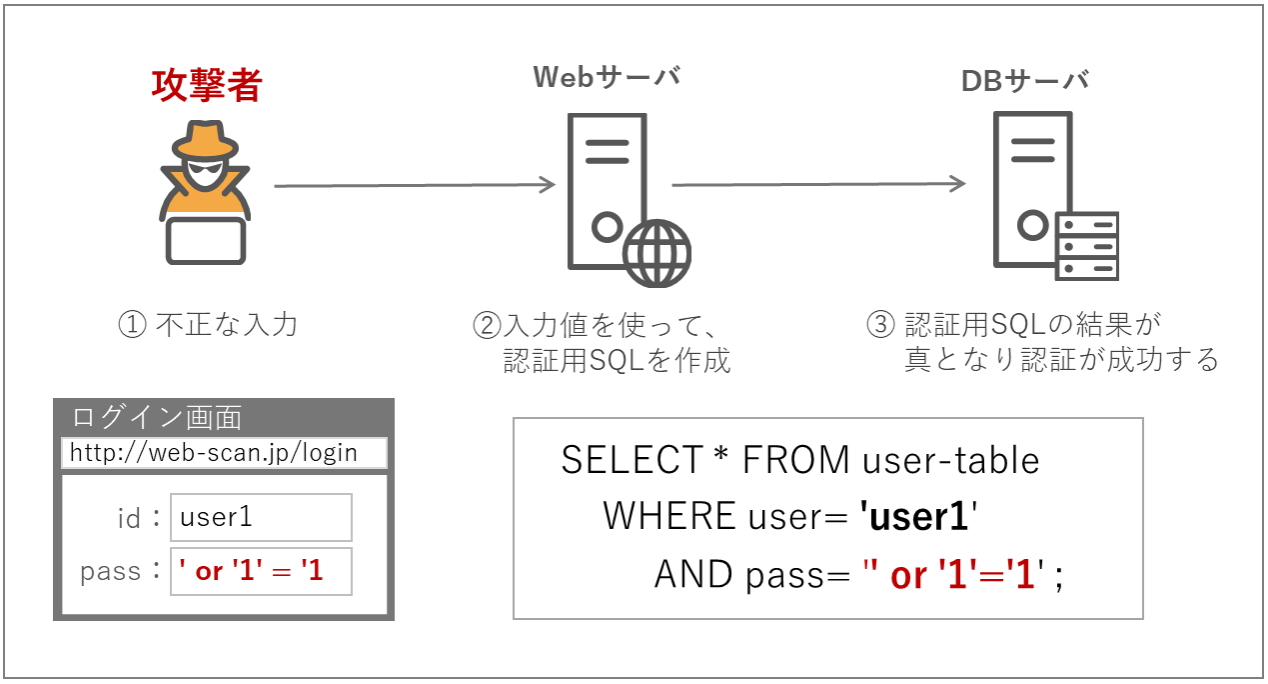
ただし、このフラグを設定することでクロスサイトスクリプティング脆弱性の影響を容認できるようになるわけではないことに注意が必要です。

