Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

Web アプリケーションのパラメタ改ざん脆弱性検出精度向上手法

北澤繁樹† 河内清人† 桜井鐘治†

本論文では，ユーザから HTTP を介して Web アプリケーションに渡されたパラメタの中から，セキュリティ上の重要な意味を持つパラメタを自動で抽出し，抽出されたパラメタの改ざんが成功しているかどうかを判断することによって，パラメタ改ざん脆弱性を高精度かつ効率的に検出する手法を提案する．

提案手法を用いることにより，セキュリティ上の重要な意味を持つパラメタのパラメタ改ざん脆弱性のみ検出できるようになる．これにより，セキュリティ試験ツールによる検出結果を人手で判断する必要がなくなるため，Web アプリケーションのセキュリティ試験に関する専門的な知識を持たない開発者であっても容易にパラメタ改ざん脆弱性を検出することができるようになる．

An Improved Accuracy Method for Detection of a Web Application Parameter Tampering Vulnerability

Shigeki Kitazawa†, Kiyoto Kawauchi† and Shoji Sakurai†

In this paper, we propose an improved accuracy method for detection of a Web application parameter tampering vulnerability. Our method can detect whether a parameter tampering is success or not, with high-accuracy and efficiency. In our method, important parameter on security is extracted automatically from parameters which send from user to a Web application via HTTP.

If you employ our method, you can detect only parameter tempering vulnerabilities which are an important parameter on security. From this feature, since you don't necessary to determinate a result of Web application assessment is correct by hand, you can detect a Web application parameter tampering vulnerability easily even if you don't have special skills about Web application security test.

1. †はじめに

近年，Web アプリケーションを狙った攻撃が増加の一途を辿っている．例えば，インターネットで観測される攻撃のうち，60%以上が Web アプリケーションを対象とした攻撃であるとの報告がある [1)]．Web アプリケーションが攻撃された場合，Web ページの改ざん，顧客情報の盗難，マルウェア配布サイトへの誘導など重大な被害に繋がる．

Web アプリケーションの多くは，各サイトで提供するサービス，デザイン等に応じて各サイトで独自に開発されている．

通常，Web アプリケーションの開発では，一般的なアプリケーションの開発と同様，開発元において開発仕様に基づいて試験を行い，開発仕様どおりに Web アプリケーションが動作することを確認してからリリースされる．

Web アプリケーションの脆弱性を狙った攻撃による被害が増加する中，Web アプリケーションに対しては，開発仕様に基づいた機能の動作試験だけではなく，Web アプリケーションのセキュリティに関する試験も実施する必要性がでてきた．

この場合，一般には Web アプリケーションの動作試験を完了した後，Web アプリケーションをインターネットへ公開する前に，セキュリティサービスベンダが提供する Web アプリケーション診断サービスなどを利用して，開発した Web アプリケーションにセキュリティの脆弱性がないことを確認する．これは，Web アプリケーションのセキュリティ試験には，Web アプリケーションの脆弱性に関する専門的な知識が必要であり，Web アプリケーションの開発元では十分な試験が行えないためである．しかしながら，Web アプリケーションのセキュリティ試験は，Web アプリケーションが完成してからの実施となるため，仮に，Web アプリケーションに脆弱性が見つかった場合には，脆弱性を修正するための手戻りが発生してしまうことが課題となる．

そこで，本論文では，Web アプリケーションの開発段階の試験において Web アプリケーションのセキュリティ試験に関する専門的な知識がない Web アプリケーションの開発者であっても，セキュリティ試験を自動化したツールを用いてセキュリティ試験を実施できるようにすることを目的として，特に，人手によって脆弱性の有無の判断が求められる Web アプリケーションのパラメタ改ざん脆弱性を，人手を介さずに検出する方法について提案する．提案手法により，Web アプリケーションのセキュリティ試験に関する専門的な知識がなくてもセキュリティ試験を開発段階において実施可能となる．したがって，脆弱性を修正するための手戻りを低減できるだけでなく，十分な Web アプリケーションのセキュリティ試験が行えるようになるため，Web アプリケーションに開発段階で内包されてしまう脆弱性を低減できる．

† 三菱電機株式会社 情報技術総合研究所
Mitsubishi Electric Corporation, Information Technology R&D Center

