管理番号＃２０－０３

平成２０年度
成果報告書

静的及び動的解析の組み合わせによる Web アプリケーションのセキュリティ診断システムに関する研究開発

委託先：　㈱NST

平成２１年４月

情報通信研究機構

平成２０年度　成果報告書

「静的及び動的解析の組み合わせによるWebアプリケーションのセキュリティ診断システムに関する研究開発」

目　次

１　研究開発課題の背景

インターネットの爆発的普及により、電子商取引を初めとしたWebサービスは企業から一般消費者まで幅広く普及している。これらWebサービスの背後にはWebアプリケーションと呼ばれる、サーバサイドのソフトウェアが実行されている。

Webサービスが拡大する中で、開発業者数の増加から競争が激化し、開発業者間の価格競争はより激しくなっている。さらに、海外でのオフショア開発も拡大の傾向にあり、国内開発業者の多くは開発受注額の低減を常に迫られている。

このような環境下で開発されたWebアプリケーションには劣悪な品質のものもあり、悪意あるユーザーの格好の標的となりつつある。特に、データベース等の重要な情報にアクセス可能なWebアプリケーションは、非常に危険な状態のまま運用されているものも少なくない。

Webアプリケーションの危険性を診断する技術、サービスは完全に人依存の状態から抜け出せていない。診断コストは少しずつ下がってはいるものの、Webアプリケーションの数に対して、セキュリティ診断を実施しているサイトは非常に少ないのが現状である。より安価なサービスを提供するには、診断技術そのものの自動化が必要であり、技術的にもサービス的にも自動化への流れが進んでいる。

現在の主たる診断方式としては、擬似攻撃診断とソースコード診断が実施されているが、それぞれ良い面、悪い面を持ち合わせており、人が介在しなければ正確な診断を実施することはできず、自動化を実現するためにはいくつかの課題をクリアしなければならない。

２　研究開発の全体計画

２－１　研究開発課題の概要

ア．網羅性向上

主要な診断方式である擬似攻撃診断は、ソースコードの網羅性が極端に低くなることがある。これは内部構造等に一切関与せず、予め用意されたパターンに従って診断を行っていることに原因がある。ソースコードにおける、特に分岐条件の網羅性を確保できないことが原因であり、ソースコード全体の網羅性を確保するには、ソースコードの解析をベースにした静的解析を組み入れる必要がある。本研究開発ではWebアプリケーションのセキュリティに特化した問題検知が可能なソースコード診断の手法に関して研究開発を行う。一般的に静的解析では70%以上のソースコード網羅率を達成でき、より広い範囲での問題検知を可能にする。

イ．誤検知の減少

一般的に静的解析では多くの誤検知が検出される。診断実施時に誤検知が大量発生すると生産効率を大幅に落とす結果となる。本提案技術においては、静的解析であるソースコード診断と動的解析である擬似攻撃診断を組み合わせることにより、ソースコード診断実施時に発生する誤検知を抑制させる手法についての研究を行う。

ウ．問題箇所の特定、絞り込み

ソースコード診断の段階で発見された問題は、発生箇所を絞り込むことが可能であるが、ブラックボックステストである擬似攻撃診断では、内部状態を把握できないことから、問題の発生要因を分析することは難しい。本研究開発では、擬似攻撃診断実施時に内部状態の把握を可能にする手法の研究を行い、問題の検知しか行えなかった擬似攻撃診断の実施時にも問題箇所の特定を行える手法の確立を目指す。

エ．修正支援

本研究開発では、最終的に静的解析及び動的解析の何れにおいても、問題発生時の実行トレースとデータの流れを追跡することが可能となる。これにより、問題箇所を如何に修正するかという点についても研究を行い、修正支援に関わる技術の研究及び開発を行う。