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

危険度：High



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。

(1) 発見場所

No.	URL	パラメータ名
1	http://localhost/demo/get_creditcard.php	JSON パラメータ userid キーの値

(2) 脆弱性発生状況

まず、userid「seki」のユーザでログインして、マイページを表示すると図 4-4 のような画面が表示されます。

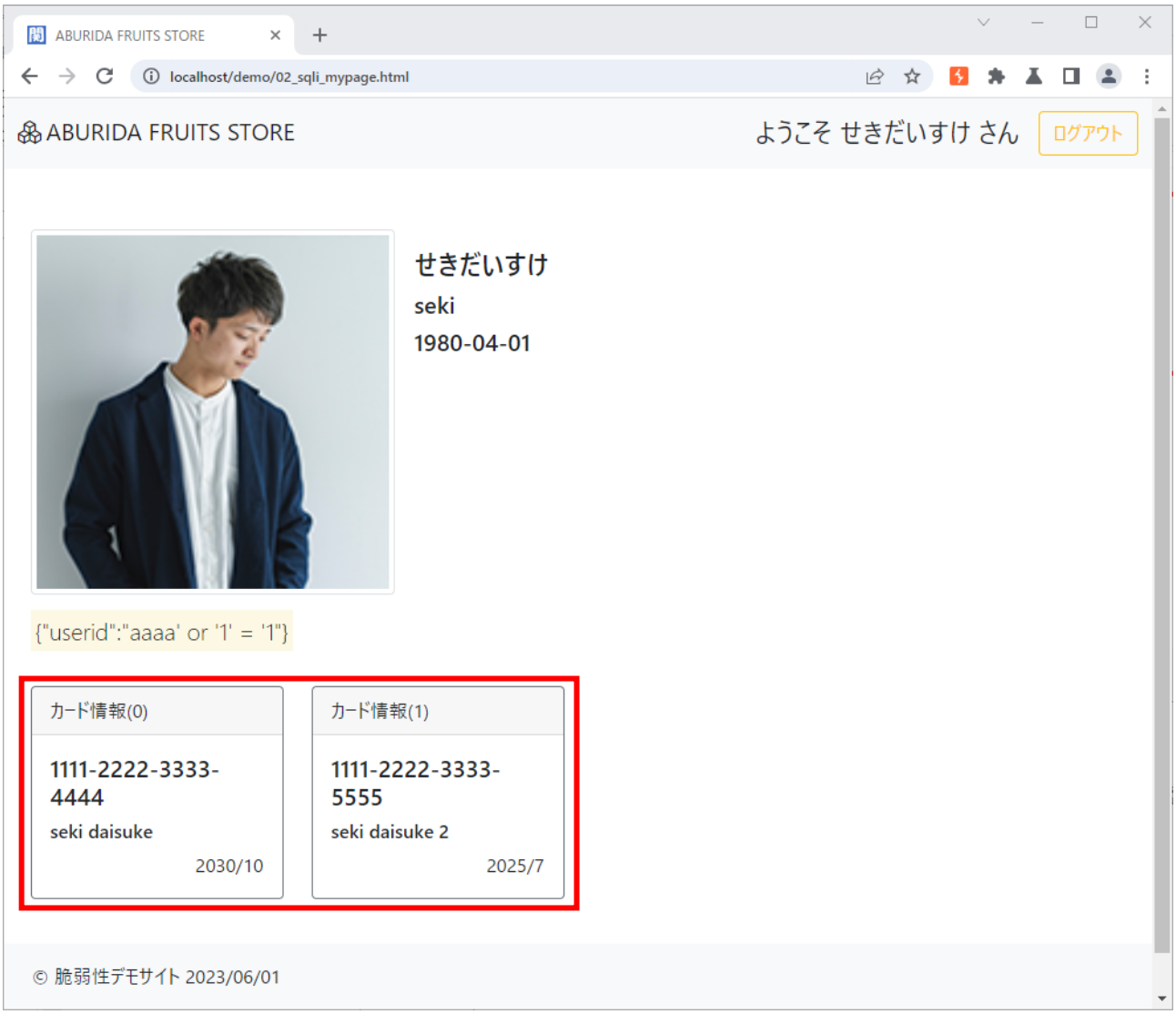


図 4-4

上図より、userid「seki」のユーザは「せきだいすけ」であることが確認されます。また、マイページにはユーザのカード情報が表示されることが確認されます。ここで、マイページを表示する際に送信されるリクエストのパラメータを以下のように改ざんして送信します。