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

本論文の流れは，次のとおりである．まず，2 章では，本論文で取り扱う Web アプリケーションのパラメタ改ざん脆弱性に関して説明する．3 章では，2 章で説明した Web アプリケーションのパラメタ改ざん脆弱性を，人手を介さずに精度よく検出する手法を提案する．4 章では提案手法についての考察を行い，5 章にて関連研究について触れた後に，6 章で本論文のまとめを行う．

2. Web アプリケーションのパラメタ改ざん脆弱性

2.1 Web アプリケーションのパラメタ

Web アプリケーションと Web クライアント間でやり取りされるパラメタとしては，以下がある．

- URL パラメタ
  Web プログラムの呼び出しを行う URL に含まれるパラメタ．
- POST パラメタ
  Web プログラムの呼び出しを行う HTTP リクエストのボディ部分に含まれるパラメタ．Web ページ間のデータの受け渡しなどに用いられる．
- Cookie
  Web サーバが，Web クライアントへ送付しておくデータ．Web サーバ側からの要求により読み出しが行われ，HTTP レスポンスに含められて Web サーバへ送信される．Web クライアントの一意な識別やユーザの識別に用いられる．

フォームを用いた URL パラメタや，POST パラメタの受け渡しでは，パラメタのタイプによって text 属性，password 属性，file 属性，hidden 属性などの属性が定義されている．図 2-1 に，フォームを用いたログイン画面の構成例を示す．図 2-1 で示したログイン画面では，「名前」，「パスワード」の入力欄と，「送信」，「取消」のボタンが表示される．hidden 属性のパラメタは画面には表示されない．

2.2 パラメタ改ざん脆弱性

パラメタ改ざん脆弱性とは，2.1 節で説明した Web アプリケーションで扱われるパラメタに格納されている値を攻撃者が自由に書き換えて Web アプリケーションに送信することによって，改ざんされたパラメタの値を読み込んだ Web アプリケーションに誤動作を引き起こさせる攻撃に対する脆弱性である．パラメタを改ざんしたことにより，管理画面などの，本来そのユーザに表示されるべきではないページが表示されたり，パラメタを基に本来処理されて得られる結果を自由に変更できたりといったことが起こりえる．

```
<form method="POST" action="login.cgi">
  <table>
    <tr>
      <td>名前：</td>
      <td><input type="text" name="ID" value=""></td>
    </tr>
    <tr>
      <td>パスワード：</td>
      <td><input type="password" name="PASSWD"></td>
    </tr>
    <tr>
      <td>
        <input type="submit" name="ok" value="送信">
        <input type="hidden" name="pageID" value="ID_PASS">
        <input type="reset" value="取消">
        <input type="hidden" name="Cancellation">
      </td>
    </tr>
  </table>
</form>
```

図 2-1 フォームを用いたログイン画面の構成例

例えば，パラメタ改ざん脆弱性が存在する商品購入 Web サイトにおいて価格に関係する重要なパラメタを改ざんすることによって，本来支払うべき値段とは異なる値段で商品が購入できてしまうといったことも考えられる．

この場合，Web アプリケーションでは，正常に決算処理が終了しているため，決算処理結果を人が確認し，不正な価格で決算処理されていることに気付かなければ，被害を受けたことすら分からないといった状況に陥ることも考えられる．

このように，パラメタ改ざん脆弱性は，その結果として様々なセキュリティ上の不具合を引き起こす原因となりうる．

2.3 パラメタ改ざん脆弱性への対策

Web アプリケーションにパラメタ改ざん脆弱性が存在してしまう原因は，ユーザから Web サーバへ送られてきたパラメタの値をサーバ側でそのまま信用して改ざんされた値を使ってその後の処理を進めてしまうことである．したがって，対策としてはその後の処理に重要な影響を及ぼす可能性のあるパラメタは，改ざんされていることを前提として，ユーザから受け取った値をそのまま使わないよう，Web アプリケーションを設計することである [2]．

しかしながら，どのパラメタが，いつ，どの処理において，どんな影響を及ぼすかを考慮しながら設計することは，開発者に十分な知識と経験が要求されるため，人的

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

な設計ミスによって，Web アプリケーションにパラメタ改ざん脆弱性が作りこまれてしまうことが後を絶たない．

3. 提案手法

3.1 提案手法が解決する課題

2 章では，Web アプリケーションのパラメタ改ざん脆弱性について，概要とその対策まで述べた．対策でも述べたとおり，パラメタ改ざん脆弱性をなくすためには，開発者に十分な知識と経験が要求されるため，人的な設計ミスを完全に防ぐことは困難である．したがって，開発した Web アプリケーションに対して，繰り返しセキュリティ試験を行うことによって，パラメタ改ざん脆弱性を検出して修正する必要がある．