２－２　研究開発の最終目標（平成２２年９月末）

1. ソースコード診断ツールの研究開発
   （1）診断実施時のソースコード網羅率が 80%～90%を達成すること
   本件研究開発ではソースコード網羅率が低くなりやすい擬似攻撃診断実施前に、予めソースコード診断を実施することで、擬似攻撃診断のパターン生成が内部構造を把握した状態で実行できる。これにより擬似攻撃診断実施時のソースコード網羅性を向上させることが可能となる。一般的なソースコード診断では 70%～90%程度が網羅性の平均であり、特に Web アプリケーションの入出力に特化した静的解析では、80%～90%程度の網羅率を確保できると考える。
   （2）ソースコード診断における現状 50%の誤検知率を 30%以下に抑えること
   ソースコード診断実施時に発生し易い誤検知を分析し、誤検知の発生頻度が高い問題に対しては、擬似攻撃診断実施時に検証することで、ソースコード診断実施時の誤検知を減少させる。
   （3）擬似攻撃診断における現状 30%の誤検知率を 10%以下に抑えること
   擬似攻撃診断実施時に判定しづらい問題点をソースコード診断により検証することで、相対的な誤検知発生率を減少させる。

2. 擬似攻撃診断と実行時内部トラッキングによる問題検知ツールの研究開発
   （1）擬似攻撃診断によって検知された問題に対して制御フローを追跡可能なこと
   本研究では擬似攻撃診断実施前に、ソースコードの各制御点に、実行時の追跡を可能とするデータ出力を行う為の関数ラッピングを行う。これにより、擬似攻撃診断を仕掛けた際の、各テストパターンに対してどのような制御フローが発生したのかを追跡することが可能となる。
   （2）擬似攻撃診断によって検知された問題に対してデータフローを追跡可能なこと
   本研究では擬似攻撃診断実施前に、ソースコード内の主要 API に対して動的に用意する検証用 API へと置換を行い、プログラム実行時にコールされた各 API への値出力を実施することで、各テストパターンに対してどのようなデータのながれが生じたのかを追跡することが可能となる。

3. 問題箇所の特定及び修正支援ツールの開発
   （1）問題発生時の制御フロー、データフローを視覚化すること
   本研究ではソースコード診断または擬似攻撃診断で検知された問題に対して、制御フロー及びデータフローの分析が可能となる。問題発生時の制御フローを視覚化することで問題箇所の特定を支援する。
   （2）修正支援機能を提供すること
   本研究では各種問題毎に一般的な修正案を示し、さらに上記制御フロー及びデータフローの結果を用いることで、どのような修正を実施すれば該当の問題を抑制できるのかといった修正案を示し、静的診断、動的診断の診断サイクルを繰り返す中で、問題の修正確認を行える。

２－３　研究開発の年度別計画

（金額は非公表）

研究開発項目	20 年度	21 年度	22 年度	計	備　考
静的及び動的解析の組み合わせによる Web アプリケーションのセキュリティ診断システムに関する研究開発					
ア　ソースコードの診断技術に関わる研究	→	→		－	20 年度 10 月から 9 カ月
イ　ソースコード診断と擬似攻撃診断の相互連携手法の研究		→	→	－	21 年度 6 月から 1 年間
ウ　擬似攻撃診断と実行時内部トラッキングによる問題検知手法の研究		→	→	－	21 年度 9 月から 1 年間
エ　問題箇所の特定及び修正支援に関わる研究				－	22 年度 4 月から 6 カ月
間接経費	－	－	－	－	
合　　計	－	－	－	－	

注）１　経費は研究開発項目毎に消費税を含めた額で計上。また、間接経費は直接経費の３０％を上限として計上（消費税を含む。）。
　　２　備考欄に再委託先機関名を記載
　　３　年度の欄は研究開発期間の当初年度から記載。