{"userid":"seki' or '1' = '1"}

その結果、図 4-5 のような画面が表示されました。

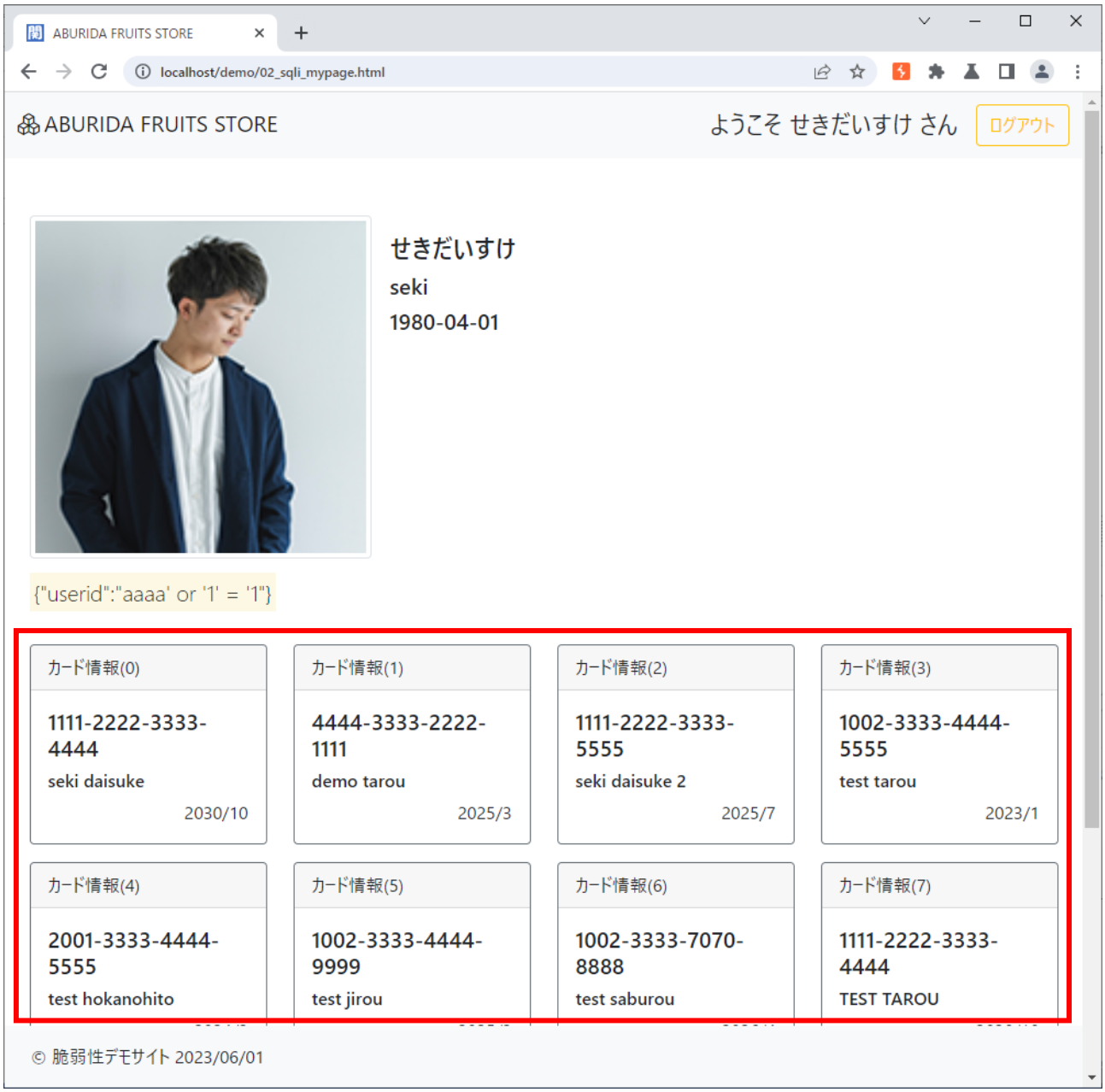


図 4-5

上図より、userid「seki」のユーザに登録されたカード以外のカード情報が表示されていることが確認されます。

本脆弱性の存在により、攻撃者によってさらに多くの SQL 文を実行されることで重要な情報の漏え

いなどが発生する可能性があります。

(3) 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。

(4) 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

③ 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。

なお、Web アプリケーションフレームワークによっては、オブジェクトとデータベースをマッピングする機能が用意されている場合があります。このような機能を活用すると SQL 文の組み立てをフレームワークに任せることになるため、安全にデータベースにアクセスすることができます。