しかしながら，セキュリティ試験を自動化したツールを用いてセキュリティ試験を実施した場合，検出された項目の中には誤検出が含まれていることがある．したがって，セキュリティ試験ツールが検出した脆弱性がセキュリティ上の脅威となりうるかどうかの最終的な判断は人手によって行わなければならないため，セキュリティ試験の実施者には，専門的な知識が要求される．特に，パラメタ改ざん脆弱性に関しては，一見，パラメタ改ざんに成功したように見えても，改ざんに成功したパラメタの重要性によって，その後の処理でセキュリティ上の不具合が発生するかどうかの結果は異なってくるため，人手による判断が難しい．したがって，セキュリティ試験ツールによる検出結果を人手で判断しなくても，容易にパラメタ改ざん脆弱性を検出することができる手法が求められる．

そこで，本論文では，上記の課題を解決するために，ユーザから HTTP を介して Web アプリケーションに渡されるパラメタの中から，セキュリティ上の重要な意味を持つパラメタを自動で抽出し，抽出されたパラメタの改ざんが成功しているかどうかを判断する手法を提案する．

3.2 提案手法の構成

提案手法では，セキュリティ上の重要な意味を持つパラメタを自動で抽出し，抽出されたパラメタの改ざんが成功しているかどうかを判断する． 図 3-1 に提案手法を組み込んだセキュリティ試験ツールの構成を示す．図 3-1 において，色付けされている部分が，提案手法によって既存のセキュリティ試験ツールの構成に新たに加わる構成要素である．

提案手法では，パラメタ改ざん脆弱性の検出精度を向上させるため，従来のセキュリティ試験ツールの構成に，6 つのデータベースと，3 つの処理を加えている．以下に，それぞれについて説明する．

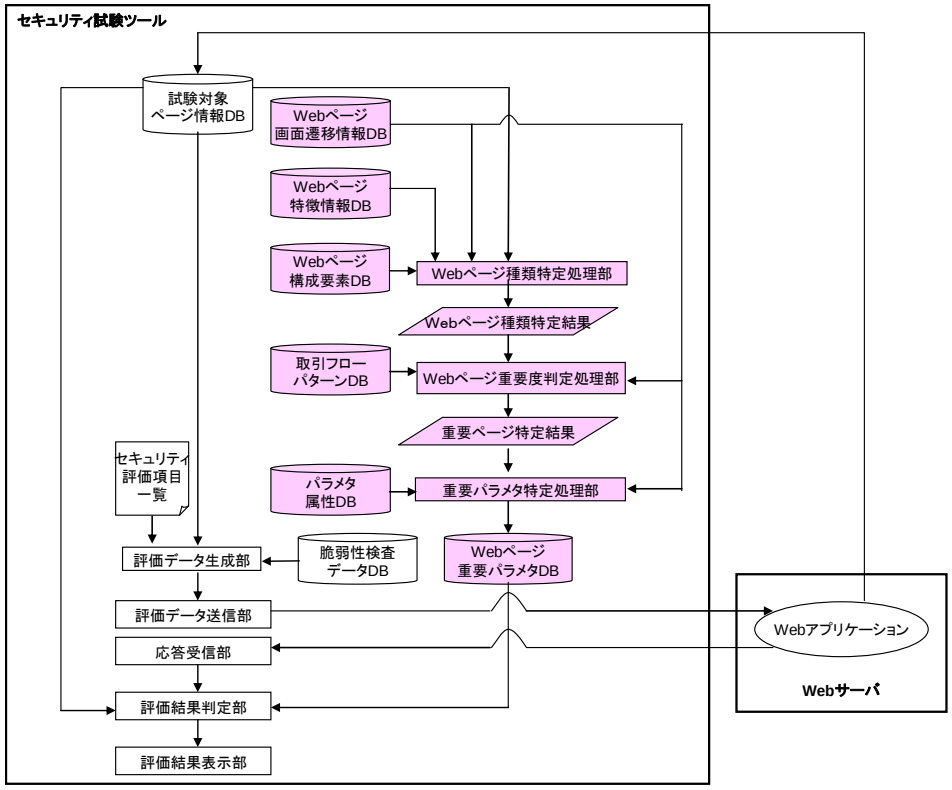

図 3-1 提案手法を組み込んだセキュリティ試験ツールの構成

データベース

(1) **Web ページ画面遷移情報 DB**
Web サーバから収集した Web ページに記載されているページのリンク関係を抽出した Web ページの画面遷移情報を格納しておくデータベース．Web ページの画面遷移情報は，各ページを頂点とし，リンク関係をエッジとした有向グラフで生成する．

