SRA GROUP
GSLetterNeo Vol.37
2011年8月

Web アプリケーションのセキュリティについて

シニアコンサルタント
渡邊　雄一
Yuichi Watanabe
yuichi@sra.co.jp

◆はじめに

残念なことに、昨今では Web アプリケーションのセキュリティに起因する話題（事件、報道）に事欠きません。それらは何か新しい技法によるものなのでしょうか？　確かに新たな脆弱性脅威は増えていますが、**現実の被害の多くは「既知の脆弱性」に適切な対処を怠った結果**でしかありません。例えば、IPAから発行されている報告書にも「指摘された脆弱性の改修対策がなかなか進んでいない」旨の記述があります。このように、確実な対応策があるにも関わらず、その対処がなされていないために、結果として被害が広がっているのが現状です。本稿ではWebアプリケーションのセキュリティについて、改めてその対策を整理・解説してみたいと思います。

◆Web アプリケーションの脆弱性

「アプリケーションの脆弱性」といえば「メモリーリーク」、「バッファ・オーバーフロー」が知られています。しかし「Web アプリケーションの脆弱性」は、それらも含めて、**『Web アプリケーションの特性の盲点を突くような攻撃に耐えられる堅牢性を持ち合わせているか？』**が問題です。具体的には「SQL インジェクション」、「クロスサイト・スクリプティング」などへの耐性です。個々の解説は他に譲るとして、問題の本質を整理してみましょう。

サーバーで稼働する Web アプリケーションは ″http″ あるいは ″https″ によって、ユーザが操作する「端末」と相手先の「サーバー」との間で『通信』を行うことで、データが送受信されて目的の操作が行われます。ところがその経路上は、不特定の第3者に「盗聴」されているかもしれませんし、『通信』の途中で、最初の「端末」を装った第3者が関わってくるかもしれません。このような**姑息な、盲点を突くような攻撃に耐えられるように「サーバー」を作り込んでおく必要がある**のです。（これを称して「Web アプリケーションの脆弱性対策」とも言います）

◆有効な対策とは？

最近の事件の影響で、「セキュリティ」をテーマにしたセミナーや雑誌の解説記事が増えています。しかし、「それって Web アプリケーションの脆弱性対策の本質とは関係ないよね？」と思われる説明を受けることも少なくありません。ここでは要らぬ誤解を生まぬように、問題の本質に有効なことを、きちんとご説明したいと思います。

1.　きちんと作られていれば良い

当たり前の話ですが、Web アプリケーションが“きちんと作られていれば良い”のです。であるならば、“どのように作られているのか？”をつぶさに確認することです。このような確認（テスト）方法としてコードレビューがあり、ホワイトボックス・テストがあります。この手のテストの目的は「要件xxにある、機能yyは、きちんと動作するか？」という「機能検証」が主でした。しかし脆弱性・セキュリティについては、それとは別の意味での「セキュリティ要件」（＝非機能要件）が規定されるべきです。具体的には「何を、どのようにして、守るのか？」を明記し、「それに基づいた実装（設計ならびにソースコードの作成）がなされているのか？」を確認します。

2.　きちんと作られているか？

“きちんと作られているか？”を調べるには、「これはいけない！」という事例を基に、不適切なソースコードの記述パターンを洗い出し、修正・対処することが最善の策です。これにはツールが威力を発揮します。これを支

援するツールは強力で、これまでの脆弱性についての知見が詰まっているのです。

・HP Fortify SCA

・IBM Rational AppScan Source Edition

が代表的なツールです。

3. 本当にきちんと作られているか？

子供の頃に聞いた怪談噺に「耳なし芳一」があります。耳だけお経を書き忘れたために耳をもがれたという、ちょっと残酷な話です。芳一の身体にお経を書いた和尚は、出かける時に耳にお経を書かなかったことを見落としてしまったのですね。

ソフトウェア開発では「結合テスト」、「総合テスト」をして見落としを含めて最終確認します。この Web アプリケーション版に「ペネトレーション・テスト」と言われるテスト方法があります。これは「実際に攻撃して既知の脆弱性が無いかを調べましょう」というものです。例えて言えば「矛盾」の関係です。万全な「ホワイトボックス・テスト」を踏まえて作ったWebアプリケーション（盾）を、実際に突いて試してみましょう（「ペネトレーション・テスト」＝矛）というわけです。これは**極めて重要なテスト**であると思いませんか？

実際にその通りだと思うのですが、実はこのテストはそれほど簡単では無いのです。一番の理由は、テストに掛ける費用（コスト）が如実に問題になってくるからです。まず、本番相当のサーバーを準備する必要があります。何故なら、脆弱性があると Web アプリケーションの不備により内部の DB のデータにゴミが入ることが予想されるからです。

「何を、どのようにテストすべきか？」について、皆さんはイメージ出来ますか？　これは実は簡単で、「可能性のある画面遷移（シナリオと称することが多いです）」を網羅するように、テスト計画を立て、実施すればよいのです。ここで多くの場合、設計と実装の不一致が露見します。しかし、その多くは機能要件レベルの話で、問題は非機能要件（脆弱性への耐性）にあります。ツールを使ったペネトレーション・テストでは、最新の攻撃パターンに基づいてサーバーを攻撃します。http（あるいは https）の文字列を送り、「サーバー」からの返答結果を調べて、それが不適当であれば脆弱性の可能性として検知・報告します。どのような文字列を送るかが重要で、知られている攻撃パターンから「Web アプリケーションの特性の盲点を突くような攻撃」のありとあらゆるものが送られます！

こちらのツールも、ホワイトボックス・テストのそれに劣らず強力で、

・IBM Rational AppScan Standard Edition

・HP WebInspect

が代表的なツールです。

4. まとめ

本稿では Web アプリケーションの脆弱性対策の基本を説明しました。上述のように有効なツールもあり、あとは如何に開発・保守のフェーズに適用してゆくかです。

セキュリティ対策については、これ以外にも多くのことが唱えられています。それらは間違いではありませんが、表面的な手段に目を奪われて、本稿で解説した「基本」が疎かになってしまっては本末転倒です。例えば、「多重防御をした場合、一番弱い（敷居の低い）ところが、セキュリティーレベルになる」といった解説を誤解して、全体のレベルの底上げを気にされる方がいらっしゃいます。確かにその観点も必要ですが、まずは守るべきもの、すなわち**Web アプリケーションに脆弱性が無いことを継続的に確認するとともに、脆弱性を作り込まない開発体制を維持継続・展開してゆくことが、重要**なことです。

SRAでは、ツールの導入・活用や開発プロセスへのコンサルティングまで、各種サービスを提供しております。お気軽にお問合せ下さい。

夢を。

GSLetterNeo Vol.37
2011 年 8 月 20 日発行
発行者●株式会社 SRA 産業開発第 1 事業部
編集者●土屋正人、柳田雅子、野島勇

ご感想・お問い合わせはこちらへお願いします●gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋２－３２－８

夢を。Yawaraka Innovation
やわらかいのべーしょん