３　研究開発体制

３－１　研究開発実施体制

研究代表者：久田雅之　研究所長
研究項目：
　　脆弱性検知を目的とした静的解析手法
　　擬似攻撃診断実施時の網羅性向上
　　擬似攻撃診断実施時の効果的テストパターン生成
　　擬似攻撃診断とソースコード診断の相互連携
研究分担者：Raymond Wu　研究所　主任研究員
研究項目：
　　脆弱性検知の多言語対応
　　擬似攻撃診断実施時の網羅性向上
　　トラッキング方法の確立・検証
　　トラッキングデータの解析法
　　脆弱性発生時の原因特定
研究分担者：張志強　研究所　研究員
研究項目：
　　脆弱性検知を目的とした静的解析手法
　　擬似攻撃診断実施時の網羅性向上
　　擬似攻撃診断実施時の効果的テストパターン生成
　　脆弱性発生時の原因特定
研究分担者：大石節　研究所　研究員
研究項目：
　　擬似攻撃診断実施時の網羅性調査
　　脆弱性検知の多言語対応
　　擬似攻撃診断実施時の網羅性向上
　　トラッキング方法の確立・検証
　　トラッキングデータの解析法
　　擬似攻撃診断とソースコード診断の相互連携
研究分担者：川本剛正　本社　研究員
研究項目：
　　脆弱性検知を目的とした静的解析手法
　　擬似攻撃診断実施時の網羅性調査
　　擬似攻撃診断実施時の網羅性向上
　　擬似攻撃診断実施時の効果的テストパターン生成
　　擬似攻撃診断とソースコード診断の相互連携
研究分担者：坂崎公一　本社　研究員
研究項目：
　　脆弱性検知を目的とした静的解析手法
　　擬似攻撃診断実施時の網羅性調査
　　擬似攻撃診断実施時の網羅性向上
　　擬似攻撃診断実施時の効果的テストパターン生成
　　脆弱性発生時の原因特定

研究指導者：加羅　淳　奈良工業高等専門学校　情報工学科　教授
研究指導者：程　子学　会津大学　教授　産学イノベーションセンター長

共同研究先：
会津大学コンピュータ産業学講座(Incheon Paik 准教授)

４　研究開発実施状況

４－１　ソースコード診断ツールの研究開発

４－１－１　PHP ソースコードに関する解析手法の研究

４－１－１－１　概要

Web アプリケーションのセキュリティ診断方式としては、アプリケーションを実行した状態で、既知の脆弱性パターンを送りその反応を解析する擬似攻撃診断が一般的である。このような診断方式は入出力に特化した解析であり、Web アプリケーションで問題を引き起こしやすいデータフローに起因する問題の検知に特に有効である。

内部状態に関与せず入力及び出力のみに着目したブラックボックス型の診断では、特定のアプリケーションコンポーネントにおける問題の内在確認は可能であるが、ブラックボックス故に問題の原因、発生箇所等の情報は一切わからない。

本研究では、擬似攻撃診断実施時にプログラム実行時の情報（制御フロー、データフロー）を取得可能にすることで、問題が発生した際に内部状態を把握可能にする手法の確立を目指した。

４－１－１－２　実施状況

本研究では、Web アプリケーションの実装言語として人気の高い PHP 言語で書かれたプログラムに特定の処理（改変）を施すことにより、上記診断実施時の内部状態を把握可能にし、問題が発現した際の制御フローを特定する為の技術開発を行った。本研究開発により、Web アプリケーションのソースコードを自動改変し、擬似攻撃診断を実施した際に各テストにおいて制御フローを特定することが可能となった。

また、診断実施時の内部状態把握を可能にすることで、擬似攻撃診断実施時に各テストケースがどの程度の分岐条件を網羅しているか評価が可能となった。

複数の Web アプリケーションに対して擬似攻撃診断を実施した場合に、どの程度の網羅率が得られるのかを、商用診断ツールで調査したところ、平均的に５０％～６０％の分岐網羅率であることがわかった。通常の診断範囲においては、分岐網羅を完全には得られないことがわかった。故に、診断実施時に網羅されなかった分岐経路において脆弱性が含まれたとしても、当該診断では検知ができないこととなる。

研究の実施内容、詳細については特許申請を控えている為、現時点では公開できないが、当該技術に関連する内容は世界最大規模のコンピュータ科学に関する国際会議 WORLDCOMP2009 の ICOMP'09 に投稿し採択を受けており、外部専門委員による査読の結果、その成果が認められたと言える。

４－１－１－３　今後の課題

現在は PHP 言語に特化した技術であるが、下記４－１－２に関する技術適用により、多言語対応を目指す。言語対応のみならず、Web アプリケーション作成時に欠かすことのできない、様々なフレームワークに対して本技術をどのように適用させるかを今後の課題として設定している。