(2) **Web ページ特徴情報 DB**
Web ページの種類を特定するための特徴情報を格納しておくデータベース．特徴情報とは，Web ページの種類別に，Web ページに表れる特徴となる条件と，条件

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

別に重み付けしたスコアを定義した情報である．例えば，商品購入サイトなどにある，購入決済完了ページの特徴として，(1) ページまでのリンクの長さが n 以上（n は自然数）：スコア=10，(2) 印刷ボタンの表示がある：スコア=1，(3) メールアドレスの入力フォームがある：スコア=2 といった情報が格納されている．

(3) **Web ページ構成要素 DB**
Web ページの構成要素を特定するための構成要素特定文字列を格納したデータベース．例えば，「印刷」という Web ページ構成要素であれば，構成要素特定文字列として「print」という文字列が，「メール」という Web ページ構成要素であれば，「mail」という文字列が含まれているというように定義しておく．

(4) **取引フローパターン DB**
Web アプリケーションで提供される各取引フローのパターンを格納したデータベース．取引フローとは，Web アプリケーションで提供される Web ページのリンク関係を定義したものである．Web ページの種類ごと，リンク元とリンク先の情報を関連付けしておき，試験対象の Web サイトが，どのような取引を行うサイトであるのかを判定するために用いる．例えば，Web ページ種類が，購入フォーム入力ページである場合，リンク元の Web ページ種類は，商品情報表示ページであり，リンク先の Web ページ種類は，入力情報の確認ページであると定義しておき，このようなリンク関係にある取引フローは，商品購入フローであると定義しておく．また，取引フローにおいて，重要な Web ページ種類には，重要フラグを立てておく．例えば，商品購入サイトであれば，購入決済完了ページへリンクしている，取引の入力情報の確認ページが重要な Web ページ種類となる．

(5) **パラメタ属性 DB**
取引フローごと重要となるパラメタ文字列を格納したデータベース．取引種別ごとに，重要なパラメタ文字列が定義されている．例えば，取引フローが商品購入である場合には，「price」，「product」，「address」といった文字列を含むパラメタが，重要パラメタであると定義しておく．

(6) **Web ページ重要パラメタ DB**
セキュリティ試験対象である Web アプリケーションが提供している取引フローにおいて，重要な Web ページの URL，ページに含まれる form タグの action，重要パラメタ名を格納しておくデータベース．

処理

(1) **Web ページ種類特定処理**
Web ページ画面遷移情報 DB, Web ページ特徴情報 DB, Web ページ構成要素 DB, の情報を基に，試験対象ページの Web ページ種類を特定する．

(2) **Web ページ重要度判定処理**
取引フローパターン DB を基に，試験対象となる一連の取引フローにおける重要ページを特定する．

(3) **重要パラメタ特定処理**
Web ページ重要度判定処理の出力である重要ページ特定結果に含まれる，取引フローパターンと重要ページからパラメタ属性 DB を参照して，重要パラメタを特定する．

3.3 提案手法を組み込んだセキュリティ試験ツールの動作

次に，提案手法を組み込んだセキュリティ試験ツール全体の処理の流れについて，(1) 準備，(2) Web ページ種類特定処理，(3) Web ページ重要度判定処理，(4) 重要パラメタ特定処理，(5) 試験実施の 5 つのステップに分けて説明する．

(1) **準備**
① Web アプリケーションによって提供される Web ページの一覧の取得し，試験対象ページ情報 DB に格納
② 取得した Web ページの一覧から画面遷移情報を生成
③ 生成した画面遷移情報を Web ページ画面遷移情報 DB へ格納