(5) 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。

PHP の場合は以下のようになります。

```
// $con という名前の DB オブジェクトを取得している場合
$sql = ”select username from usertable where userid = ? and password = ?”;
$parameter = array(”sst”,”secret”);
$result = $con->getAll($sql, $parameter);
```

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 お問合せ

5.1. 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

5.2. 再診断について

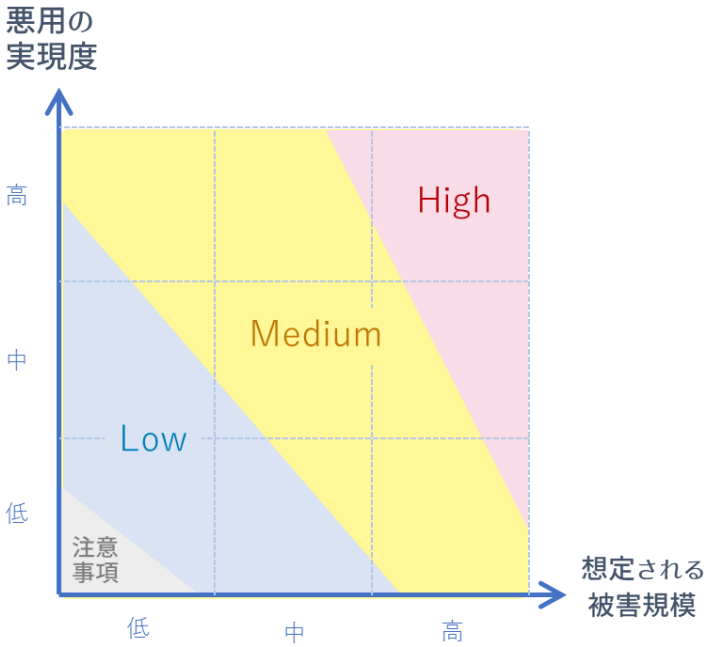
弊社が指摘した脆弱性発生箇所をお客様にて修正された後、無料で再診断をご利用可能です。なお、再診断実施に当たっては以下の条件がございますので、ご注意ください。

- ✓ 再診断実施回数　1 回（2 回目以降は有償で承ります）
- ✓ 期間　報告書提出から 1 ヶ月以内
- ✓ 診断箇所　危険度 Medium および High で検出された脆弱性発生箇所
- ✓ 成果物　再診断報告書

付録 A

危険度の判定基準について

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。



- ✓ 悪用の実現度　検出された脆弱性を環境や人的要因を考慮し実現の可能性を考慮
- ✓ 想定される被害規模　被害者数や被害の種類などを考慮

上図の通り、検出された脆弱性の種別だけではなく、被害の規模や実現度などを考慮した上で、危険度の判定を行っております。

なお、提出した報告書の危険度判定の修正を行う場合は、その理由を併記させて頂きます。

付録 B

参考文献

検出された脆弱性の内容や対策方法などは、以下の文献やサイト情報を参考にしております。

- ✓ IPA 安全なウェブサイトの作り方
  https://www.ipa.go.jp/security/10threats/index.html
- ✓ IPA 情報セキュリティ 10 大脅威
  https://www.ipa.go.jp/security/vuln/websecurity/about.html
- ✓ NPO 日本ネットワークセキュリティ協会「情報セキュリティインシデントに関する調査報告書」
  https://www.jnsa.org/result/incident/index.html
- ✓ OWASP Japan
  https://owasp.org/www-chapter-japan/