現時点での分岐網羅に関する解析は、全分岐条件に対して判定を行っている。各制御点での可達性判定をしておらず、網羅率は正確な値とは言えない。擬似攻撃診断実施時にどの程度ソースコードの網羅性を得られたのかは、診断を実施する上でも重要な情報となり得る。より正確な網羅率評価を今後の課題としたい。

４－１－２　ソースコードの抽象化に関する研究

４－１－２－１　概要

Web アプリケーションは、Java、PHP、Ruby 等様々な言語で実装される。脆弱性の検知に関わる解析手法はどの言語に対しても適用可能であり、言語依存性は無い。しかしながら、コード解析の下層においては言語解釈が必要であり、字句解析及び構文解析に関わる部分は言語依存性がある。

ソースコードの解析や改変にあたって、中核となるエンジンを各種言語に対応させる為に、解析エンジンと各種言語の間にコード抽象化のための１層を設け（ユニバーサルアダプタと呼ぶ)、解析エンジンの言語依存性を排除することで、各種言語仕様の変更や新規対応言語の追加が容易になる。

本研究では、PHP 言語と Java 言語を共通エンジンで解析する為にユニバーサルアダプタの設計構築を行った。

４－１－２－２　実施状況

オープンソースの統合開発環境 Eclipse 上で、PHP 言語、Java 言語の開発が可能である。統合開発環境には、プログラムに含まれる関数名（メソッド名）や変数等がビジュアルに表示され、内部的に各言語を解釈している。下層（言語解釈）では言語依存性があるが、上層（可視化）では共通化されており、本研究でも Eclipse 内で実装されている機能の流用が可能かどうかを調査した。

Eclipse 上での Java 開発環境には JDT と呼ばれるフレームワークが、PHP 開発環境には PDT と呼ばれるフレームワークが、字句解析から構文解析までを行っており、Java 及び PHP の構文木をほぼ同じ形式で構築できることがわかった。Java 及び PHP の構文解析木における相違は表記の違いであり、簡易的な１層の抽象化によって吸収が可能な程度であり、Java 及び PHP で書かれたプログラムを共通表記の構文木上で解析できるようになった。

４－１－２－３　今後の課題

今回は PHP 言語の構文解析木を Java 側の表記に合わせる形で変換を行ったが、他の言語も含めた抽象化の為に構文解析木のフォーマットを改めて定義し直し、全ての言語において表記可能な形式を再検討する予定である。

４－１－３　脆弱性検知手法の研究

４－１－３－１　概要

Web アプリケーションのセキュリティ診断方式としては、動的解析としての擬似攻撃診断と、静的解析としてのソースコード診断に大別される。

擬似攻撃診断はその仕組み上、アプリケーションの稼働が必要であり、開発工程の後よりでしか実施できない。問題発見時の修正コストは、開発の後工程になるに従って非線形に増大する傾向にあり、擬似攻撃診断によって発見された問題の修正コストは必然的に高くなる傾向にある。

近年、Web アプリケーションのソースコードから脆弱性の検知、問題特定をする技術開発が行われており、一部製品化されている技術もある。擬似攻撃診断に比べて、開発の早期に問題の検知、修正を実施することができ、修正に関わるコスト効率も高い。

本研究では、Web アプリケーションのデータフローに起因する問題を効果的に発見する

為の既存手法とその問題点について調査、検証を行った。

４－１－３－２　実施状況

PHP 言語は人気が高く、多くの Web サイトで使用されている。既存手法の検証に使う Web アプリケーションのサンプルも Java 言語に比べれば豊富なことから、本研究においては PHP 言語で書かれた Web アプリケーションを対象言語に選定した。

検証に使う Web アプリケーションとしては、オープンソースプロジェクトを中心に選定し、脆弱性が発現するよう意図的にプログラムを組み換えた。

Web アプリケーションで発生する脆弱性の多くは、入力チェックまたは出力チェックの不足によるもので、クロスサイトスクリプティングや SQL インジェクションが代表的例としてあげられる。データフローに起因する問題には、汚染データ解析と呼ばれる手法が用いられ、PHP 言語を対象とした汚染データ解析ツールとして、Web サイトの参照状況及び論文調査から Pixy を選定した。