(2) **Web ページ種類特定処理**
① 試験対象ページ情報 DB から Web アプリケーションの応答メッセージ（Web ページ）を読み込む
② 試験対象ページの URL を基に，Web ページ画面遷移情報 DB を参照して，TOP ページからのリンクの長さを計算
③ 得られた TOP ページからのリンクの長さを基に，Web ページ特徴情報 DB を参照し，リンクの長さが条件と一致する Web ページ種類を検索
④ TOP ページからのリンクの長さの特徴に当てはまる Web ページ種類が検索で抽出された場合は，特徴の当てはまる Web ページ種類のスコアを加算
⑤ Web ページ画面遷移情報 DB から試験対象ページのリンク先 URL を抽出

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

⑥ 抽出したリンク先URLの文字列を基にWebページ構成要素DBを検索して，試験対象ページの構成要素を特定
⑦ Webページ画面遷移情報DBから抽出した全てのリンク先URLについて①~⑥の処理を実施して構成要素を特定
⑧ 特定した構成要素を基にWebページ特徴情報DBを検索し，構成要素の特徴が当てはまるWebページ種類が存在した場合には，特徴の当てはまるWebページ種類のスコアを加算
⑨ 試験対象ページの構成要素全てに関してWebページ特徴情報DBを検索して算出されたWebページ種類のスコアが，予め定められた閾値を超えているかどうかを判定し，スコアが閾値を超えたWebページ種類あれば，検査対象ページのWebページ種類であると判定
⑩ 閾値を超えたWebページ種類が複数あった場合や，閾値を超えたWebページ種類が無かった場合は，スコアが高いものを選択し，Webページ種類と判定
⑪ 複数同じスコアであった場合には，試験対象ページは複数Webページ種類の特徴を持つものと判定
⑫ ①~⑪を試験対象ページ情報DBに含まれる全ての試験対象ページに対して行い，Webページ種類を特定

(3) Webページ重要度判定処理

① Webページ種類特定処理の出力である試験対象ページのWebページ種類特定結果から，試験対象ページのURLと，そのWebページ種類を選択
② 得られた試験対象ページのURLを基に，Webページ画面遷移情報DBを検索し，試験対象ページのリンク元URLとリンク先URLを取得
③ 試験対象ページのリンク元URLとリンク先URLのWebページ種類を，Webページ種類特定部の出力から特定
④ 特定された試験ページのWebページ種類を基に，取引フローパターンDBを検索して，取引フローパターンと重要フラグを取得
⑤ 取引パターンが特定された場合，試験対象ページ，取引フローパターン，重要フラグを，取引フロー特定結果として記憶
⑥ ①~⑤の処理を全ての試験対象ページについて行い，取引フローパターンを特定
⑦ 取引フローパターンの特定結果から，重要フラグが1である試験対象ページを，取引フロー上重要ページとして抽出し，取引フローパターンと重要ページを，重要ページ特定結果として出力

(4) 重要パラメタ特定処理

① Webページ重要度判定処理の出力である重要ページ特定結果を参照し，重要ページに含まれている全てのformタグを抽出
② 抽出したformタグ内のaction（リンク先ページ）の値を基に，Webページ画面遷移情報DBを参照し，リンク先ページへリンクしている全てのページを抽出
③ リンク先ページへリンクしている全てのページを抽出した結果から，リンク先ページへリンクしているページが重要ページのみからリンクしているページへのformタグを特定
④ Webページ重要度判定処理で特定された取引フローパターンを基に，パラメタ属性DBを検索して重要パラメタ文字列を抽出し，重要ページのfromタグ内に含まれているパラメタと重要パラメタ文字列を比較
⑤ 比較の結果，重要パラメタ文字列と一致するパラメタ名がformタグ内に存在した場合，そのパラメタを重要パラメタと判定
⑥ fromタグ内に存在する全てのパラメタ名について比較を行った後に，比較結果として重要パラメタ（複数可）を出力
⑦ 出力された重要ページのURL，重要パラメタが含まれているformタグのactionの値，重要パラメタをWebページ重要パラメタDBに格納

(5) 試験実施

① 試験対象ページに含まれる全ての重要ページについてWebページの重要パラメタの特定処理が完了した後に，Webサーバ上で動作するWebアプリケーションの試験を開始
② 試験対象ページ情報DB，及び，脆弱性検査データDBの情報を基に，試験対象ページに対する試験に必要な評価データを生成
③ 作成した評価データを試験対象ページ別に評価データ送信部からWebアプリケーションに対して送信
④ 送信した評価データに対するWebアプリケーションからの応答メッセージを応答受信部で受信
⑤ 評価結果判定部では，応答受信部で受信した応答メッセージと，試験対象ページ情報DBに格納されているWebアプリケーションへ正常にアクセスした際のWebアプリケーションからの応答メッセージとを比較し，送信した評価データによってWebアプリケーションの応答メッセージに変化が見られたかどうかを判定
⑥ 評価結果判定部において，パラメタを改ざんした評価データを送信した場合のWebアプリケーションの応答メッセージと，Webページ画面遷移情報

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

DB に格納されている Web アプリケーションの遷移情報を比較

⑦ エラーページなど正常ページとは異なるページへ遷移しなかった場合，パラメタの改ざん攻撃に成功したと判定

⑧ Web ページ重要パラメタ DB を参照し，改ざん攻撃に成功したパラメタが重要パラメタかどうか判定

⑨ 改ざん攻撃に成功したパラメタが重要パラメタであった場合には，セキュリティ上重要なパラメタの改ざん攻撃に成功したものとして検出

⑩ 重要パラメタでなかった場合は，パラメタ改ざん攻撃の成功とは見なさない

⑪ 評価結果判定部で判定された結果は，評価結果表示部へ表示することで，ユーザへ通知

⑩で，「改ざん攻撃に成功したパラメタが重要パラメタでなかった場合は，パラメタ改ざん攻撃の成功とは見なさない」のは，⑦で，改ざんに成功したと判定されたパラメタが重要パラメタでなかった場合は，改ざんされてもセキュリティ上のリスクが低いパラメタか，改ざんされたパラメタを Web アプリケーションが検出し，エラー処理したことを意味するためである．

4. 考察

3 章では，提案手法が解決する課題，提案手法の構成，及び，提案手法を組み込んだセキュリティ試験ツールの動作について述べた．本章では，提案手法の効果について考察する．

提案手法の特長は，Web アプリケーションが提供する取引フロー上，重要なページに含まれているパラメタ改ざん脆弱性の検査対象となる重要パラメタを自動で絞り込む点である．これを実現するため，試験対象となる Web アプリケーションから収集した Web ページの一覧から，Web ページの画面遷移情報を生成し，個々の Web ページの種類を特定した上で，Web ページの一連の繋がりから Web アプリケーションが提供する取引フローを特定し，取引フロー上，重要なページなページから重要パラメタを特定している．

既存のパラメタ改ざん脆弱性の試験では，パラメタが改ざんされた評価データを送信した場合の Web アプリケーションの応答メッセージが，Web ページ画面遷移情報ＤＢに格納されている Web アプリケーションの遷移情報と比較した結果，エラーページなど正常ページとは異なるページへ遷移しなかった場合，改ざんされてもセキュリティ上のリスクが低いパラメタの改ざんや，後の処理に影響を与えないパラメタの改ざんまで検出してしまう．

提案手法では，重要パラメタを事前に絞り込んでおくことによって，既存のパラメタ改ざん脆弱性の試験において発生する誤検出（修正が不要なパラメタ改ざんの成功の検出）をなくすことができるため，人手による検出結果の判断が不要となる．

加えて，パラメタ改ざん脆弱性の検査対象となる重要パラメタを，Web アプリケーションの設計情報とは，別の観点から自動で特定するため，セキュリティ試験を実施する開発者に対して，Web アプリケーションの設計やセキュリティ試験についての専門的な知識を要求しない．これにより，Web アプリケーションの設計やセキュリティ試験についての専門的な知識を持たない開発者であっても，Web アプリケーションのセキュリティ試験を実施できるようになり，Web アプリケーションの開発段階においてセキュリティ試験の実施が可能となるため，Web アプリケーションの動作試験完了後にセキュリティの脆弱性が見つかることによって Web アプリケーションの開発工程を遡って脆弱性を修正しなければならないといった状況の発生を低減できる．

提案手法の課題としては，Web ページ種類の判定，取引フローの判定，重要パラメタの判定の各処理において参照するデータベースへ予め格納しておいたパターンを辞書的に用いている点が挙げられる．辞書的な判定のアプローチは，高速かつ明確にそれぞれの判定を行える反面，各データベースに登録されていない特殊なケースへ柔軟に対応することができない．各データベースに予め格納しておくパターンの種類を充実化させることで，対応できる Web アプリケーションの範囲を広げることは可能であるが，人手がかかってしまうとともに，全てのパターンを網羅することは，現実的ではないといえる．