Pixy は、Technical University of Vienna、Secure Systems Lab により開発され、PHP 言語で書かれた Web アプリケーションを対象に、クロスサイトスクリプティング及び SQL インジェクションの脆弱性検知が可能なツールである。字句解析及び構文解析は汎用的なものが用いられており、構文解析木を生成した上で制御フロー解析を行い、入力と出力を結ぶ特定の実行フローにおいて、予め想定した API（関数）を通過しているかどうかを解析している。全入力データが汚染データであるという仮定の基に、出力状態において汚染データが到達しているかどうかをチェックしている。

静的解析では誤検知（False Positive）が多くなる傾向にあり、Pixy でも５０％程度の誤検知が発生した。Pixy で発生している誤検知についてそれぞれ原因を調査したところ、誤検知の大半はデータの 2 次利用に関連する誤検知であった。

データベースやファイルからの読み込みで、入力されるデータが絶対に汚染されない事が確定されている場合、該当の実行フローにおいて出力までにデータチェックが存在しなかったとしても何ら問題は発生しない。Pixy における誤検知の大半は、DB また File からの読み込みにおいて、全入力データが汚染データと仮定されることにより発生していた。

よって、汚染データ解析実施前に入力値の汚染可否判定を実施することで、誤検知を大幅に低減させられる可能性が非常に高いことが判明した。

Pixy における汚染データ解析手法は PHP 以外の言語にも適用可能である。オープンソースのソースコード解析ライブラリ WALA を用いて、Java 言語に対応した汚染データ解析エンジンを作成中であり、最終的には同エンジンを用いて、複数言語の汚染データ解析を実施可能にする計画となっている。

４－１－３－３　今後の課題

DB や File に含まれるデータが、同一アプリケーションによってアクセスされるという前提であれば、書き込まれているデータが汚染可能であるかどうかは、依存関係の解析から可能であると思われる。DB や File を経由した、アプリケーション内での入出力に関して依存関係をどのように構築、解析するかを今後の課題として設定した。

４－２　総括

平成 20 年度における研究開発の主な内容は、「サブテーマ：ソースコード診断ツールの研究開発」に関わるものであり、特に Web アプリケーションの実装言語として人気が高く、言語自体の簡易性も高い PHP 言語を対象に関連する研究開発を実施した。

Web アプリケーションの脆弱性診断としては一般的な擬似攻撃診断について、診断実施後の問題解決に役に立つ技術として、診断実施時の各テストケースそれぞれに該当する制

御フローの特定、診断実施時のソースコード分岐網羅率を評価できるようになった。これにより、診断実施後の修正コストが大幅に削減され、また診断自体の品質チェックにも非常に有効な技術である。本技術は言語依存性が高い為、言語依存性の排除を目的としたソースコードの抽象化について、Java 言語及び PHP 言語に適用可能な抽象構文木生成ライブラリを作成し、解析エンジンの言語依存性排除を実現できた。多言語対応も下層の言語解釈部分を作成すれば、ほぼ全ての Web アプリケーション実装言語に対して適用可能であり、継続的に言語対応を進めていく予定である。

上記、擬似攻撃診断に関わる技術に加え、より開発の前工程で実施可能な Web アプリケーションのソースコードに対する汚染データ解析について、PHP 言語に対応したオープンソースツール Pixy について、評価、検証を行い、誤検知の低減に向けた解決指針の決定に至った。オープンソースの技術、ライブラリを有効活用しながら、多言語対応可能な汚染データ解析エンジンを次年度の研究開発において完成させる計画となっている。

現時点で平成 20 年度における当初計画に対して大きな遅れは無く、計画において未達成な項目については平成 21 年度の研究開発において早期に達成される予定であり、全体に及ぼす影響は特にない。

５　参考資料・参考文献

５－１　研究発表・講演等一覧

なし