上記の課題を解決するためには，Web ページに含まれている各パラメタが，Web アプリケーション上でどう処理されるのか，Web アプリケーションのコード自体を静的に解析し，その結果を基に，パラメタ改ざん脆弱性の検査対象とする重要パラメタを判定するといったアプローチと組み合わせた手法を，今後考えていく必要があると考えられる．

5. 関連研究

Web アプリケーションの脆弱性検出手法としては，大きく分けて，動的解析手法（ブラックボックス試験）と，静的解析手法（ホワイトボックス試験）がある．

動的解析手法は，動作中の Web アプリケーションに対して擬似的な攻撃を行い，脆弱性を検出する手法である．提案手法は，動的解析手法の 1 手法であり，また，市販されているセキュリティ試験ツール [3)4)]や，フリーで公開されているセキュリティ試験ツール [5)6)]などの多くは，動的解析手法に分類される．

Vol.2011-DPS-146 No.36
Vol.2011-CSEC-52 No.36
2011/3/11

一方の静的解析手法は，Web アプリケーションのソースコードを検査することによって脆弱性を検出する手法である [7].

既存の動的解析手法としては，文献 8)がある．文献 8)では，Web アプリケーションのセキュリティに関する検証を，より正確かつ効率的に行うことを目的として，Amberate と呼ばれるフレームワークが提案されている．文献 8)では，Amberate 上に SQL インジェクション攻撃，クロスサイト・スクリプティング，JavaScript Hijacking に対する脆弱性を検出するプラグインを実装している．

提案手法は，基本的に，動的解析手法における一連の流れの中でセキュリティ上重要なパラメタのパラメタ改ざん脆弱性を検出する手法である．したがって，既存の動的解析手法に分類される多くのセキュリティ試験ツールに組み込んで動作させることができる．

6. まとめ

本論文では, Web アプリケーションの脆弱性の 1 つであるパラメタ改ざん脆弱性を，高精度かつ効率的に検出する手法について述べた．

これまで，セキュリティ試験を自動化したツールを用いてセキュリティ試験を実施した場合，検出された項目の中には誤検知が含まれていることがあり，ツールが検出した脆弱性がセキュリティ上の脅威となりうるかどうかの最終的な判断は，人手によって行わなければならないため，セキュリティ試験の実施者には，専門的な知識が求められていた．特に，パラメタ改ざん脆弱性に関しては，一見，パラメタ改ざんに成功したように見えても，改ざんに成功したパラメタの重要性によって，その後の処理でセキュリティ上の不具合が発生するかどうかの結果は異なってくるため，人手による判断が難しいことが課題であった．

提案手法では, ユーザから HTTP を介して Web アプリケーションに渡されたパラメタの中から，セキュリティ上の重要な意味を持つパラメタを自動で抽出し，抽出された重要パラメタの改ざんが成功しているかどうかを判断する．提案手法を用いることにより，セキュリティ上の重要な意味を持つパラメタのパラメタ改ざん脆弱性のみ検出できるようになった．これにより，セキュリティ試験ツールによる検出結果を人手で判断する必要がなくなるため，Web アプリケーションのセキュリティ試験に関する専門的な知識を持たない開発者であっても容易にパラメタ改ざん脆弱性を検出することができるようになる．

今後は，提案手法を実装して有効性を検証するとともに，静的解析手法との組み合わせによって，より柔軟に，Web ページに含まれている各パラメタの中から重要パラメタを絞り込む手法について検討していく．

参考文献

1) Top Cyber Security Risks - Executive Summary, SANS, http://www.sans.org/top-cyber-security-risks/summary.php
2) 安全な Web サイトの作り方, 独立行政法人 情方処理推進機構（IPA）, http://www.ipa.go.jp/security/vuln/websecurity.html
3) AppScan, IBM, http://www-06.ibm.com/software/jp/rational/products/test/appscan/
4) WebInspect, HP, https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-201-200^9570_4000_306__
5) WebScarab, The Open Web Application Security Project (OWASP), http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project
6) Paros, Chinotec Technologies Company, http://www.parosproxy.org/
7) 小黒博昭, 市原直久, 道坂修: Web アプリケーション脆弱性発見のためのソースコード診断ツールの開発, 情報処理学会研究報告, 2008-CSEC-41 (17), (2008).
8) 小菅祐史, 河野健二: 効果的な攻撃テストによる Web アプリケーションの脆弱性検出手法, 情報処理学会研究報告, Vol.2009-ARC-183, No.11 (2009).